COLING 2012

24th International Conference on Computational Linguistics

Proceedings of COLING 2012: Technical Papers

Program chairs:
Martin Kay and Christian Boitet

8-15 December 2012
Mumbai, India

Diamond sponsors

Tata Consultancy Services
Linguistic Data Consortium for Indian Languages (LDC-IL)

Gold Sponsors

Microsoft Research
Beijing Baidu Netcon Science Technology Co. Ltd.

Silver sponsors

IBM, India Private Limited
Crimson Interactive Pvt. Ltd.
Yahoo
Easy Transcription & Software Pvt. Ltd.

Proceedings of COLING 2012: Technical Papers
Martin Kay and Christian Boitet (eds.)
Revised preprint edition, 2012

Published by The COLING 2012 Organizing Committee
Indian Institute of Technology Bombay,
Powai,
Mumbai-400076
India
Phone: 91-22-25764729
Fax: 91-22-2572 0022
Email: pb@cse.iitb.ac.in

Also available online in the ACL Anthology at http://aclweb.org

Introduction from the Program Chairs

The members of the International Committee on Computational Linguistics were immensely privileged to be able to accept the invitation to hold our 24th COLING Conference here in India, a country which can justly be thought of as the center of the linguistic world. This is where Panini wrote the first formal grammar in the 6th century BC and where a linguistic diversity flourishes today that is nothing short of astounding to the rest of us. This conference has received twice as many submissions as any of its predecessors and, in many ways, is twice as rich because of the high proportion of contributions by teachers, researchers and, above all, students. Many are from India and other countries, such as Iran, with long and diverse linguistic traditions. There are challenges here for linguists of all varieties, most especially for those who put their faith in n-grams and machine learning.

The 195 full-length technical papers in 5 parallel tracks, 138 posters, and 66 demonstrations that will be presented still constitute no more than a quarter of the total number of submissions. The chairs of some of the 26 program subcommittees were overwhelmed with both their number and their quality. The International Committee is always greatly indebted to the area chairs and reviewers for the invaluable work that they do. Never so much as on this occasion.

Our greatest debt is clearly to our colleagues here in Mumbai, as will become clear to all as the week proceeds. They were even less well equipped than we on the permanent committee to predict what they were getting into, but they have risen to the occasion in every way and you will find them to be immensely warm, helpful, and resourceful hosts.

COLING's founding fathers wanted these conferences to be more than learned presentations. They wanted them to be opportunities to meet, and talk and delight in the company of other who share our fascination with language and the processes that make it work. Some call this the COLING spirit. There is nowhere that could nurture this spirit more effectively than here in India.

Martin Kay
Christian Boitet
(Program chairs)
December 2012, Mumbai

Introduction from the Organizing Chairs

It is a matter of great pride that the 24^{th} International Conference on Computational Linguistics (COLING 2012) is taking place in India, the land of multilinguality and multiculure. The organization of an event of COLING's scale takes enormous energy, planning and time. Two years back, in Beijing, when COLING was awarded to India, we knew that the task will be demanding, and happily for us, the NLP team at IIT Bombay, the organzing institute, has risen to the occasion.

At the time of going to press, the total number of registrants in COLING has exceeded 700. With delegates coming from 60 countries, COLING 2012 will witness a colourful diversity of language and culture, and fittingly so. Conforming to current practices of international conferences, there are two days of workshops and tutorials before the main conference and one day of workshop immediately after. 15 focussed and topical workshops will be attended by about 300 delegates, as will be 6 high quality tutorials of contemporary interest.

Social events include a reception in the evening of 10^{th} Dec, a banquet in a nearby 7 star hotel on 11^{th} Dec, an excursion to the famed Bhaja caves on 12^{th} Dec and a cultural evening of Indian classical music on 13^{th} Dec. Bhaja Caves, built in the period 3^{rd} century BC to 2^{nd} century CE, is a set of Buddhist monastery-caves near the hill station of Lonavala, nestled in the Sahayadri mountain ranges, about 90 km to the south-east of Mumbai. There will be cultural evening on the fourth day of the conference, featuring a solo performance on "tabla", the representative of Indian percussion instruments, and another solo on Sitar, an instrument that drew world's attention Indian classical music tradition.

Indian Institute of Technology Bombay is fittingly the host of COLING 2012. IITs have, over the years, emerged as the premier institutes of technology in India. The Computer Science and Engineering Department at IIT Bombay is one of the largest and oldest Departments of CSE in the country. Each and every member of the 40 strong NLP group at IIT Bombay is toiling hard to make COLING 2012 a resounding success.

The Government and industries have been our generous sponsors. All their names and logos are to be found in printed and USB proceedings. We thank them wholeheartedly.

Technology Development in Indian Languages (TDIL) project of Department of IT, Ministry of Communication and Information Technology, has been the harbinger of growth of NLP in India. COLING happening in India is a result of this long history of active patronage.

Logistics wise, the “large events” – inauguration, invited speeches, reception and the cultural program – are in the convocation hall of IIT Bombay. Oral presentations are all in the newly constructed Victor Menezes Convention Center (VMCC) about 200 mtrs from the convocation hall. Poster presentations are in the convocation hall, except on the first day, when it is VMCC.

A very competent team of volunteers will be available for any assistance. We hope COLING participants will have a memorable time in India.

Pushpak Bhattacharyya
Rajeev Sangal
(Organizing chairs)
December 2012, Mumbai

Program Committee:

Program Chair: Martin Kay (Stanford University)
Program Co-chair: Christian Boitet (University of Grenoble)
Workshop Chair: Prof. Laurent Besacier
Tutorial Chair: Prof. Sadao Kuroshashi

Area Chairs:

Indian language technology: Dipti Sharma (IIIT Hyderabad)
Underresourced languages: Vincent Berment (C&S, Paris & LIG, GETALP, Grenoble)
Morphology & POS Tagging: Gábor Prószéky (MorphoLogic & Pázmány Péter Catholic University, Budapest)
Grammar and formalisms: Hans Uszkoreit (DFKI, Saarbrücken)
Parsing: Mark Johnson (Macquirie Univ., Sydney)
Semantics: Igor Boguslavsky (RAS, IPPI/IITP, Moscow & UPM, Madrid)
Discourse and pragmatics: Eva Hajičova (Charles Univ., Prague)
Coreference analysis resolution: Alexander Gelbukh (Instituto Politécnico Nacional, Mexico-city)
Ontologies and terminology: Christophe Roche (Univ. de Savoie, Chambéry)
Textual Entailment: Lauri Karttunen (Stanford University)
Resources and annotation: Nicoletta Calzolari (CNR – ILC, Pisa)
Psychological and neurological modelling: Véronique Aubergé CNRS – LIG lab Grenoble) & Rohit Manchanda (IITB, Mumbai)
Empirical Machine Translation: Philipp Koehn (University of Edinburgh)
Expert or Hybrid Machine Translation: Mandel Shi (Xiamen University)
Hybrid man+machine architectures & human factors: Hervé Blanchon (Université de Grenoble (LIG, GETALP))
Information Retrieval: Jian-Yun Nie (Univ. de Montréal (RALI))
Summarization: Horacio Saggion (Universitat Pompeu Fabra) & Sivaji Bandyopadhyaya (Jadavpur University)
Named Entity recognition: Sergei Nirenburg (Univ. of Maryland, Baltimore)
Word Sense Disambiguation: Mathieu Lafourcade (Univ. de Montpellier II (LIRMM))
Sentiment and text classification: Yorick Wilks (Univ. of Sheffield)
Information & content extraction, text mining: Junichi Tsujii (MSRA, Beijing)
Question Answering: Constantin Orasan (University of Wolverhampton)
Speech recognition and synthesis: Roland Kuhn (National Research Council of Canada, Ottawa)
Software internationalization & localization: Andy Way (Univ. of Dublin and Applied Language)
Deployment of NLP-based applications, software integration & quality: Rajeev Sangal (IIIT Hyderabad) & Christian Boitet (Université de Grenoble (UJF, LIG, GETALP))
Natural Language Generation: Donia Scott (Univ. of Sussex)

Invited Speakers:

Prof. Paul Kiparsky (Stanford University)
Prof. Makoto Nagao (Kyoto University)
Prof. Dipti Misra (Sharma IIIT-Hyderabad)
Prof. Barbara Moser-Mercer (University of Geneva)

Organizing Committee:

Organizing Chair: Pushpak Bhattacharrya
Organizing Co-Chair: Rajeev Sangal
Vasant Zende
Rupash Modak
Ganesh Ramakrishnan
Balamurali A R
N. Vasudevan
Rahul Sharnagat
Raj dabre
Jaya Sarswati
Rajita Shukla
Gajanan Rane
Samir Soni
Ritesh Shah (CDAC Mumbai)
Deepak Jagtap
Anup Kunchukuttan
Abhijeet Mishra
Brijesh Bhatt
Swapnil Choudhary
Laxmi Kashyap
Kahsyap Popat
Manish Shrivastava

Publication committee:

Publication chair: Roger Evans (University of Brighton)
Ms. Sudha (IIT Bombay)
Mrs. Rajita Shukla (IIT Bombay)
Mr. Rahul Sharnagat (IIT Bombay)
Mr. Kashyap Popat (IIT Bombay)

List of the 723 reviewers for COLING-2012, Mumbai

Eneko Agirre, University of the Basque Country, Spain.
Lupe Aguado, Universidad Politécnica de Madrid, Spain.
Khurshid Ahmad, Trinity College, University of Dublin, Ireland.
Rania Al-Sabbagh, University of Illinois at Urbana-Champaign, United States.
Vicente Alabau, Universitat Politècnica de València, Spain.
Inaki Alegria, University of the Basque Country, Spain.
Laura Alonso Alemany, Universidad Nacional de Córdoba, Argentina.
Le An Ha, Univ. Wolverhampton, United Kingdom.
Sophia Ananiadou, University of Manchester, United Kingdom.
R Ananthakrishnan, IBM Research - India, India.
Ion Androutsopoulos, Athens University of Economics and Business, Greece.
Gabor Angeli, Stanford, United States.
Marianna Apidianaki, LIMSI-CNRS, France.
Eiji Aramaki, The university of Tokyo, Japan.
Karunesh Arora, CDAC, India.
Nicholas Asher, CNRS, IRIT, Université Paul Sabatier, Toulouse, France.
Corinne Astesano, Univeristé de Toulouse, UTM, Laboratoire Octogone-Lordat, France.
Véronique Aubergé, CNRS, Grenoble (LIG, GETALP), France.
Tania Avgustinova, Saarland University, Germany.
Julia Aymerich, Pan American Health Organization, United States.
Wilker Aziz, University of Wolverhampton, United Kingdom.
Bruno Bachimont, Université de Technologie de Compiègne, France.
B Lakshmi Bai, IIIT Hyderabad, India.
Collin Baker, International Computer Science Institute, United States.
Timothy Baldwin, Affiliation unknown, United States.
Kalika Bali, Microsoft Research Labs India, India.
Rafael E Banchs, Institute for Infocomm Research, Singapore.
Sivaji Bandyopadhyay, Jadavpur University, India.
Sivaji Bandyopadhyay, Jadavpur University, India.
Carmen Banea, University of North Texas, United States.
Srinivas Bangalore, ATT, India.
Eva Banik, Computational Linguistics Ltd, United Kingdom.
Regina Barzilay, MIT, United States.
Núria Bel, Universitat Pompeu Fabra, Spain.
Valérie Bellynck, Université de Grenoble (UJF, LIG, GETALP), France.
Emily M. Bender, University of Washington, United States.
Shane Bergsma, Johns Hopkins University, United States.
Vincent Berment, CS, Paris & Inalco, Paris & GETALP (LIG, UJF), Grenoble, France.
Nicola Bertoldi, FBK, Italy.
Laurent Besacier, Université de Grenoble (UJF, LIG, GETALP), France.
Steven Bethard, University of Colorado Boulder, United States.
Rajesh Bhatt, UMass Amherst, United States.
Pushpak Bhattacharyya, CFILT, IIT Bombay, India.
Kristín Bjarnadóttir, The Arni Magnusson Institute for Icelandic Studies, Iceland.
Patrick Blackburn, Roskilde University, Denmark.
Herve Blanchon, Univ. of Grenoble (UPMF, LIG, GETALP), France.
Victor Bocharov, Saint-Petersburg State University, Russian Federation.
Igor Boguslavsky, RAS, IPPI/IITP, Moscow & UPM, Madrid, Russian Federation.
Christian Boitet, Université de Grenoble (LIG, GETALP), France.

Ondrej Bojar, Charles University, Prague, Czech Republic.
Danushka Bollegala, University of Tokyo, Japan.
Anastasia Bonch-Osmolovskaya, Higher School of Economics, philology department Moscow, Russian Federation.
Francis Bond, Nanyang Technological University, Singapore.
Kalina Bontcheva, University of Sheffield, United Kingdom.
Lars Borin, University of Gothenburg, Sweden.
Vladimir Borschev, University of Massachusetts, United States.
Johan Bos, University of Groningen, Netherlands.
Nadjet Bouayad-Agha, University Pompeu Fabra, Spain.
Florian Boudin, Université de Nantes, France.
Mohand Boughanem, IRIT, CNRS, France.
Gosse Bouma, University of Groningen, Netherlands.
Paolo Bouquet, University of Trento, Italy.
António Branco, University of Lisbon, Portugal.
Pavel Braslavski, Kontur labs, Russian Federation.
François Brown de Colstoun, Lingua et Machina, France.
Rebecca Bruce, University of North Carolina @ Asheville, United States.
Gerhard Budin, University of Vienna, Austria.
Paul Buitelaar, DERI, National University of Ireland, galway, Ireland.
Harry Bunt, Tilburg Uiversity, Netherlands.
Miriam Butt, University of Konstanz, Germany.
Lynne Cahill, University of Brighton, United Kingdom.
Shu Cai, USC/ISI, United States.
Nicoletta Calzolari, CNR, ILC, Pisa, Italy.
Erik Cambria, National University of Singapore, Singapore.
Nick Campbell, TCD, Ireland.
Yunbo Cao, Microsoft Research Asia, China.
Jesus Cardeñosa, Universidad Politecnica de Madrid (Spain), Spain.
Claire Cardie, Cornell University, United States.
Michael Carl, Copenhagen Business School, Denmark.
Marine Carpuat, National Research Council Canada, Canada.
Francisco Casacuberta, Universitat Politècnica de València, Spain.
Eric Castelli, International Research Institute MICA - CNRS, Viet Nam.
Daniel Cer, Stanford - NLP Group, United States.
Özlem Çetinoğlu, IMS, University of Stuttgart, Germany.
Vineet Chaitanya, IIIT Hyderabad, India.
Baobao Chang, Peking University, China.
Eugene Charniak, Brown University, United States.
Jacques Chauché, lirmm Montpellier France, France.
Wanxiang Che, Harbin Institute of Technology, China.
Yu-N Cheah, Universiti Sains Malaysia, Malaysia.
Ying Chen, China Agricultural University, China.
Wenliang Chen, I2R, Singapore, Singapore.
Jiajun Chen, Nanjing University, China.
Boxing Chen, National Research Council, Canada.
Yidong Chen, Xiamen University, China.
Colin Cherry, National Research Council Canada, Canada.
Jean-Pierre Chevallet, Université de Grenoble (UJF, LIG, MRIM), France.
Jean-Pierre Chevrot, Lidilem, Université Stendhal, Institut Universitaire de France, France.
David Chiang, USC/ISI, United States.
Jen-Tzung Chien, National Chiao Tung University, Taiwan.

Manoj Chinnakotla, Relevance and Data Sciences Team, Bing, Microsoft, India.
Raymond Chiong, Swinburne University of Technology, Australia.
Key-sun Choi, KAIST, Republic of Korea.
Jinho Choi, University of Massachusetts Amherst, United States.
Monojit Choudhury, Microsoft Research Lab India, India.
Khalid Choukri, ELRA/ELDA, France.
Ken Church, IBM, United States.
Massimiliano Ciaramita, Google, Switzerland.
Philipp Cimiano, University of Bielefeld, Germany.
Michael Collins, Columbia University, United States.
Sherri L Condon, The MITRE Corporation, United States.
José Carlos Cortizo Pérez, BrainSINS, Spain.
Rute Costa, CLUNL - Universidade Nova de Lisboa, Portugal.
Dan Cristea, Al. I. Cuza University of Iasi, Romania.
Xiaodong Cui, IBM T. J. Watson Research Center, United States.
Aron Culotta, Northeastern Illinois University, United States.
Hamish Cunningham, University of Sheffield, United Kingdom.
Iria da Cunha, Universitat Pompeu Fabra, Spain.
Walter Daelemans, CLiPS, University of ANtwerp, Belgium.
Ido Dagan, Bar-Ilan University, Israel.
Beatrice Daille, Université de Nantes - LINA, France.
Om Damani, IIT Bombay, India.
Luc Damas, Université de Savoie, France.
Sandipan Dandapat, CNGL, School of Computing, Dublin City University, Ireland.
Laurence Danlos, University Paris DIderot, France.
Kareem M Darwish, QF, Qatar.
Amitava Das, NTNU, Norway.
Marie-Catherine de Marneffe, Stanford University, United States.
Maarten de Rijke, University of Amsterdam, Netherlands.
Koenraad De Smedt, ULB, Norway.
Thierry Declerck, DFKI, Language Technology Lab, Germany.
Steve DeNeefe, SDL Language Weaver, United States.
Pascal Denis, INRIA, France.
Tejaswini Deoskar, University of Edinburgh, United Kingdom.
Heidi Depraetere, CrossLang NV, Belgium.
Leon Derczynski, University of Sheffield, United Kingdom.
Alain Désilets, National Research Council of Canada, Canada.
Barbara Di Eugenio, Univ. of Illinois at Chicago, United States.
Gaël Dias, University of Caen Basse-Normandie, France.
Fernando Diaz, Microsoft, United States.
Alberto Diaz, Universidad Complutense de Madrid, Spain.
Mike Dillinger, TOPs Globalization Consulting, United States.
Bill Dolan, Microsoft Research, United States.
Zhendong Dong, Canada Keentime Inc., Canada.
Sophie Donnadieu, Université de Savoie, France.
Iustin Dornescu, University of Wolverhampton, United Kingdom.
Mark Dras, Macquarie University, Australia.
Markus Dreyer, SDL Language Weaver, United States.
Jinhua Du, Faculty of Automation and Information Engineering, Xi'an University of Technology, China.
Xiangyu Duan, Institute for Infocomm Research, Singapore, Singapore.
Pablo Duboue, Les Laboratoires Foulab / Universite de Montreal, Canada.

Jonathan Dunn, Purdue University, United States.
Georges Dupret, Yahoo! Labs, United States.
Chris Dyer, Carnegie Mellon University, United States.
Kurt Eberle, Lingenio GmbH, Germany.
Terumasa Ehara, Yamanashi Eiwa College, Japan.
Jacob Eisenstein, Georgia Institute of Technology, United States.
Jason Eisner, Johns Hopkins University, United States.
Asif Ekbal, IIT Patna, India.
Michael Elhadad, Ben Gurion University, Israel.
Jeremy Ellman, Northumbria University, United Kingdom.
Jakob Elming, Copenhagen Business School, Denmark.
Micha Elsner, Ohio State University, United States.
Brigitte Endres-Niggemeyer, Noapps, Germany.
Chantal Enguehard, LINA, University of Nantes, France.
Tomaz Erjavec, Jozef Stefan Institute, Slovenia.
Katrin Erk, University of Texas at Austin, United States.
Xavier Blanco Escoda, Universitat Autònoma de Barcelona, Spain.
Emmanuelle Esperança-Rodier, UJF LIG Getalp, France.
Jérôme Euzenat, INRIA & LIG, France.
Roger Evans, University of Brighton, United Kingdom.
Achille Falaise, LIG-GETALP, France.
Ji Fang, Medallia.com, United States.
Atefeh Farzindar, NLP Technologies Inc., Canada.
Christiane Fellbaum, Princeton University, United States.
Zhiwei FENG, Institute of Applied Linguistics, The Ministry of Education, China.
Oscar Ferrandez, Nuance Communications, Inc., United States.
Antonio Ferrandez, University of Alicante, Spain.
Gabriela Ferraro, Universitat Pompeu Fabra, Spain.
Dan Flickinger, CSLI, Stanford University, United States.
Radu Florian, IBM, United States.
Ray Flournoy, Adobe, United States.
Corina Forascu, Univ. Al.I. Cuza of Iasi, Faculty of Computer Science, Romania.
George Foster, National Research Council Canada, Canada.
Gil Francopoulo, CNRS-LIMSI-IMMI + Tagmatica, France.
Anette Frank, Universität Heidelberg, Germany.
Guohong Fu, School of Computer Science and Technology, Heilongjiang University, China.
Piotr Fuglewicz, TiP Sp. z o. o., Poland.
Atsushi Fujii, Tokyo Institute of Technology, Japan.
Robert Gaizauskas, University of Sheffield, United Kingdom.
Michel Galley, Microsoft, United States.
Suryakanth Gangashetty, International Institute of Information Technology Hyderabad, India.
Jianfeng Gao, Microsoft Research, Redmond, United States.
Claire Gardent, CNRS/LORIA UMR 7503 Nancy, France.
Éric Gaussier, Univ. J. Fourier, France.
Alexander Gelbukh, Instituto Politécnico Nacional, Mexico.
Josef Van Genabith, Dublin City University, Ireland.
Kim Gerdes, Sorbonne Nouvelle & Chinese Academy of Sciences, France.
Salvatore Giammarresi, PayPal, United States.
George Giannakopoulos, NCSR Demokritos, Greece.
Dafydd Gibbon, Universität Bielefeld, Germany.
Daniel Gildea, University of Rochester, United States.
Kevin Gimpel, Carnegie Mellon University, United States.

Filip Ginter, University of Turku, Finland.
Corina R Girju, UIUC, United States.
Oren Glickman, Unaffiliated, Israel.
Asunción Gómez Pérez, Universidad Politecnica de Madrid, Spain.
Jerome Goulian, LIG - GETALP, France.
Cyril Goutte, National Research Council Canada, Canada.
Vishal Goyal, Department of Computer Science, Punjabi University Patiala, India.
Jorge Gracia, Universidad Politécnica de Madrid, Spain.
Didier Grandjean, University of Geneva, Switzerland.
Daniel Grasmick, Lucy Software and Services GmbH, Germany.
Brigitte Grau, LIMSI-CNRS, France.
Gregory Grefenstette, 3DS Exalead, France.
Nikolai Grigoriev, Yandex, Russian Federation.
Ralph Grishman, New York University, United States.
Iryna Gurevych, UKP Lab, Technische Universität Darmstadt and DIPF, Germany.
Louise Guthrie, University of Brighton, United Kingdom.
Barry Haddow, University of Edinburgh, United Kingdom.
Eva Hajicova, Charles University, Prague, Czech Republic.
Tanmay Haldankar, OC, IITB, India.
Keith Hall, Google Research, United States.
Rejwanul Haque, Applied Language Solutions, India.
Sanda Harabagiu, University of Texas at Dallas, United States.
Christian Hardmeier, Uppsala University, Sweden.
Tony Hartley, Tokyo University of Foreign Studies, Japan.
Sven Hartrumpf, SEMPRIA GmbH, 40237 Düsseldorf, Germany.
chikara hashimoto, NICT, Japan.
Jun Hatori, Apple Inc., United States.
Petter Haugereid, University of Haifa, Israel.
Katsuhiko Hayashi, NAIST, Japan.
Xiaodong He, Microsoft Research Redmond, United States.
Jing He, University of Montreal, Canada.
Kenneth Heafield, University of Edinburgh, Carnegie Mellon University, United States.
Sigrún Helgadóttir, The Árni Magnússon Institute for Icelandic Studies, Iceland.
Christian Hempelmann, Texas A&M-Commerce, United States.
James Henderson, Xerox Research Centre Europe, France.
Iris Hendrickx, Centro de Linguística da Universidade de Lisboa, Portugal.
Jesus M. Hermida, University of Alicante, Spain.
Nicolas Hernandez, University of Nantes, France.
Huang Heyan, Department of Computer Science and Technology,Beijing Institute of Technology, China.
Amanda Hicks, University at Buffalo, United States.
Djoerd Hiemstra, University of Twente, Netherlands.
Erhard Hinrichs, Tuebingen University, Germany.
Hieu Hoang, University of Edinburgh, United Kingdom.
Thomas Hoar, Precision Translation Tools Co., Ltd., Canada.
Julia Hockenmaier, University of Illinois at Urbana-Champaign, United States.
Jim Hogan, Queensland University of Technology, Australia.
Kristy Hollingshead, Department of Defense, United States.
Fred Hollowood, Symantec, CNGL, Ireland.
Matthew Honnibal, Macquarie University, Australia.
Ales Horak, Masaryk University, Czech Republic.
Véronique Hoste, University College Ghent, Belgium.

Eduard Hovy, Carnegie Mellon University, United States.
Chang Hu, University of Maryland, United States.
Yunhua Hu, Affiliation unknown, China.
Liang Huang, CUNY, United States.
Shujian Huang, Nanjing University, China.
Jimmy Huang, York Univ., Canada.
Chu-Ren Huang, Affiliation unknown, China.
Matthias Huck, RWTH Aachen University, Germany.
Samar Husain, University of Potsdam, Germany.
Sarmad Hussain, KICS, Pakistan.
Nancy Ide, Vassar College, United States.
Adrian Iftene, Al. I. Cuza University of Iasi, Faculty of Computer Science, Romania.
Ryu Iida, Tokyo Institute of Technology, Japan.
Kentaro Inui, Tohoku University, Japan.
Leonid Iomdin, Institute for Information Transmission Problems, RAS, Russian Federation.
Boris Iomdin, Russian Language Institute, RAS, Russian Federation.
Radu Ion, Research Institute for Artificial Intelligence, Romanian Academy, Romania.
Pierre Isabelle, National Research Council, Canada.
Hitoshi Isahara, NiCT, Japan.
Mustafa Jarrar, Mustafa Jarrar, Sina Institute, Birzeit University, Palestine, Palestine.
Girish Jha, Jawaharlal Nehru University, New Delhi, India.
Heng JI, City University of New York, United States.
Jie Jiang, Applied Language Solutions, United Kingdom.
Wenbin Jiang, Institute of Computing Technology, CAS, China.
Jing Jiang, Singapore Management University, Singapore.
Narsu JIN, Hangzhou Normal University, China.
Mark Johnson, Macquirie univ., Sydney, Australia.
Howard Johnson, National Research Council, Canada.
Kristiina Jokinen, University of Helsinki and University of Tartu, Finland.
Gareth Jones, Dublin City University, Ireland.
Pamela Jordan, University of Pittsburgh, United States.
Aravind K Joshi, University of Pennsylvania, United States.
Alain Joubert, LIRMM, University of Montpellier, France.
Heiki-Jaan Kaalep, University of Tartu, Estonia.
Sylvain Kahane, Modyco, Université Paris Ouest Nanterre, France.
Nobuhiro KAJI, Tokyo University, Japan.
Hiroshi Kanayama, IBM Research - Tokyo, Japan.
Lauri Karttunen, Stanford University, United States.
Graham Katz, CACI, Inc., United States.
Daisuke Kawahara, Kyoto University, Japan.
Martin Kay, Stanford University, United States.
Junichi Kazama, NiCT, Japan.
Maxim Khalilov, TAUS, Netherlands.
Mitesh Khapra, IBM Research India, India.
Genichiro KIKUI, Okayama Prefectural University, Japan.
Adam Kilgarriff, Lexical Computing Ltd, United Kingdom.
Tracy Holloway King, eBay, United States.
Ewan Klein, University of Edinburgh, United Kingdom.
Kevin Knight, USC/ISI, United States.
Philipp Koehn, University of Edinburgh, United Kingdom.
Natalia Konstantinova, University of Wolverhampton, RIILP, United Kingdom.
Ruud Koolen, Tilburg University, Netherlands.

Valia Kordoni, DFKI GmbH and Saarland University, Germany.
Ioannis Korkontzelos, National Centre for Text Mining, School of Computer Science, The University of Manchester, United Kingdom.
Kimmo Koskenniemi, University of Helsinki, Finland.
Leila Kosseim, Concordia University, Canada.
Milen Kouylekov, Celi S.R.L., Italy.
Zornitsa Kozareva, Information Sciences Institute/University of Southern California, United States.
Elena Kozerenko, Institute of Informatics Problems of the RAS, Russian Federation.
Emiel Krahmer, Tilburg University, Netherlands.
Roland Kuhn, National Research Council, Canada.
Amba Kulkarni, University of Hyderabad, India.
Shankar Kumar, Google, United States.
Oren Kurland, Technion, Israel.
Mathieu Lafourcade, Univ. de Montpellier II (LIRMM), France.
Sobha Lalitha Devi, AU-KBC Research Centre, India.
Mounia Lalmas, Yahoo! Labs, United States.
Lori Lamel, LIMSI-CNRS, France.
Terry Langendoen, National Science Foundation, United States.
Philippe Langlais, Université de Montréal (RALI), Canada.
Guy Lapalme, Université de Montréal (RALI), Canada.
Lynda Tamine Lechani, Institut de Recherche en Informatique (IRIT), France, France.
Alain Lecomte, Université Paris 8, France.
Gary Geunbae Lee, POSTECH, Republic of Korea.
G S Lehal, Punjabi University, Patiala, India.
Alessandro Lenci, University of Pisa, Italy.
Yves Lepage, IPS, Waseda university, Japan.
Lori Levin, Carnegie Mellon University, United States.
Roger Levy, UCSD, United States.
Will Lewis, Microsoft Research, United States.
Haizhou Li, Institute for Infocomm Research, Singapore.
Juanzi Li, Tsinghua University, China.
Fennie Liang, Guy's and St Thomas' Hospital, United Kingdom.
Donghui Lin, Department of Social Informatics, Kyoto University, Japan.
Krister Linden, University of Helsinki, Finland.
Kenneth C Litkowski, CL Research, United States.
Qun Liu, Dublin City University, Ireland & Chinese Academy of Sciences, China, China.
Kang Liu, Institute of Automation, Chinese Academy of Sciences, China.
Xiaohua Liu, Microsoft Research Asia, China.
Yang Liu, University of Texas at Dallas, United States.
Elena Lloret, University of Alicante, Spain.
N. V. Loukachevitch, Lomonosov Moscow State University, Russian Federation.
António Lucas Soares, INESC Porto and University of Porto - Faculty of Engineering, Portugal.
Robert Luk, The Hong Kong Polytechnic University, Hong Kong Special Administrative Region of China.
Yanjun Ma, Baidu, China.
Bill MacCartney, Google, United States.
Bente Maegaard, University of Copenhagen, Denmark.
Mathew Magimai Doss, Idiap Research Institute, Martigny, Switzerland.
Bernardo Magnini, FBK, Italy.
Gudrun Magnusdottir, Chairperson ESTeam AB, Germany.
Prasenjit Majumdar, IIIT Hyderabad, India.
M W Mak, The Hong Kong Polytechnic University, China.

Brian Mak, The Hong Kong University of Science and Technology, Hong Kong Special Administrative Region of China.
Abbas Malik, Faculty of Computing and IT, King Abdulaziz University, Jeddah, Saudi Arabia.
Radhika Mamidi, IIIT Hyderabad, India.
Suresh Manandhar, University of York, UK, United States.
Rohit Manchanda, IITB, Mumbai, India.
Mathieu -nagata Mangeot, GETALP-LIG Laboratory, France.
Prashanth Mannem, LTRC, IIIT-Hyderabad, India.
Christopher Manning, Stanford University, United States.
Stephane Marchand-Maillet, Dept of Computer Science - University of Geneva, Switzerland.
Daniel Marcu, ISI/USC, United States.
Joseph Mariani, LIMSI-CNRS & IMMI, France.
Jose Mariño, Universitat Politècnica de Catalunya, Spain.
Ronaldo Martins, UNDL Foundation, Switzerland.
Roche Mathieu, LIRMM, CNRS, Univ. Montpellier 2, France.
Spyros Matsoukas, Raytheon BBN Technologies, United States.
Yuji Matsumoto, Nara Institute of Science and Technology, Japan.
Kazuyuki Matsumoto, The University of Tokushima, Japan.
Takuya Matsuzaki, National Institute of Informatics, Japan.
Arne Mauser, Google, United States.
Aurélien Max, LIMSI-CNRS, France.
Mark Maybury, Department of Defense, United States.
James Mayfield, Johns Hopkins University, United States.
Diana Maynard, University of Sheffield, United Kingdom.
Joseph F McCarthy, University of Washington Bothell, United States.
David McClosky, Stanford University / IBM Research, United States.
John McCrae, University of Bielefeld, Germany.
Kathleen McKeown, Columbia University, United States.
Paul McNamee, Johns Hopkins University, United States.
Marjorie McShane, University of Maryland Baltimore County, United States.
Karine Megerdoomian, MITRE, United States.
Beáta Megyesi, Uppsala University, Sweden.
Arul Menezes, Microsoft, United States.
Elisabeth Metais, CNAM, France.
Detmar Meurers, Universität Tübingen, Germany.
Haitao Mi, IBM Watson Research Center, United States.
Rada Mihalcea, University of North Texas, United States.
Márton Miháltz, Research Institute for Linguistics, Hungarian Academy of Sciences, Hungary.
Eleni Miltsakaki, University of Pennsylvania, United States.
Jean-luc Minel, Modyco, Université Paris Ouest Nanterre La Défense, France.
Jiri Mirovsky, Charles University, Prague, Czech Republic.
Hemant Misra, Philips Electronics India Limited, India.
Ruslan Mitkov, University of Wolverhampton, United Kingdom.
Mandar Mitra, Indian Statistical Institute, India.
Makoto Miwa, National Centre for Text Mining, The University of Manchester, Japan.
Yusuke Miyao, NII, Japan.
Riichiro Mizoguchi, Osaka University, Japan.
Marie-francine Moens, Hogent, Belgium.
jacques Moeschler, University of Geneva, Switzerland.
Abdel-rahman Mohamed, University of Toronto, Canada.
Karo Moilanen, Department of Computer Science, University of Oxford, United Kingdom.
Dan Moldovan, Univ. Dallas, United States.

Saeedeh Momtazi, Ubiquitous Knowledge Processing Lab (UKP-DIPF), German Institute for Educational Research and Educational Information, Germany.
Monica Monachini, CNR, ILC, Pisa, Italy.
Elena Montiel-Ponsoda, Universidad Politécnica de Madrid, Spain.
Christof Monz, University of Amsterdam, Netherlands.
Robert C Moore, Google, United States.
Lidia Moreno, Universidad Politécnica de Valencia, Spain.
Asuncion Moreno, Universitat Politècnica de Catalunya, Spain.
Alessandro Moschitti, University of Trento, Italy.
Isabelle Moulinier, Thomson Reuters, United States.
Stefan Müller, Freie Universität Berlin, Germany.
Rafael Muñoz, DSI, Univ. Alicante, Spain.
Masaki Murata, Tottori University, Japan.
Yugo Murawaki, Kyoto University, Japan.
Didier Nakache, Affiliation unknown, France.
Preslav Nakov, QCRI, Qatar Foundation, Qatar.
Tahira Naseem, MIT, United States.
Costanza Navarretta, University of Copenhagen, Denmark.
Roberto Navigli, Sapienza University of Rome, Italy.
Anja Nedoluzko, Charles University, Prague, Czech Republic.
Matteo Negri, Fondazione Bruno Kessler - irst, Italy.
Ani Nenkova, University of Pennsylvania, United States.
John Nerbonne, University of Groningen and Freiburg Institute of Advanced Studies, Netherlands.
Guenter Neumann, DFKI, Germany.
Nelson Ng, eBay, United States.
Hwee Tou Ng, National University of Singapore, Singapore.
Vincent Ng, University of Texas at Dallas, United States.
Hong-Thai Nguyen, PolySpot, France.
Tu Anh T. Nguyen, The Open University, UK, United Kingdom.
Jian-Yun Nie, Université de Montréal (RALI), Canada.
Takashi Ninomiya, Ehime University, Japan.
Sergei Nirenburg, Univ. of Maryland, Baltimore, United States.
Malvina Nissim, University of Bologna, Italy.
Zhengyu Niu, Baidu, China.
Joakim Nivre, Uppsala University, Sweden.
Attila Novák, MTA-PPKE Language Technology Research Group – Pázmány Péter Catholic University, Faculty of Information Technology, Budapest, Hungary.
Morgan O'Brien, McAfee, Ireland.
Douglas Oard, University of Maryland, United States.
J E J M Odijk, Utrecht University, Netherlands.
Stephan Oepen, Universitetet i Oslo, Norway.
Kemal Oflazer, Carnegie Mellon University, Qatar.
Jong-hoon OH, NICT, Japan.
Naoaki Okazaki, Tohoku University, Japan.
Manabu Okumura, Tokyo Institute of Technology, Japan.
Constantin Orasan, University of Wolverhampton, United Kingdom.
Csaba Oravecz, Research Institute for Linguistics, Hungarian Academy of Sciences, Hungary.
Daniel Ortiz, Polytechnic University of Valencia, Spain.
Petya Osenova, Sofia University "St. Kl. Ohridski", Bulgaria.
Ekaterina Ovchinnikova, USC ISI, United States.
Sebastian Pado, Heidelberg University, Germany.
Karel Pala, NLP Centre, Faculty of Informatics, Masaryk University, Brno, Czech Republic.

Matteo Palmonari, University of Milan-Bicocca, Italy.
Manuel Palomar, University of Alicante, Spain.
Thiago Pardo, University of São Paulo, Brazil.
Jong C Park, KAIST, Republic of Korea.
Barbara Partee, University of Massachusetts Amherst, United States.
Ranjani Parthasarathi, Anna University, Chennai, India, India.
Marius Pasca, Google Inc., United States.
Siddharth Patwardhan, IBM T. J. Watson Research Center, United States.
Soma Paul, International Institution of Information Technology (IIIT), India.
Bolette Sandford Pedersen, University of Copenhagen, Denmark.
Anselmo Peñas, NLP & IR Group, UNED, Spain.
Sergio Penkale, Applied Language Solutions, United Kingdom.
Gerald Penn, University of Toronto, Canada.
Bhaskararao Peri, International Institute of Information Technology, India.
Wim Peters, University of Sheffield, United Kingdom.
Maxim Petrenko, The International Monetary Fund, Machine Translation Section, United States.
Slav Petrov, Google, Germany.
Christian Pietsch, Bielefeld University, Germany.
Maria Mercedes Piñango, Yale University, United States.
Prasad Pingali, Guest faculty, IIIT Hyderabad and CEO, SETU Software Systems, India.
Mārcis Pinnis, Tilde, Latvia.
Stelios Piperidis, ILSP/Athena RC, Greece.
Tommi Pirinen, University of Helsinki, Finland.
Vito Pirrelli, Institute for Computational Linguistics CNR Pisa, Italy.
Paul Piwek, The Open University, United Kingdom.
Laura Plaza, Universidad Complutense de Madrid, Spain.
Mirko Plitt, Autodesk, Switzerland.
Massimo Poesio, University of Essex, United Kingdom.
Thierry Poibeau, LATTICE-CNRS, France.
Simone Paolo Ponzetto, Universita' di Roma "La Sapienza", Italy.
Hoifung Poon, Microsoft Research, China.
Kashyap Popat, OC, IITB, India.
Octavian Popescu, FBK-HLT, Italy.
Andrei Popescu-Belis, Idiap Research Institute, Switzerland.
Maja Popovic, DFKI, Germany.
Fred Popowich, Simon Fraser University, Canada.
Matt Post, Johns Hopkins University, United States.
Christopher Potts, Stanford, United States.
R Power, The Open University, United Kingdom.
Sameer Pradhan, Raytheon BBN Technologies, United States.
John Prager, IBM T.J. Watson Research Ctr., United States.
Kishore Prahallad, IIIT Hyderabad, India.
Rashmi Prasad, University of Wisconsin-Milwaukee, United States.
Violaine Prince, LIRMM-CNRS/Univ. Montpellier 2, France.
Jean-Philippe Prost, Affiliation unknown, France.
Gábor Prószéky, MorphoLogic & Pázmány Péter Catholic University, Hungary.
Adam Przepiórkowski, Institute of Computer Science, Polish Academy of Sciences, Poland.
James Pustejovsky, Brandeis University, United States.
Sampo Pyysalo, NaCTeM and University of Manchester, United Kingdom.
Vahed Qazvinian, University of Michigan, Ann Arbor, United States.
Chris Quirk, Microsoft Research, United States.
Valeria Quochi, Istituto di Linguistica Computazionale "A. Zampolli", CNR, Italy.

Marta R. Costa-jussà, Barcelona Media Innovation Center, Spain.
Achla Raina, Indian Institute of Technology Kanpur, India.
Owen Rambow, Columbia University, United States.
Balisoamanandray Ranaivo-Malançon, Universiti Malaysia Sarawak, Malaysia.
Aarne Ranta, University of Gothenburg, Sweden.
Umamaheshwara G Rao, IIIT Hyderabad, India.
Victor Raskin, Purdue University, United States.
Paul Rayson, Lancaster University, United Kingdom.
Marta Recasens, Stanford University, United States.
Steve Richardson, The Church of Jesus Christ of Latter-day Saints, United States.
Stefan Riezler, Heidelberg University, Germany.
German Rigau, University of the Basque Country, Spain.
Luca Rigazio, Panasonic, United States.
Brian Roark, OHSU, United States.
Christophe Roche, Université de Savoie, France.
Horacio Rodriguez, UPC, Spain.
Eiríkur Rögnvaldsson, University of Iceland, Iceland.
Hannah Rohde, University of Edinburgh, Linguistics & English Language, United Kingdom.
Laurent Romary, Inria & HUB, France.
Paolo Rosso, Universitat Politècnica de València, Spain.
Dan Roth, University of Illinois at Urbana-Champaign, United States.
Johann Roturier, Symantec, France.
David Rouquet, LIG-GETALP, France.
Bogdan Sacaleanu, IBM Research Ireland, Ireland.
Kenji Sagae, University of Southern California, United States.
Horacio Saggion, Universitat Pompeu Fabra, Spain.
Yoshinori Sagisaka, Waseda University & NICT, Japan.
Tetsuya Sakai, Microsoft Research Asia, China.
Mark Sammons, University of Illinois at Urbana-Champaign, United States.
Felipe Sánchez-Martínez, Universitat d'Alacant, Spain.
Rajeev Sangal, IIIT Hyderabad, India.
Diana Santos, University of Oslo, Norway.
Sudeshna Sarkar, IIT Kharagpur, India.
Anoop Sarkar, SFU, Canada.
Felix Sasaki, DFKI / W3C Fellow, Germany.
Roser Sauri, Barcelona Media Innovation Center, Spain.
Jacques Savoy, University of Neuchatel, Switzerland.
Kevin Scannell, Saint Louis University, United States.
Dag Schmidtke, Microsoft, Germany.
Klaus-Dirk Schmitz, Cologne University of Applied Sciences, Germany.
Didier Schwab, Université de Grenoble, France.
Lane Schwartz, Air Force Research Lab, United States.
Holger Schwenk, LIUM, Université du Mans, France.
Donia Scott, Univ. of Sussex, United Kingdom.
Satoshi Sekine, New York University, United States.
Vladimir Selegey, Russian State University for the Humanities, Russian Federation.
Mark Seligman, Spoken Translation, Inc., United States.
Jean Senellart, Systran, France.
Gilles Sérasset, GETALP-LIG, Université Joseph Fourier, Grenoble 1, France.
Sam Sethserey, Institute of Technology of Cambodia, Cambodia.
Ritesh Shah, OC, IITB, India.
Dipti Sharma, IIIT Hyderabad, India.

Serge Sharoff, University of Leeds, United Kingdom.
Bernadette Sharp, Staffordshire University, United Kingdom.
Wade Shen, MIT, United States.
Svetlana Sheremetyeva, LanA Consulting ApS, Denmark.
Shuming SHI, Microsoft, China.
Xiaodong SHI, Xiamen University, China.
Tomohide Shibata, Kyoto University, Japan.
Hiroyuki Shindo, NTT Communication Science Laboratories, Japan.
Eyal Shnarch, Bar Ilan University, Israel.
Luo Si, Purdue University, United States.
Advaith Siddharthan, University of Aberdeen, United Kingdom.
Grigori Sidorov, Instituto Politecnico Nacional (IPN), Mexico, Mexico.
Khe Chai SIM, National University of Singapore, Singapore.
Khalil Simaan, Institute for Logic, Language and Computation, University of Amsterdam, Netherlands.
Kiril Simov, IICT, Bulgarian Academy of Sciences, Bulgaria.
Navjyoti Singh, IIIT Hyderabad, India.
Smriti Singh, Insideview Technologies (India) Pvt. Ltd., India.
Nandini Chatterjee Singh, National Brain Research Centre, India.
Mukul Sinha, Expert Software Consultants Ltd., India.
Ravi Sinha, University of North Texas, United States.
Milena Slavcheva, Institute of Information and Communication Technologies, Bulgarian Academy of Sciences, Bulgaria.
Raymond Slyh, AFRL, United States.
David Smith, Northeastern University, United States.
Alexei Sokirko, Yandex, Russian Federation.
Ruihua Song, Microsoft Research Asia, China.
Fei Song, University of Guelph, Canada.
Claudia Soria, CNR, ILC, Pisa, Italy.
Virach Sornlertlamvanich, NECTEC, Thailand.
Lucia Specia, University of Sheffield, United Kingdom.
Marcus Spies, Professor, Chair of Knowledge Management, LMU University of Munich, Germany.
Valentin I Spitkovsky, Stanford University and Google Inc., United States.
Bangalore Srinivas, ATT, India.
Narayanan Srinivasan, CBCS, India.
Sanja Stajner, University of Wolverhampton, United Kingdom.
Efstathios Stamatatos, University of the Aegean, Greece.
Manfred Stede, University of Potsdam, Germany.
Mark Steedman, University of Edinburgh, United Kingdom.
Ralf Steinberger, European Commission - Joint Research Centre (JRC), Italy.
Josef Steinberger, Joint Research Centre, Italy.
Mark Stevenson, University of Sheffield, United Kingdom.
Matthew Stone, Rutgers, United States.
Veselin Stoyanov, Johns Hopkins University, United States.
Carlo Strapparava, FBK-irst, Italy.
Michael Strube, HITS gGmbH, Germany.
Tomek Strzalkowski, SUNY Albany, United States.
Keh-Yih Su, Behavior Design Corp., Taiwan.
Jian Su, Institute for Infocomm Research, Singapore, Singapore.
Mari Carmen Suárez-Figueroa, Ontology Engineering Group (OEG), Universidad Politécnica de Madrid (UPM), Spain.
Le Sun, ISCAS, China.
Bapi Raju Surampudi, University of Hyderabad, India, India.

Jun Suzuki, NTT CS Lab., Japan.
Stan Szpakowicz, School of Electrical Engineering & Computer Science, University of Ottawa, Canada.
Marko Tadić, University of Zagreb, Faculty of Humanities and Social Sciences, Croatia.
Hirotoshi Taira, NTT Communication Science Laboratories, Japan.
John Tait, Johntait.net Ltd., United Kingdom.
Koichi Takeda, IBM Research, Japan.
Songbo Tan, Institute of Computing Technology, China.
Tien Ping Tan, Universiti Sains Malaysia, Malaysia.
Hristo Tanev, Joint Research Centre, United Kingdom.
Enya Kong Tang, Linton University College, Malaysia.
Kristian Tangsgaard Hvelplund, University of Copenhagen, Denmark.
Julia Taylor, Purdue University, United States.
Joel Tetreault, Educational Testing Service, France.
Michael Thelwall, University of Wolverhampton, United Kingdom.
Mariët Theune, University of Twente, Netherlands.
Joerg Tiedemann, Uppsala University, Sweden.
Christoph Tillmann, IBM Research, Germany.
Lana Timoshenko, Institute for Information Transmission Problems, Russian Academy of Sciences, Russian Federation.
Tomoki Toda, Nara Institute of Science and Technology, Japan.
Takenobu Tokunaga, TITECH, Japan.
Sara Tonelli, Fondazione Bruno Kessler, Italy.
kentaro Torisawa, National Institute of Information and Communications Technology (NICT), Japan.
Juan-manuel Torres, LIA Université d'Avignon, France.
Kristina Toutanova, Microsoft, United States.
Dolf Trieschnigg, University of Twente, Netherlands.
Harald Trost, Medical University of Vienna, Austria.
Andrew Trotman, University of Otago, New Zealand.
José A Troyano, University of Seville, Spain.
Richard Tzong-Han Tsai, Yuan Ze University, Taiwan.
Reut Tsarfaty, Uppsala University, Sweden.
Daniel Tse, University of Sydney, Australia.
Jun-ichi Tsujii, MSRA, Beijing, China.
Yoshimasa Tsuruoka, The University of Tokyo, Japan.
Zhaopeng Tu, Institute of Computing Technology, Chinese Academy of Sciences, China.
Francis Tyers, Universitat d'Alacant, Spain.
Hiroshi Uchida, UNDL, UNU, Japan.
Raghavendra Udupa, Microsoft Research, India.
Olga Uryupina, University of Trento, Italy.
Hans Uszkoreit, DFKI, Saarbrücken, Germany.
Masao Utiyama, NICT, Japan.
Takehito Utsuro, University of Tsukuba, Japan.
Ashwini Vaidya, University of Colorado, Boulder, United States.
Kees van Deemter, University of Aberdeen, United Kingdom.
Antal van den Bosch, LET, Netherlands.
Josef van Genabith, CNGL, NCLT, Dublin City University, Ireland.
Dániel Varga, Budapest University of Technology and Economics, Hungary.
Vasudeva Varma, IIIT Hyderabad, India.
Sriram Venkatapathy, Xerox Research Centre Europe, France.
Suzan Verberne, Radboud University Nijmegen, Netherlands.
Marc Verhagen, Brandeis University, United States.

Jose Luis Vicedo, University of Alicante (Spain), Spain.
Renata Vieira, PUCRS, Brazil.
David Vilar, DFKI, Germany.
Aline Villavicencio, Federal University of Rio Grande do Sul, Brazil.
Anne Vilnat, LIMSI-CNRS et Université Paris-Sud, France.
Jorge Vivaldi, Institut Universitari de Lingüística Aplicada (UPF), Spain.
Klaus von Heusinger, University of Cologne, Germany.
Piek Vossen, VU University Amsterdam, Netherlands.
Atro Voutilainen, University of Helsinki, Finland.
Xiaojun Wan, Peking University, China.
Haifeng Wang, Baidu, China.
Haifeng Wang, Baidu, China.
Rui Wang, DFKI GmbH, Germany.
Kewen Wang, Griffith University, Australia.
Bin Wang, Institute of Computing Technology, Chinese Academy of Sciences, China.
Leo Wanner, ICREA and Pompeu Fabra University, Spain.
Taro Watanabe, NICT, Japan.
Andy Way, Univ. of Dublin and Applied Language, Ireland.
Bonnie Webber, University of Edinburgh, United Kingdom.
Éric Wehrli, University of Geneva, Switzerland.
Furu Wei, Microsoft Research Asia, China.
Ben Wellner, Brandeis University, United States.
Michael White, The Ohio State University, United States.
Jan Wiebe, University of Pittsburgh, United States.
Yorick Wilks, University of Sheffield, United Kingdom.
Sandra Williams, The Open University, United Kingdom.
Philip James Williams, University of Edinburgh, United Kingdom.
Shuly Wintner, University of Haifa, Israel.
Andreas Witt, Institut für Deutsche Sprache, Germany.
Sue Ellen Wright, Affiliation unknown, United States.
Dekai Wu, HKUST, Hong Kong Special Administrative Region of China.
Fei Xia, University of Washington, United States.
Deyi Xiong, Institute for Infocomm Research, Singapore, Singapore.
Feiyu Xu, DFKI LT-Lab, Germany.
Gu Xu, Microsoft Bing, United States.
Nianwen Xue, Brandeis University, United States.
Serge Yablonsky, St. Petersburg University, Russia, Russian Federation.
Hongfei Yan, Peking University, China.
Muyun YANG, Harbin Institute of Technology, China.
Heng Yu, Institute of Computing Technology, Chinese Academy of Sciences, China.
Francois Yvon, LIMSI/CNRS & Univ Paris Sud, France.
Zdenek Zabokrtsky, Charles University, Prague, Czech Republic.
Annie Zaenen, Affiliation unknown, United States.
Fabio Massimo Zanzotto, University of Rome "Tor Vergata", Italy.
B Y Zarin, Universiti Malaysia Sarawak, Malaysia.
Torsten Zesch, Technische Universität Darmstadt, Germany.
Luke Zettlemoyer, University of Washington, United States.
Yi Zhang, German Research Center for Artificial Intelligence, Germany.
Min Zhang, Institute for Infocomm Research, Singapore.
Yue Zhang, Singapore University of Technology and Design, Singapore.
Hui Zhang, USC/ISI, United States.
Tiejun Zhao, Harbin Institute of Technology, China.

Yinggong Zhao, State Key Laboratory for Novel Software Technology at Nanjing University, China.
Ventsislav Zhechev, Autodesk, Switzerland.
Jing Zheng, SRI International, United States.
Ming Zhou, Microsoft Research Asia, China.
Guodong Zhou, Soochow University, China.
Xiaodan Zhu, National Research Council Canada, Canada.
ZhuJB Zhu, Northeastern University, China.
Sarka Zikanova, Charles University, Prague, Czech Republic.
Chengqing Zong, Institute of Automation, CAS, China.
Pierre Zweigenbaum, LIMSI-CNRS, France.

Table of Contents

Author Index

COLING 2012

24th International Conference on Computational Linguistics

Invited Speaker Abstracts

Minimum Description Length as the basis of Panini's grammar

Prof. Paul Kiparsky
Robert M. and Anne T. Bass Professor in the School of Humanities and Sciences
Department of Linguistics, Stanford University

Abstract
Panini attempted, and to a considerable extent succeeded, in constructing the shortest possible complete grammar of Sanskrit that contains a description of its own metalanguage. Minimizing the total length of the grammar required introducing a rule or convention just in case it achieves overall economies in the grammar which outweigh the cost of stating it.

The grammar presupposes nothing beyond certain elementary relations (such as "before" and "after") and operations (such as "replace"). Based on them, it defines a rich descriptive formalism. Simplicity dictates the inclusion of rules of grammar that encode all generalizations about Sanskrit phonology, morphology, and syntax, as well as of rules that define its grammatical categories, and of metarules that stipulate how rules of grammar apply and interact with each other. The grammar uses a fixed rule format, phonological and morphosyntactic features, rule ordering, cyclicity, blocking, the equivalent of Theta roles, inheritance hierarchies, and several hundred technical terms denoting classes of lexemes and morphemes. Four levels of representation (approximately corresponding to semantics, syntax, morphology, and phonology) emerge from the analysis.

Completeness of empirical coverage requires, among other things, the exhaustive treatment of derivational morphology, clausal syntax, variation (three degrees of optionality), and even of certain dialectal and sociolinguistic facts. Some of the abovementioned devices could be dispensed with in a less exhaustive description, as I will illustrate with the karaka system.

The grammar appears to be very nearly optimal. Although this has not be proved (except for certain subsystems), it appears likely, for no-one has been able to shorten the grammar in non-trivial ways (without losing content), either by modifying the rules without changing the metalanguage, or by modifying the metalanguage with additional devices and conventions, or removing some of the existing ones.

It would be anachronistic to construe the formal apparatus used by Panini as embodying a "theory": from his perspective it merely serves to compress the grammar. But the fact that many of the same conventions and principles that modern generative grammar posits as universals of language emerge just from the attempt to construct the maximally compact description of a single language is quite remarkable. It could

be taken as a challenge to the widespread assumption that learners are innately equipped with a format for grammatical description. For if a generative grammar can be arrived at purely by minimizing description length, without relying on any further prior assumptions, might not language acquisition by humans proceed in the same way? I will argue that this is not the case, because there is no effective procedure for constructing the maximally compact representation without prior analytic bias. In fact, the near-perfection of Panini's grammar and its metalanguage required hand-crafting by many generations of grammarians. In contrast, the rapidity of normal language acquisition, and the existence of robust cross-linguistic generalizations, remains a persuasive argument for UG.

The adaptive brain: acquiring a complex cognitive skill in complex contexts

Prof. Barbara Moser-Mercer
Director, Department of Interpretation
Faculty of Translation and Interpretation, University of Geneva

Abstract
Real-time human communication across language barriers relies on consecutive and simultaneous interpretation, a complex cognitive skill that can be acquired only over a certain period of time. Interpreting novices differ from interpreting experts in terms of their knowledge and knowledge organization, their analytical strategies, their use of memory processes, and the smoothness and speed with which they execute the interpreting task. In order to be able to move from comprehending a speech in one language and simultaneously interpreting that speech into another, the learner needs to make considerable adaptations to component processes of tasks already mastered, for the most part, before even being admitted to an interpreter training program. These adaptations concern mostly language comprehension and knowledge organization, component skills non-interpreters need in order to communicate. One must thus assume that significant changes occur in brain activity (functional changes or plasticity) and brain structure (structural plasticity) during the acquisition of interpreting skills that are the result of learning, knowledge re-organization, strategy acquisition, and task monitoring.

In investigating this hypothesis we have recently found evidence for brain structural plasticity in individuals training to become simultaneous interpreters as they develop expertise in this skill. We found that in interpreting students, but not in matched multilingual controls, there is an increase in gray matter volume over the course of a 15-month training program in brain regions known to be involved not only in semantic processing but also in aspects of executive function and error monitoring. Tasks involving the conversion of content from one language to another (i.e., translation and interpretation) mainly engage a left-lateralized cortico-subcortical circuit, including the basal ganglia, inferior frontal gyrus, and DLPFC. There is strong anatomical support for functional links between these regions. We propose that the evidence suggests the presence of two distinct networks contributing to the executive control of language. Although perturbing either may have superficially similar behavioral consequences, they are likely to have differing roles.

The plasticity of the brain allows for reshaping and reorganization, acquiring expertise in a task involves the generation of new neuronal connections whose survival is dependent on stimulation through extended electrical pulses that reverberate in the neural net to establish associations and connections between areas of specialized

information. The interpreter is constantly establishing logical connections on-line between what has already been comprehended and what is being heard, relying on multiple associations that have been formed while preparing for a specific assignment. The interpreter's ability to link new to existing information is thus one of the prime skills to be developed during skill acquisition: being able to associate multiple facets of data in neural networks with only a single sensory trigger firing up the entire network of associated facts emerges as one of the most important factors for successful skill acquisition. Fluency and speeded expert performance develop in a learning environment that is highly contextualized and provides multiple exposures to information so that the task can be executed efficiently as the multilingual brain adapts to complex demands.

Our understanding of skill acquisition in interpreting then informs our pedagogical approach and allows us to design learning environments for even the most extreme contexts, with socio-cultural environments characterized by political instability and conflict, where skills need to be acquired swiftly and reliably.

Digital Book, Digital Library, and Natural Language Processing

Prof. Makoto Nagao
Professor Emeritus, Kyoto University
Former President, National Diet (Congress) Library, Japan

Abstract
The following topics will be discussed with the author's experience with natural language processing and its applications to digital library.

1. Features of forth-coming digital books compared to the present-day paper books
2. Features of digital library which organizes digital books and offers highly sophisticated utilization of knowledge accumulated in digital library. This includes problems in digitization, structuring of a book according to the table of contents, varieties of retrieval methods which extract sections of a book, linking related parts of books as a hypertext structure, etc.
3. Construction of an ideal digital library based on these features
4. Natural language processing technologies which are required for the construction of future digital libraries.

NLP from Paninian Perspective

Prof. Dipti Misra Sharma
Prof. Dipti Misra Sharma

Abstract
Akshar Bharati, for several years now, has been pursuing NLP basing its linguistic models on Paninian grammatical framework. The talk will re-look at how the concepts from Panini's Grammar help in selecting and modelling linguistically informed NLP (both building resources and systems). (Akshar Bharati et al, 1995) proposed Computational Paninian Grammar models for various levels of linguistic analysis. It is noticed that it works well for relatively free word order languages. Not only, the insights from Panini's grammar help in exploiting morphological properties in computationally efficient parsing but also help in the selection of appropriate features for better machine learning.

Panini's grammar focusses on how language is used for comunication. Thus, language is viewed as a system which encodes information. There are three major schools of thought in the Indian grammatical tradition, the grammarians, the logicians and the text analysts. All of these schools lay emphasis on interpretation of meaning (Shaabdabodha)from what is given in a sentence. The grammar provides ways of identification of various linguistic units, their generation processes, relations across units and the syntactic realization of these relations. The talk will look at how a principled application of the concepts and the methods given in this tradition help in developing efficient computational models.

Most parsing approaches in NLP adopt either a constituency based grammar model or a dependency based one. Conversion from one to the other, combining constituency and dependency representation and producing a hyprid tree are some of the areas that the scholars in NLP have been looking at in the direction of bringing the two approaches together. However, Paninian approach suggests that languages encode information both ways. Thus, the talk will also explore whether both consituency and dependency can be incorporated in a single model and whether this would lead to better parsing.

COLING 2012

24th International Conference on Computational Linguistics

Technical Papers

Multi-dimensional feature merger for Question Answering

Apoorv Agarwal[1] *J. William Murdock*[2]
Jennifer Chu-Carroll[2] *Adam Lally*[2] *Aditya Kalyanpur*[2]
(1) Department of Computer Science, Columbia University, New York, NY, U.S.A.
(2) IBM T.J. Watson Research Center, Yorktown Heights, NY, U.S.A.
apoorv@cs.columbia.edu, {murdockj, jencc, alally,
adityakal}@us.ibm.com

Abstract

In this paper, we introduce new features for question-answering systems. These features are inspired by the fact that justification of the correct answer (out of many candidate answers) may be present in multiple passages. Our features attempt to combine evidence from multiple passages retrieved for a candidate answer. We present results on two data-sets: Jeopardy! and Doctor's Dilemma. In both data-sets, our features are ranked highest in correlation with gold class (in the training data) and significantly improve the performance of our existing QA system, Watson.

Keywords: Question Answering, multi-dimensional feature merger, Watson.

Proceedings of COLING 2012: Technical Papers, pages 1–16,
COLING 2012, Mumbai, December 2012.

1 Introduction

Most existing factoid question answering systems adopt search strategies and scoring algorithms with the assumption that a short passage exists in the reference corpus which contains sufficient information to answer each question. This assumption largely holds true for short and focused factoid questions such as those found in the TREC QA track (Voorhees and Tice, 2000). Examples of TREC QA questions include "*When did Hawaii become a state?*" and "*What strait separates North America from Asia?*" However, some more complex factoid questions contain facts encompassing multiple facets of the answer, which often cannot be found together in a short text passage. Consider the following examples, selected from collections of Jeopardy![1] and Doctor's Dilemma[2] questions, respectively:

> (1) WHO'S WHO IN SPORTS: Born in 1956, this Swedish tennis player won 6 French Opens & 5 straight Wimbledons (A: Björn Borg)
>
> (2) CARDIOLOGY: Murmur associated with this condition is harsh, systolic, diamond-shaped, and increases in intensity with Valsalva (A: Hypertrophic cardiomyopathy)

In both examples, information presented in the question can reasonably be expected to be in documents that describe the respective answer entities. However, it is quite unlikely that all the information will be present in one or two adjacent sentences in the document. More specifically, in example (1), we find birth year and nationality information in the basic biographic section of documents about Björn Borg, while statistics about his tennis record can generally be found in a section about Borg's career. Similarly, for example (2), the descriptions of typical murmurs associated with hypertrophic cardiomyopathy (harsh, systolic, and diamond-shaped) may not fall under the same section as the impact of Valsalva maneuver on the murmur (which is a factor used to distinguish hypertrophic cardiomyopathy from aortic stenosis). As a result, a typical passage retrieved from most reference corpus would cover only a portion of the facts given in the question.

These multi-faceted factoid questions present a challenge for existing question answering systems which make the aforementioned assumption. Consider the following short passages relevant to the question in example (2):

> (2.1 a) Hypertrophic cardiomyopathy generates a harsh late-systolic murmur, ending at S2.
>
> (2.1 b) The straining phase of the Valsalva maneuver induces an increase in the intensity of the systolic ejection murmur of hypertrophic cardiomyopathy.
>
> (2.2 a) A harsh, late-peaking, basal murmur radiating to the carotid arteries suggests aortic stenosis.
>
> (2.2 b) A classic physical finding of aortic stenosis is a harsh, crescendo-decrescendo systolic murmur that is loudest over the second right intercostal space and radiates to the carotid arteries.

Existing systems which evaluate each passage separately against the question would view each passage as having a similar degree of support for either hypertrophic cardiomyopathy or aortic

[1] http://www.jeopardy.com; Jeopardy! is a registered trademark of Jeopardy! Productions, Inc.
[2] http://www.acponline.org/residents_fellows/competitions/doctors_dilemma

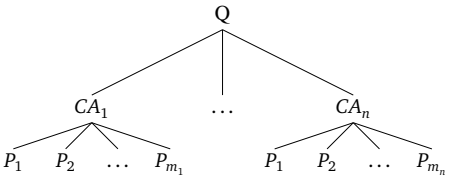


Figure 1: Typical question answering scenario. *Q* refers to question. *CA* are candidate answers for question *Q*, and *p* refers to passages supporting candidate answers.

stenosis as the answer to the question. However, these systems lose sight of a crucial fact, namely, that even though each passage covers half of the facts in the question, (2.1 a) and (2.1 b) cover disjoint subsets of the facts, while (2.2 a) and (2.2 b) address the same set of facts.

In this paper, we introduce the notion of *multi-dimensional feature merger* or *MDM* features, which allow for passage scoring results to be combined across different dimensions, such as question segments and different passage scoring algorithms. In this motivating example, MDM features that combine results across question segments would capture the broader coverage of passages (2.1 a) and (2.2 b), and thus enable the system to recognize hypertophic cardiomyopathy as a better answer for the question than aortic stenosis. We describe a general-purpose MDM feature merging framework that can be adopted in question answering systems that evaluate candidate answers by matching candidate-bearing passages against the question. We discuss our implementation of this MDM feature merging framework on top of our own question answering system, Watson. Finally, we demonstrate how passage scoring results can be merged across various dimensions in our system, resulting in 1) new features that are more highly correlated with correct answers than the base features from which they were derived, and 2) significant component level performance improvement and 3) end-to-end performance improvement. We present a comprehensive set of experiments for our current domain of interest – the medical domain and a less comprehensive set of experiments for Jeopardy! data.

The rest of the paper is organized as follows. In section 2, we describe our feature set. Since we build on existing state-of-the-art QA system, in section 3, we briefly describe the current system, focusing on the component of the system that we enhance in this paper. In section 4, we describe passage scorers in the current system, with specific examples of features that leverage scores assigned to passages by these scorers. In section 5, we presents a detailed description of the data we use for training and testing. Additionally, we present experiments and results to show the impact of our features. Section 6 presents a survey of current work in question answering. Finally, we conclude and present future direction of research in the last section.

2 Multi-dimensional feature merger (MDM)

Given a question, *Q*, each of its candidate answer, *CA*, has a set of supporting passages (Figure 1). In a typical question-answering system, support of each passage for a candidate answer is quantified. Then a merging strategy is used to combine the support of all passages for a particular candidate answer. In this paper, we introduce a general framework for merging support from supporting passages.

The methodology of calculating the support of a passage for a candidate answer is called *passage scoring* (Murdock et al., 2012a). At an abstract level, a passage scorer is responsible for quantifying how well a passage *matches* a question. We represent a question and a passage as an ordered set of

sum($\vec{s}$)	avg($\vec{s}$)	std($\vec{s}$)	max($\vec{s}$)	min($\vec{s}$)	non-zero($\vec{s}$)
$\sum_{j=1}^{cols} s_j$	$\frac{sum(\vec{s})}{cols}$	$\sqrt{\frac{\sum_{j=1}^{cols}(s_j - avg(\vec{s}))^2}{cols-1}}$	$\arg\max_{j\in[1,cols]} s_j$	$\arg\min_{j\in[1,cols]} s_j$	$\lvert\{s_j \lvert s_j \neq 0 \forall j \in [1, cols]\}\rvert$

Table 1: Standard formulae that constitute $g(M)$

Question	large	land	animal	has	large	ears	
P1.1	x_1	x_2	x_3	x_4	x_5	x_6	$\vec{f_{1,1}}$
P1.2	x_7	x_8	x_9	x_{10}	x_{11}	x_{12}	$\vec{f_{1.2}}$
P2.1	x_{13}	x_{14}	x_{15}	x_{16}	x_{17}	x_{18}	$\vec{f_{2,1}}$
P2.2	x_{19}	x_{20}	x_{21}	x_{22}	x_{23}	x_{24}	$\vec{f_{2.2}}$

Table 2: Passage match scores for question and passages in Figure 2.

terms ($Q = \{q_1, q_2, \ldots, q_n\}$), and ($P = \{p_1, p_2, \ldots, p_m\}$) respectively, Passage scorers align question terms to passage terms and assign a *score* based on how well the terms align. For example, a passage scorer will take as input Q and P and output a vector of scores that represents how well the passage matches the question. We denote this vector for P as $\vec{f}$ such that f_i is the score of how well one of the passage terms matches the i^{th} term in the question. Note the length of this vector is fixed per question but may vary across questions.

We collect all these vectors per question, per candidate answer into a matrix, M. For example, CA_1 may be represented as a matrix where row i corresponds to the passage scoring vector for passage P_i. An element of this matrix, $f_{i,j}$ is the score assigned by one of the passage scorers of how well passage P_i aligns with the term j in the question Q.

This matrix is of variable dimensions for different candidate answers per question. Number of rows could be different because the number of supporting passages could be different for each candidate answer for the same question. Since different questions have different number of question terms, the number of columns could be different for candidate answers across questions. Therefore, we cannot capture the distribution of this matrix simply by linearizing the matrix.

In this paper, we define a function $f : M \rightarrow \mathbf{R}^N$, that maps each matrix into feature vector of fixed length, N. This function is defined as follows:

$$f(M) =< g(M), g(M') >$$

where M' is the transpose of matrix M and g is a function $g : M \rightarrow \mathbf{R}^{N/2}$ that maps a matrix into feature vector of fixed length, defined as follows:

$$g(M) =<\text{sum}(\vec{s}), \text{avg}(\vec{s}), \text{std}(\vec{s}), \max(\vec{s}), \min(\vec{s}), \dim(\vec{s}), \text{non-zero}(\vec{s})>$$

where $\vec{s}$ is a vector of dimensionality $\dim(\vec{s})$, such that $s_j = \sum_{i=1}^{rows} f_{i,j}$ and the remaining standard formulae are given in Table 1.

Consider an example Jeopardy! question:[3] *This large land animal also has large ears.* Consider two candidate answers and their supporting passages:

[3] modified for readability.

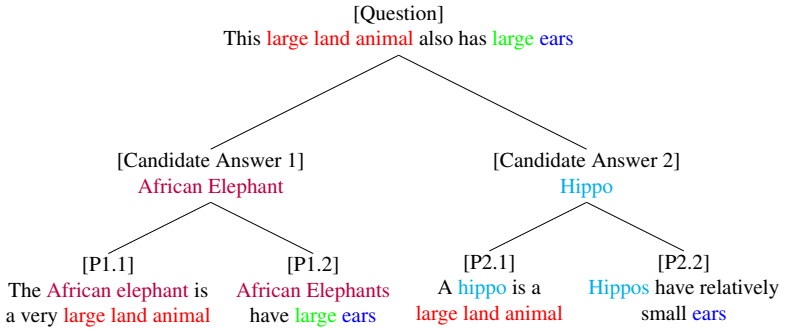


Figure 2: A specific example showing candidate answers and supporting passages for a modified Jeopardy! question. P1.1 means first justifying passage for the first candidate answer.

1. Candidate answer 1: African Elephant
   (a) P1.1: The African Elephant is a very large land animal.
   (b) P1.2: African elephants have large ears.
2. Candidate answer 2: Hippo
   (a) P2.1: A hippo is a large land animal.
   (b) P2.2: Hippos have relatively small ears.

This example is shown pictorially in Figure 2

Table 2 abstractly shows how passage scorers assign values to specific question terms for specific passages. For example, consider the P1.1 row, which represents how well the passage *The African elephant is a very large land animal* supports the answer *elephant* for the question *This large land animal also has large ears*. If the passage scorer is effective, it will give a high score to x_1, x_2 and x_3 (because the passage does, indeed, provide strong justification for "elephant" satisfying the requirements of being large land animal). It will give a very small score (typically 0) to x_4, x_5, and x_6, because the passage says nothing about elephants having large ears. However, some passage scorers may be mislead by the fact that the term "large" appears twice question and either one could align to the one occurrence in the passage. Often some passage scorers match too many terms and thus assign credit to terms that don't deserve it while others match too few and miss important content; this is why we have a diverse collection of scorers and let the classifier sort out how much to trust each of them.

Using one of the existing merging strategy, say MAX, candidate answer 1, African Elephant, will get assigned a feature value equal to $MAX\{(x_1+x_2+x_3+x_4+x_5+x_6),(x_7+x_8+x_9+x_{10}+x_{11}+x_{12})\}$. So either passage P1.1 or passage P1.2 will be selected as an optimal passage. As is apparent from this merger strategy, it does not attempt to leverage the complementary information in the two passages. Our merging strategy will attempt to capture the distribution of alignment across passages. For the matrix for African Elephant, M, $f(M) =< g(M), g(M') >$. First dimension of vectors

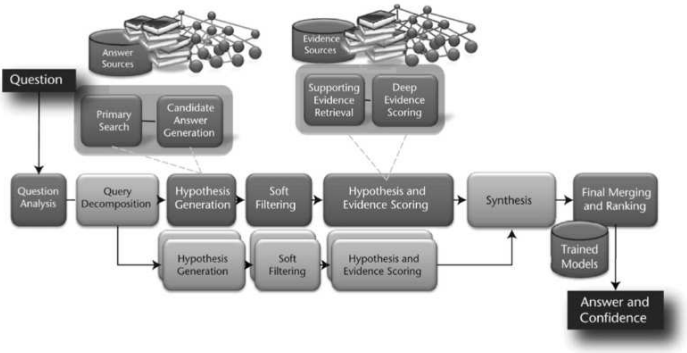


Figure 3: Architecture of Watson, state-of-the-art DeepQA system (taken from (Ferrucci et al., 2010)).

$g(M)$ and $g(M')$ will be the same, because $sum(\vec{s}) = sum(\vec{s'}) = \sum_{i=1}^{12} x_i$. But others will be different. For example, $mean(\vec{s}) = \frac{1}{6} * sum(\vec{s})$, whereas, $mean(\vec{s'}) = \frac{1}{2} * sum(\vec{s})$.

Note, the $sum(\vec{s})$ feature is aggregating the information across passages. In a passage scorer, which assigns 1 for a match and 0 otherwise, it is clear why this feature will have a higher value for African Elephant, the correct answer, than Hippo (because Hippo's don't have *large* ears).

Our framework is general in three ways: 1) It is independent on the type of passage scorer, 2) More matrix operations (like *rank(M)*), may be easily added to the definition of function $g(M)$, and 3) Our framework is easily extensible to beyond two dimensions, which can be used to capture additional orthogonal feature dimensions (see future work section for an example).

In the following sections, we first describe a specific, and state-of-the-art QA system, Watson. We present where our features fit in the larger architecture. Then we give an overview of specific passage scorers and merging strategies in the current system, followed by experiments and results showing that the new features we introduce add value to the current system.

3 Overview of Watson

IBM undertook the challenge to build a question-answering system named *Watson* that is able to answer open domain questions, such as those posed in a U.S. quiz show Jeopardy!. An overview of the architecture of Watson is illustrated in Figure 3. We refer the reader to (Ferrucci et al., 2010) for a detailed description of the architecture. In this section, we present a high level overview of the system pointing out where our features fit in.

The DeepQA system analyzes a question, *Question Analysis* (Lally et al., 2012), and generates multiple possible candidate answers, *Hypothesis Generation* (Chu-Carroll et al., 2012).It then applies many different answer scoring algorithms, each of which produces features that are used to evaluate whether the answer is correct. One way in which DeepQA evaluates candidate answers is to first retrieve passages of text that contain the candidate answer, via a technique called Supporting Evidence Retrieval; each passage is then scored using a variety of algorithms called passage scorers

$$< Q_1, CA_1, -1 >, < Q_1, CA_2, -1 >, \ldots, < Q_1, CA_i, 1 >, \ldots, < Q_1, CA_{n_1}, -1 >$$
$$< Q_2, CA_1, -1 >, < Q_2, CA_2, -1 >, \ldots, < Q_2, CA_j, 1 >, \ldots, < Q_2, CA_{n_2}, -1 >$$
$$\ldots$$
$$< Q_m, CA_1, -1 >, < Q_m, CA_2, -1 >, \ldots, < Q_m, CA_k, 1 >, \ldots, < Q_m, CA_{n_m}, -1 >$$

Figure 4: Training and test data for a question-answering system. Each question Q has multiple candidate answers, CA, where few, if any, are correct (class = 1).

in the *Hypothesis and Evidence Scoring* phase (Murdock et al., 2012a). All of the features are sent to a *Final Merging and Ranking* (Gondek et al., 2012) component, which uses machine learning techniques to weigh and combine features to produce a single confidence value estimating the probability that the candidate answer is correct. The features we introduce are extracted and made available to the machine learning model in the Final Merging and Ranking component, where the scores assigned by different passage scorers are available. In the next section 4, we give details of existing passage scorers and their feature merging strategies used prior to the framework introduced in this paper.

4 Passage scoring

Our question-answering system works by finding candidate answers, employing a variety of algorithms to compute feature values relating to those answers, and then using a statistical classifier to determine which candidate answer is correct. A question-answering scenario is shown in Figure 1. For a given question Q, search components find a set of candidate answers $\{CA_1, CA_2, \ldots, CA_n\}$. The task of the classifier is to decide which of the candidate answers is the correct answer. Hence the training and test data for that classifier looks as in Figure 4.

Each candidate answer is associated with one or more passages that contain the candidate answer. A subset of the algorithms that compute feature values in our system are the passage scoring components. These components evaluate the evidence that a single passage provides relating to how well the candidate answer satisfies the requirements of the question. Thus among the feature values associated with a candidate answer, some will be passage scoring features.

Our passage scorers are described in detail elsewhere (Murdock et al., 2012a). Here we provide only a brief introduction to provide context for later sections of this paper. We have a variety of passage scoring algorithms that use different strategies for determining which parts of a question to attempt to match to each part of a passage and for determining whether two parts of a passage match. Some attempt to align question terms to passage terms using syntactic structure and/or semantic relations, while others use word order or ignore the relationship among terms completely (e.g., simply counting how many question terms appear in the passage, regardless of whether those terms are similarly arranged).

Watson's passage scorers leverage available annotation components developed for the DeepQA framework, such as dependency parsing, Named Entity (NE) recognition, coreference resolution and relation detection. The question and the passage are decomposed into sets of terms, where a term can either be a single token or a multiword token. All of these scorers try to determine the amount of overlap between the passage and the question by looking at which terms match. The individual scorers put different restrictions on when a term is considered to *match*.

Currently, there are four scorers being used in the system:

1. **Passage Term Match**: Assigns a score based on which question terms are included in the passage, regardless of word order or grammatical relationship.

2. **Skip Bigram**: Assigns a score based on whether pairs of terms that are connected or nearly connected in the syntactic-semantic structure of the question match corresponding pairs of terms in the passage.

3. **Textual Alignment**: Assigns a score based on how well the word order of the passage aligns with that of the question, when the focus is replaced with the candidate answer.

4. **Logical Form Answer Candidate Scorer** (LFACS): Targets high-precision matching between the syntactic structures of passages and questions, and is therefore quite restrictive concerning structural overlap of the question and the passage. Like Skip Bigram, it operates on syntactic-semantic structural graphs, which contain one node for each lexical item.

Each passage scoring component produces a fixed number of feature value pairs for each candidate answer within each passage. Some of these values range from 0 to 1, where a high score indicates that the passage matches the question well based on that passage scorer's evaluation criteria; other passage scorers have other ranges. Watson's final answer merging and ranking component considers a pre-defined set of features and applies a machine learned model to score each candidate answer. However, since each candidate has multiple, and generally a varying number of supporting passages, we use a merger to combine passage scores for $<$ candidate answer, passage $>$ pairs into a fixed set of features. For example, if a candidate answer has three passages and a passage scorer assigns a value of 0.5, 0.6, and 0.7 to each passage, these scores may be merged using a merger strategy like MAX. Using this merger strategy, the feature added to the learning model for the candidate answer under consideration will be $MAX(0.5, 0.6, 0.7) = 0.7$.

We have the following three distinct algorithms that we use to merge features across passages (Gondek et al., 2012).

1. Maximum: The final score for the candidate answer is the maximum score for that answer in any passages found for that answer.

2. Sum: The final score for the candidate answer is the sum of the scores for that answer in each of the passages found for that answer.

3. Decaying sum: The final score for the candidate answer is computed to be $\sum_{i=0}^{m} \frac{p_i}{2^i}$, where $p_0, p_1, \ldots, p_m$ are the scores of the passages that contain the answers, sorted in descending order.

A key limitation of our earlier work is that the passage scorers capture limited complementary information that the passages have to offer. For example, in Figure 2, a passage scoring component may assign scores $s_{1.1}, s_{1.2}$ to passages P1.1 and P1.2 respectively. A merger strategy that takes maximum across passages will choose $MAX(s_{1.1}, s_{1.2})$ as the optimal supporting passage. However, since these passages have complementary information to offer, it would be better to somehow aggregate this information. This is exactly where our multi-dimensional merging features come into the picture.

As described in earlier publications (Gondek et al., 2012), for each of our features, we have two other derived features: a feature for whether that feature is missing and a standardized version of the

Feature name	Explanation	In terms of Table 2
MDM-TextualAlignment-sum-then-mean	For each question term, compute the sum of the Textual Alignment scores across all passages, and then compute the mean of the sums	$f(M) = [(x_1 + x_7) + (x_2 + x_8) + (x_3 + x_9) + (x_4 + x_{10}) + (x_5 + x_{11}) + (x_6 + x_12)]/6$
MDM-SkipBigram-transpose-sum-then-mean	For each passage, compute the sum of the Skip-Bigram scores across all question terms, and then compute the mean of the sums	$f(M) = [(x_1 + x_2 + \ldots + x_6) + (x_7 + x_8 + \ldots + x_{12})]/2$
MDM-LFACS-max-then-sum	For each question term, compute the maximum of the LFACS scores across all passages, and then compute the mean of the maxima	$f(M) = max(x_1, x_2, \ldots, x_6) + max(x_7, x_8, \ldots, x_{12})$
MDM-SkipBigramScore-transpose-sum-then-nonZeroColumns	For each passage, compute the sum of the Skip-Bigram scores across all question terms, and then compute the number of sums that are non-zero	Set $cnt = 0$. If $(x_1 + x_2 + \ldots + x_6) > 0, cnt = cnt + 1$. If $(x_7 + x_8 + \ldots + x_{12}) > 0, cnt = cnt + 1$. $F(M) = cnt$.

Table 3: Examples of MDM features. First column is the feature name, column 2 a natural language description of the feature and the third column is the exact mathematical formula in reference to Table 2 for passages P1.1 and P1.2 belonging to the candidate answer 1.

feature. When the value of a feature is missing, we assert a value of 0 for the feature and a value of 1 for the corresponding derived missing feature; this allows the learner to distinguish between cases where the feature actually has 0 value versus cases where it simply did not apply at all. The standardized version of a feature is computed by subtracting the mean value of that feature and dividing by the standard deviation for that feature. Both mean and standard deviation are computed across all answers to a single question, *not* across all answers to all questions in the test set. The purpose of the standardized feature is to encode how much the base feature differs from a typical value of that feature for a single question.

In Table 3, we present examples of some top scoring (in terms of correlation with the gold class) MDM features. For a passage scoring feature X, we produce the following MDM features: MDM-X-sum-then-mean ($avg(\vec{s})$), MDM-X-transpose-sum-then-mean ($avg(\vec{s'})$), MDM-X-sum-then-max ($max(\vec{s})$) etc.

5 Experiments and Results

To demonstrate the generality of our approach, we experimented with two data sets, an open-domain question set and one focused on the medical domain. We briefly describe these data sets in this section. Our first open-domain test set is a randomly selected set of 3,505 Jeopardy! questions. Jeopardy! questions span a large number of domains, including arts and entertainment, history, geography, and science. These questions are also generally more complex, incorporating multiple loosely related facts about the correct answers, particularly as compared with typical questions from the TREC QA track. The last characteristic makes Jeopardy! questions an excellent test set for our MDM feature merging framework.

Our second test set is a collection of 905 Doctor's Dilemma questions. Doctor's Dilemma, also

	#Questions	#Positive	#Negative	#Average cand. per Q
Jeopardy!	11,520	12,173	2,555,396	222.87
Doctor's Dilemma	1,322	2,338	543,963	413.23

Table 4: Data distribution for our data-sets. #Question refers to number of questions. #Positive refers to number of positive instances i.e. correct answers to questions, #Negative refers to number of negative instances and #Average cand. per Q refers to the average number of candidates considered for a particular question. Note, this is simply total number of positive and negative examples divided by the number of questions in the data-set.

known as Medical Jeopardy, is a competition organized by the American College of Physicians for medical interns and residents and held each year at the Internal Medicine meeting. The format of these questions is modeled after Jeopardy!, while their content is focused solely on topics related to medicine. Although not as linguistically complex as Jeopardy! questions, Doctor's Dilemma questions generally also consists of multiple facts about the correct answer, making it suitable as a test set for MDM features. Following are some examples from the Doctor's Dilemma domain:

1. The syndrome characterized by joint pain, abdominal pain, palpable purpura, and a nephritic sediment. Answer: Henoch-Schonlein Purpura.

2. Familial adenomatous polyposis is caused by mutations of this gene. Answer: APC Gene.

3. The syndrome characterized by narrowing of the extra-hepatic bile duct from mechanical compression by a gallstone impacted in the cystic duct. Answer: Mirizzi's Syndrome.

We use a supervised learning paradigm, with features extracted as described in previous sections. We use logistic regression classifier for training and testing. We report results on a held-out test set for both data-sets. The distribution of training set for the two data-sets are in Table 4. We test on 3,505 Jeopardy! questions and 905 DD questions.

We present three types of analyses to show the usefulness of our features. First, we present the correlation of our features with the gold class (for the training set only) i.e. correctness of a candidate answer. Second, we present a *component level analysis*, where we add our features to a baseline QA system and show improvement. Third, we present results on the end-to-end Watson system.

5.1 Correlation

A standard way to judge the goodness of features is to look at the features' Pearson's r correlation with the gold class (Hall, 2000). The Pearson's r correlation coefficient between feature X and gold standard Y is given by:

$$r = \frac{\sum_{i=1}^{n}(X_i - \bar{X})(Y_i - \bar{Y})}{\sqrt{\sum_{i=1}^{n}(X_i - \bar{X})^2}\sqrt{\sum_{i=1}^{n}(Y_i - \bar{Y})^2}}$$

where $\bar{X}$ and $\bar{Y}$ are the arithmetic mean of feature values and gold class values respectively. We refer to the degree of correlation between the feature and the gold class as the "informativeness" of the feature. Naturally, we would like to keep features that have high informativeness.

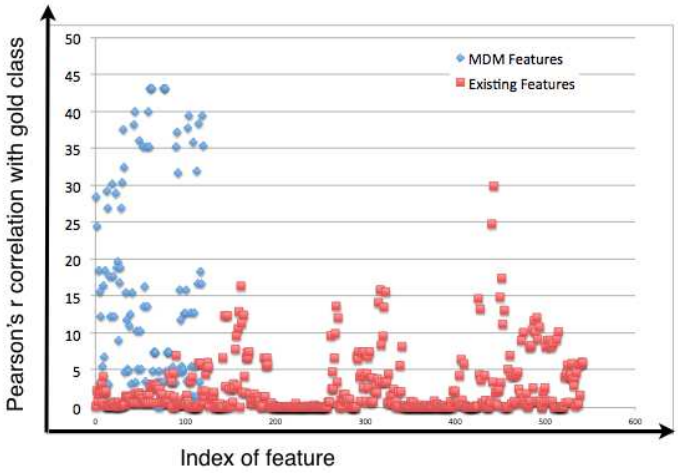


Figure 5: Inform analysis comparison of MDM features with the existing features in the system trained on **Jeopardy!** data. X-axis is the feature index (in no specific order) and Y-axis is the % correlation of features with the gold class.

Figure 5 presents the informativeness of existing features (red squared dots) and MDM features (blue diamond dots) for the Jeopardy! data-set. In figure 5, the x-axis is the feature index (existing features indexed from 1 to 535 and new features indexed from 1 to 110) and the y-axis is the informativeness of the features. For example, the highest informativeness of existing features (square red dot) is 30% ($100 \cdot r$), while the highest informativeness of MDM features is 43.2%. Many of the MDM features have higher informativeness than the most correlated feature in the existing system.

Similar is the case with the medical domain data. Figure 6 presents the informativeness of existing features (red squared dots) and MDM features (blue diamond dots) for the Doctor's Dilemma data-set. The highest informativeness of MDM features is 21.5%, which is comparable to the three existing features with highest informativeness (between 20% to 21%). However, as the graph shows, the vast majority of MDM features have substantially higher informativeness than the original features. the Jeopardy! domain, many of the MDM features are more correlated with answer correctness than most of the original features.

5.2 Component level analysis

As described in section 3, we add new features in the final merger stage of the system. Our features are calculated for each of the four passage scorers described in section 4. In this section, we evaluate the impact of these MDM features when only a single passage scoring component is employed in the system. To do so, we create a *component level baseline* for each of our four passage scorers as follows: on top of the Watson answer-scoring baseline configuration(Ferrucci et al., 2010), which includes all of the standard question analysis, search, and candidate generation, but only one answer scorer (which checks answer types using a named entity detector (Murdock et al., 2012b)) and a simplified configuration for merging and ranking answers. We add each of our existing Passage

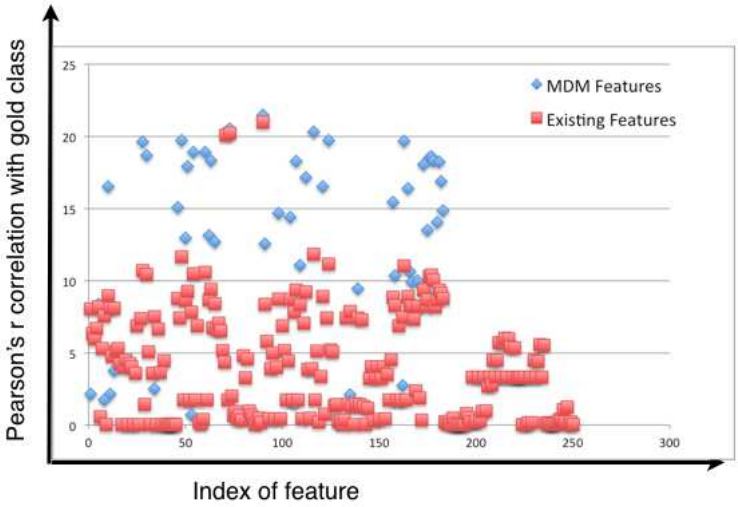


Figure 6: Inform analysis comparison of MDM features with the existing features in the system trained on **Doctor's Dilemma** data. X-axis is the feature index (in no specific order) and Y-axis is the % correlation of features with the gold class.

Term Match, Skip Bigram, Textual Alignment, and LFACS passage scoring, to create four baseline systems. We then compare each baseline to the system with our MDM features for the corresponding passage scorer and show a significant gain in Precision@70% and accuracy.

We often consider Precision@70% as a numerical measure that combines the ability to correctly answer questions and the ability to measure confidence; this metric corresponds to the precision when the system answers 70% of the questions of which it is most confident.

Table 5 present results for our component level analysis for Doctor's Dilemma questions. A component level baseline for each passage scorer was computed as described above. System performance improves across the board after adding MDM features for a passage scorer. Using

	Component Level Baseline		With MDM features	
Passage Scorer	Precision@70%	%Accuracy	Precision@70%	%Accuracy
Passage Term Match	24.9	20.2	29.2	23.4
Skip Bigram	26.8	21.5	28.7	23.3
Textual Alignment	22.9	18.8	25.7	21.1
LFACS	25.7	20.3	28.5	22.4

Table 5: Component level comparison for Doctor's Dilemma data-set for each of the four passage scorers. Each component level baseline is the answer-scoring baseline plus features for one of the passage scorers. All the numbers after adding MDM features for a passage scorer are significantly better than the baseline by $p < 0.05$, using McNemar's significance testing.

	Baseline		With MDM features	
Data-set	Precision@70%	%Accuracy	Precision@70%	%Accuracy
Doctor's Dilemma	37.2	29.2	40.2	31.3

Table 6: End-to-End comparison for medical domain data, Doctor's Dilemma. Baseline refers to the configuration with all the current features in the system. With MDM features refers to the configuration when we add all our MDM features to the existing feature set. This difference in performance is statistically significant with $p < 0.05$, using McNemar's significance testing.

McNemar's significance test, these are statistically significant improvements over the baseline at $p < 0.05$. As is clear from the results, for each of the four passage scorers, adding MDM features that capture the distribution of the passage scores across multiple passages improves the performance, in terms of both Precision@70% and % accuracy, by a significant amount.

For the Jeopardy! data-set, for the LFACS passage scorer, Precision@70% improves from 64.9% to 71.3% and % Accuracy improves from 52.2% to 57.3%. Both these improvements are statistically significant at $p < 0.05$, using McNemar's significance testing.

Based on these experimental results, we conclude that addition of MDM features for passage scorers significantly improves the performance of our QA system.

5.3 End-to-End Analysis

In this section, we present results for running the full Watson system with and without MDM features. Table 6 shows the Precision@70% and % accuracy performance on the Doctor's Dilemma test set. The results show that by adding MDM features to existing system, we are able to get a statistically significantly better performance than the baseline system: Precision@70% improves from 37.2 to 40.2 and % accuracy improves from 29.2% to 31.3%.

6 Literature Survey

Question answering has had a long history (Simmons, 1970) and has seen considerable advancement over the past decade (Maybury, 2004; Strzalkowski and Harabagiu, 2006). However, to the best of our knowledge, there is no general purpose framework integrated into a QA system that is capable of aggregating information across multiple pieces of evidence, each analyzed using different analytics (features), and comparing this with coverage of terms/facts in the input question.

A technique that is complementary to ours is corpus expansion (Schlaefer et al., 2011), in which corpus documents are expanded to include topically related facts from an external resource (e.g. Web). Sometimes in this process, pseudo documents are created which contain aggregate information about a particular entity. This approach helps standard document search by providing better document-level evidence/scores for the input search terms. The system is more likely to find a single document that addresses all of the parts of the question in a corpus after it has been expanded. However, passage scoring still encounters the same underlying problem even with an expanded corpus: in some cases, there will not be any single passage that addresses all of the requirements of the question.

The second related approach is question decomposition (Kalyanpur et al., 2012; Felshin, 2005), which aims at decomposing the question into different facts that need to be independently or sequentially solved in order to arrive at the correct answer. However, question decomposition does not deal with the issue of combining multiple pieces of evidence (possibly assessed using different

analytics) for the same fact within a decomposed question (which our approach does). In addition, the process of decomposing a question into multiple subquestions is an extremely challenging linguistic one, and is very sensitive to how questions are phrased; a set of rules that are effective at formulating subquestions from Jeopardy! clues may not be as effective for other types of questions. Multi-dimensional merging also requires that the question be divided up, but it does not require that the parts of the question form coherent subquestions, since it is performed *after* all of the linguistic analysis and comparison to evidence. In our implementation of multi-dimensional merging, we simply divide up the question into single terms.

We consider both corpus expansion and question decomposition as complementary to our approach. Both approaches are included in our baseline Jeopardy! system, and corpus expansion is included in our baseline medical system. The fact that our results show postive impact on effective question answering shows that multi-dimensional merging can add value to a system that already uses both corpus expansion and question decomposition techniques.

Conclusion and perspectives

We introduced a general framework for aggregating evidence from different passages retrieved for a candidate answer. Moreover, we introduced a novel set of features, multi-dimensional feature merger or MDM features, that fit this framework and significantly improve the performance of the current state-of-the-art QA system, Watson. However, our framework is general and not restricted to Watson. It may be employed in any QA system that captures how well retrieved passages match the question under consideration.

In this paper, we only considered merging evidence across passages and question terms. However, this may be easily extended to merging evidence across passage scorers. There might be value in considering how different passage scorers match supporting passages with candidate answers. Using our framework, all that is required is adding a new dimension: depth to the two-dimensional matrix M, thus giving rise to a $3-D$ matrix, say $M3D$. Each two dimensional matrix, M in $M3D$ belongs one passage scorer. Therefore, depth of $M3D$ is the number of passage scorers used to match supporting passages with the question. In the future, we will explore decomposing and thus deriving features from this $3-D$ matrix, possibly using Tensor algebra (Kolda and Bader, 2008).

References

Chu-Carroll, J., Fan, J., Boguraev, B. K., Carmel, D., Sheinwald, D., and Welty, C. A. (2012). Finding needles in the haystack: Search and candidate generation. *IBM Journal Research and Developement*, 56.

Felshin, B. K. . G. B. . S. (2005). Syntactic and semantic decomposition strategies for question answering from multiple resources. *AAAI*.

Ferrucci, D. A., Brown, E. W., Chu-Carroll, J., Fan, J., Gondek, D., Kalyanpur, A., Lally, A., Murdock, J. W., Nyberg, E., Prager, J. M., Schlaefer, N., and Welty, C. A. (2010). Building watson: An overview of the deepqa project. *AI Magazine*, 31:59–79.

Gondek, D. C., Lally, A., Kalyanpur, A., Murdock, J. W., Duboue, P. A., Zhang, L., Pan, Y., Qiu, Z. M., and Welty, C. A. (2012). A framework for merging and ranking of answers in deepqa. *IBM Journal Research and Developement*, 56.

Hall, M. A. (2000). Correlation-based feature selection for discrete and numeric class machine learning. *17th International Conference of Machine Learning (ICML)*, pages 359–366.

Kalyanpur, A., Patwardhan, S., Boguraev, B. K., Lally, A., and Chu-Carroll, J. (2012). Fact-based question decomposition in deepqa. *IBM Journal Research and Developement*, 56:13:1–13:11.

Lally, A., Prager, J. M., McCord, M. C., Boguraev, B. K., Patwardhan, S., Fan, J., Fodor, P., and Chu-Carroll, J. (2012). Question analysis: How watson reads a clue. *IBM Journal of Research and Development*, 56.

Maybury, M. T. (2004). *New Directions in Question-Answering*. Melno Park CA: American Association for Artificial Intelligence.

Murdock, J. W., Fan, J., Lally, A., Shima, H., and Boguraev, B. K. (2012a). Textual evidence gathering and analysis. *IBM Journal Research and Developement*, 56.

Murdock, J. W., Kalyanpur, A., Welty, C. A., Fan, J., Ferrucci, D. A., Gondek, D. C., Zhang, L., and Kanayama, H. (2012b). Typing candidate answers using type coercion. *IBM Journal Research and Developement*, 56.

Schlaefer, N., Chu-Carroll, J., Nyberg, E., Fan, J., Zadrozny, W., and Ferrucci, D. (2011). Statistical source expansion for question answering. In *Proceedings of the 20th ACM international conference on Information and knowledge management*, CIKM '11, pages 345–354, New York, NY, USA. ACM.

Simmons, R. F. (1970). Natural language question-answering systems: 1969. *Commun. ACM*, 13:15–30.

Strzalkowski, T. and Harabagiu, S. (2006). *Advances in Open-Domain Question-Answering*. Berlin Germany: Springer-Verlag.

Voorhees, E. and Tice, D. (2000). Building a question answering test collection. *SIGIR*, pages 200–207.

Unsupervised Discovery of Relations and Discriminative Extraction Patterns

Alan Akbik[1] *Larysa Visengeriyeva*[1]
Priska Herger[2] *Holmer Hemsen*[1] *Alexander Löser*[1]
Technische Univeristät Berlin
Database Systems and Information Management Group
Einsteinufer 17, 10587 Berlin, Germany
{firstname.lastname}@tu-berlin.de[1], priska@campus.tu-berlin.de[2]

ABSTRACT
Unsupervised Relation Extraction (URE) is the task of extracting relations of *a priori* unknown semantic types using clustering methods on a vector space model of entity pairs and patterns. In this paper, we show that an informed feature generation technique based on dependency trees significantly improves clustering quality, as measured by the F-score, and therefore the ability of the URE method to discover relations in text. Furthermore, we extend URE to produce a set of weighted patterns for each identified relation that can be used by an information extraction system to find further instances of this relation. Each pattern is assigned to one or multiple relations with different confidence strengths, indicating how reliably a pattern evokes a relation, using the theory of Discriminative Category Matching. We evaluate our findings in two tasks against strong baselines and show significant improvements both in relation discovery and information extraction.

KEYWORDS: Unsupervised Relation Extraction, Clustering, Vector Space Models.

Proceedings of COLING 2012: Technical Papers, pages 17–32,
COLING 2012, Mumbai, December 2012.

1 Introduction

Recently, there has been great interest in broadening information extraction methods to allow for unsupervised discovery of relational information in large document collections of unknown content. Contrary to classic information extraction in which relationship types (such as BORNIN or MARRIEDTO) are specified in advance, such methods automatically identify *a priori* unknown relationship types in a given corpus. For these identified semantic relations[1], they subsequently or simultaneously perform an information extraction step, thereby transforming the corpus into structured, relational data without any supervision or previous knowledge about its content.

One such approach, Unsupervised Relation Extraction (URE), addresses this challenge by building on the *latent relation hypothesis* which states that pairs of words that co-occur in similar patterns tend to have similar relations (Turney, 2008; Rosenfeld and Feldman, 2007). Current techniques capture this in a vector space model by computing a *pair-pattern matrix* in which each row represents an entity pair and each column a distinct pattern, with co-occurrence counts as cell values. This representation allows us to compute the similarity of two entity pairs by comparing the distribution over observed patterns. Using such a similarity metric, clustering methods can find clusters of entity pairs that share similar patterns and can therefore be assumed to represent a relation. Ideally, a clustering method returns three kinds of structured information, each of which is highly relevant to information discovery and extraction in unknown corpora: Firstly, a set of clusters, each of which represents one distinct *relation*. Secondly, for each cluster a set of *entity pairs* between which this relation holds. Thirdly, for each cluster a set of discriminative *patterns* that extensionally describe the relation and may be used by an information extraction system to find further *relation instances* of this relation.

The choice of patterns as well as their significance within a cluster assignment is crucial to the success of this endeavor. Each pattern may be underspecified or ambiguous and give different amounts of explicit or implicit evidence to different relations. Worse, as the complexity of language permits for one relation to be expressed in a multitude of ways, we may expect the distribution of patterns observed for each relation to be heavy-tailed, with a few patterns observed in high numbers and a large number of very rare patterns.

Take, for instance, the relation MARRIEDTO: Patterns that indicate this relation range from explicit and discriminative expressions, such as "*X married Y*" and "*X married to Y*", over entailment, such as "*X divorced from Y*" and "*X ex-wife of Y*", to mere implicit evidence, such as "*X fell in love with Y*". Here, *X* and *Y* are placeholders for an entity each. At the same time, these patterns may also express other relations at varying degrees; the pattern "*X divorced from Y*", for example, explicitly expresses the DIVORCEDFROM relation, while also entailing the MARRIEDTO relation. The desired result should reflect this and allow one-to-many assignments of patterns to relations, in which each pattern-relation assignment is weighted according to a *distinctiveness* value: High distinctiveness indicates a pattern that explicitly and unambiguously evokes a relation, low distinctiveness more implicit or ambiguous patterns.

In this paper, we examine more closely the task of discovering and ranking discriminative patterns for each relation and the impact of the choice of pattern generation scheme on overall URE results. By focusing on patterns, URE benefits in two ways: On the one hand we show that an informed feature generation strategy can markedly reduce the amount of underspecified and ambiguous patterns in the pair-pattern matrix, thereby significantly improving clustering

[1] In this paper, we refer to relationship types as *relations* and to instances of relationship types as *relation instances*.

approaches. On the other hand, this allows us to extend URE to not only identify relations, but also to find and rank a list of patterns for each relation that can be used in subsequent information extraction.

Contributions. We propose an unsupervised approach that identifies relations in a corpus of unknown content by clustering entity pairs and characterizes each relation by finding a list of patterns ordered according to the amount of explicit evidence they give to the presence of the identified relation. The contributions of this paper can be summarized as follows:

Algorithm for feature selection in a dependency graph. We propose an algorithm that selects possible patterns for a given entity pair in a dependency path, as an extension of the *shortest path* method. The approach is capable of capturing a wider range of phenomena than previous part-of-speech based feature generation and filtering approaches by incorporating syntactic elements for long range dependencies, complements for light or support verbs, appositions and context for arguments in direct conjunction. We show that the proposed feature selection technique increases the clustering quality F-measure by 65% over baseline approaches and that identified patterns are better suited to be used in an information extraction task.

Method for computing weighted pattern-relation assignments. We propose an approach that uses clustering results to compile a set of pattern-relation assignments, weighted according to the amount of discriminative evidence each pattern gives to an assigned relation. The method is based on the theory of Discriminative Category Matching (Fung et al., 2002). We experimentally show that these assignments produce patterns suitable for the task of information extraction.

We evaluate the proposed method in two different tasks: A *clustering task* in which we evaluate our clustering approach on three ground truth datasets of different composition against three baseline approaches [2]. We investigate the impact of our proposed feature selection algorithm on overall clustering quality and its ability to find the optimal amount of relationships in different datasets. Secondly, an *information extraction task* in which we evaluate the ranked patterns on two gold standard corpora and compare precision and recall with a baseline approach.

The remainder of the paper is organized as follows: Section 2 reviews previous work in the area of clustering for unsupervised relation discovery. We outline several approaches used as baselines in our evaluation. Section 3 outlines our clustering approach. In Section 3.1 we illustrate our proposed algorithm for pattern extraction in dependency trees, and in Section 3.3 our proposed method for identifying and ranking discriminative patterns. Section 4 describes evaluation methods, experimental setup and datasets, and reports the results on the two evaluation tasks. Finally, Section 5 concludes this paper.

2 Related Work

Most previous work utilizes the pair-pattern matrix to either measure the similarity of pairs of words, or to measure the similarity between patterns for a number of different purposes. In this section we review this work with respect to our pattern extraction and clustering approach and identify evaluation baselines.

[2] The datasets used in our experiments are available on request for research purposes.

Relation discovery. (Rosenfeld and Feldman, 2007) cluster entity pairs in the pair-pattern matrix to identify semantic relations. The resulting clusters are interpreted as each representing one relation that holds between all entity pairs in the cluster. They use the text between two entities in a sentence as patterns, but also allow arbitrary word skips, meaning that for each sentence containing an entity pair a large number of features are generated. They cluster the matrix using *k*-means and hierarchical agglomerative approaches and find that better results are reached with a complex feature space. (Bollegala et al., 2010) propose a co-clustering approach that simultaneously clusters both entity pairs and patterns for identifying relations, using not only lexical, but also shallow syntactic patterns. They expand the feature set to also include prefix and postfix spans. More recently, (Wang et al., 2011) analyzed the impact of filtering techniques and found that overall clustering quality F-measure significantly increases by using a set of filters to eliminate patterns that are unlikely to represent a relation. They filter out a total of 80% of all observed patterns. They use the text between entities as patterns, without word skips, and include named entity class information into the feature set.

Contrary to previous approaches in relation discovery, we employ a feature generation technique that utilizes information from a dependency parser. Our observation is that current dependency parsers are becoming orders of magnitudes faster while retaining a sufficiently high precision and recall, see (Rush and Petrov, 2012) and (Zhang and Nivre, 2011). We comparatively evaluate our feature generation technique against baselines modeled after the three approaches mentioned above.

Similarity of patterns or words. Instead of using clustering to identify relations, much work has focused on measuring the pairwise similarity of patterns or words. (Turney, 2006) computes the pairwise similarity of lexical patterns to solve the problem of finding analogies between word pairs. (Turney, 2011) compares pairs of words using the distribution over patterns to find proportional analogies and evaluate this on corpora of word comprehension tests, such as analogy questions in SAT or TOEFL tests. By contrast, (Lin and Pantel, 2001) directly measure the pairwise similarity between patterns in dependency trees using the distribution over word pairs to find inference rules from text. (Sun and Grishman, 2010) extend this with a clustering approach to group patterns into clusters, which they use to guide semi-supervised relation extraction methods. While this approach returns clusters of patterns for each discovered relation, the clustering is "hard", meaning that each pattern is assigned to exactly one cluster. This is contrary to our intuition that each pattern may give different amounts of evidence to different semantic relations. Nevertheless, we use a reimplementation of this approach as baseline for the evaluation of our proposed pattern ranking method.

3 Relation Discovery and Pattern Ranking

We propose a method that takes as input a document collection, identifies relations by clustering entity pairs with a similar pattern distribution, and outputs a ranked list of patterns for each relation. This is done in three steps: First, we generate the pair-pattern matrix using a feature generation approach based on deep syntactic analysis as explained in Section 3.1. Second, we run a clustering algorithm to group entity pairs into clusters representing relations (see Section 3.2). Finally, we compute the distinctiveness for each pattern in each cluster based on the distribution of patterns both within and across generated clusters as detailed in Section 3.3.

3.1 Feature Generation Using Dependency Trees

The proposed feature generation algorithm takes as input a set of dependency parsed sentences and entity pairs[3]. For each sentence and entity pair it generates a list of patterns that are used as features for the entity pair. The method determines a set of *core tokens* by collecting all tokens on the shortest path between the two entities. It then finds a set of *optional tokens* by collecting all tokens linked to a core token with certain typed dependency. It generates one feature for each combination of the core tokens and the power set (the set of all possible subsets) of the optional tokens.

Typed dependencies that indicate possibly important information even if not on the shortest path were determined through experimentation. Simple examples of cases in which important information is not on the shortest path are negation and particles, which are directly connected to a verb (with the dependencies "*neg*" and "*prt*" respectively) but never function as a link on the path between two arguments bound by this verb. Other examples are appositions, which may be connected to an entity but are not themselves part of the shortest path (indicated by "*nn*" or "*appos*"), and light verb constructions in which only the verb, but not the typically more important noun is part of the shortest path. Another example - discussed in detail below - are two entities in conjunction that function as an argument for a verb.

The method consists of four steps:

Step 1: Compute the shortest path between subject and object. The shortest path between two entities in a dependency path serves as basis for our extraction method. Recent research shows that lexical tokens along the shortest path represent particularly discriminative patterns for extraction of binary and even higher-order relations (Etzioni et al., 2011; Akbik and Broß, 2009). By focusing on the tokens that syntactically link both entities, we can skip over tokens that are less likely to be relevant to the relationship. This step yields a list of core tokens likely to be relevant to the relation expressed between the two entities.

Step 2: Collect of a set optional tokens on the path. We collect all tokens that *may* be relevant to identifying a relation by iterating over each token on the shortest path and examining all typed dependencies of each token to non-path tokens. If the dependency is one of {*nn*, *neg*, *prt*, *poss*, *possessive*, *nsubj*, *nsubjpass*} we collect the target token into a list of optional tokens. This step yields a list of tokens to be added to the core list to produce a good extraction pattern.

Step 3: Generate features. We build the power set over all optional tokens and generate one feature for each combination of the shortest path and optional set. This power set includes the empty set as well, so the shortest path without any optional tokens is included in the features.

Step 4: Remove uninformative features. We filter out all features that consist only of closed-world word classes. Examples are features like "*X and Y*" or "*X of Y*". The intuition for this step is that such patterns are semantically too weak to be used as patterns and not suitable for clustering approaches.

The following example sentence illustrates the feature generation process: *"James Joyce and his longtime lover Nora Barnacle got married in 1931"*. Figure 1 depicts the sentence's dependency parse. Here, the shortest path is a "*conj*"-link, directly connecting the two entities "*James Joyce*"

[3]We use the Stanford dependency parser (Klein and Manning, 2003) and Stanford typed dependencies (De Marneffe et al., 2006) in our experiments.

and "*Nora Barnacle*". The resulting pattern "*X and Y*"[4] is highly ambiguous and therefore of limited use. We collect the tokens "*and*", "*his*", "*lover*" and "*married*" into a set of optional tokens and build its power set. By taking each combination of the power set and the shortest path (and after filtering non-informative features) we arrive at a total of five features. Table 1 lists them and compares them to shallow patterns as generated by (Turney, 2011) and (Wang et al., 2011).

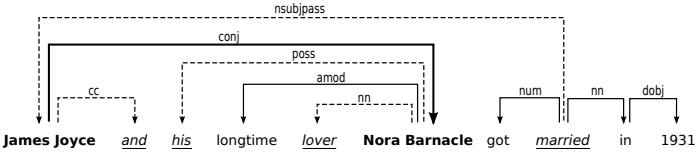


Figure 1: Dependency parse of the example sentence. The entity pair and shortest path are marked in bold. "*James Joyce*" and "*Nora Barnacle*" are directly connected with a "*conj*" link. Links to optional tokens are illustrated as dotted lines; optional tokens are underlined.

(Turney, 2011)	**X** and his longtime lover **Y**, **X** and his longtime * **Y**, **X** and his * lover **Y**, **X** and * longtime lover **Y**, **X** * his longtime lover **Y**, [..]
(Wang et al., 2011)	**PERSON** and his longtime lover **PERSON**
PROPOSED	**X** and lover **Y**, **X** and **Y** married, **X** and lover **Y** married, **X** and his lover **Y**, **X** and his lover **Y** married

Table 1: Features from different generation methods for the sentence in Figure 1. We observe that the features generated by the proposed approach all indicate the MARRIEDTO relation either explicitly or implicitly. By contrast, the shallow feature generation technique used by (Turney, 2011) produces a total of 24 patterns, many of which are highly underspecified. (Wang et al., 2011) generate only one overspecified feature.

3.2 Relation Discovery by Clustering Entity Pairs

From the features as generated according to Section 3.1 we build a pair-pattern matrix for all entity pairs observed at least 20 times in the corpus. This is accomplished by counting co-occurrences between patterns (features) and entity pairs. We cluster this vector space model using the k-means algorithm which partitions entity pairs into k clusters of similar variance[5]. Following (Bullinaria and Levy, 2007), we use the Cosine similarity to measure distances between feature vectors – a measure useful for highly sparse vectors like the ones at hand.

[4] In this case, the pattern "X and Y" is a verbalization of the entities X and Y being linked by the typed dependency "*conj*" for readability reasons.

[5] We use Apache Mahout (*http://mahout.apache.org/*) in our experiments.

The k-means algorithm requires us to manually specify the number of output clusters which in turn allows us to control the granularity of discovered relations. For instance, the cluster representing the relation ChildOf for low values of k is split into two clusters representing the relations SonOf and DaughterOf given higher values of k. Following (Rosenfeld and Feldman, 2007) we interpret each cluster within a clustering as a distinct, unlabeled relation and all entity pairs as relation instances.

3.3 Ranking Patterns by Distinctiveness

Clustering assigns each entity pair to a cluster and thereby implicitly produces a set of patterns per cluster, namely all non-zero features of the entity pairs in that cluster. The approach proposed here for ranking these patterns is based on two intuitions. The first being that clusters are representative of relations, meaning that the distribution of patterns in a cluster dominantly contains a single relation. This implies that patterns that are shared by a majority of entity pairs in a cluster are common ways of expressing a relation, while patterns that are shared only by few entity pairs are either less commonly used or provide only implicit evidence of a relation. To capture this intuition we compute pattern weights by adding up the counts per pattern over all entity pairs within a cluster. We normalize the value to compute the *significance* of a pattern:

$$Significance_{i,R} = \frac{\log_2(f_{i,R}+1)}{\log_2(P_R+1)} \tag{1}$$

As the equation shows, the significance of a pattern i within a cluster R is denoted as the logarithmic ratio of its weight $f_{i,R}$ normalized by the sum over all pattern weights P_R in the cluster.

The second intuition is that patterns that occur in more than one cluster may be ambiguous and lend different amounts of evidence to different relations. Such patterns therefore have low *clarity*. We collect the distributed evidence of these patterns across clusters and relate it to a pattern's highest significance over all clusters. The following equation measures this clarity for each pattern, with $0 \leq Clarity_i \leq 1$.

$$Clarity_i = \begin{cases} \log_2 \frac{n \cdot \max\limits_{j \in \{1..n\}} \left\{ Significance_{i,R_j} \right\}}{\sum\limits_{j=1}^{n} Significance_{i,R_j}} \cdot \frac{1}{\log_2 n}, & n > 1 \\ 1, & n = 1 \end{cases} \tag{2}$$

Thereby, a pattern i has a high $Clarity_i$ if it is significant in one cluster and insignificant in the others. (Note that $1/\log_2 n$ is a normalization factor.) If we observe a pattern only once and in one cluster its $Clarity$ is 1.

Following the theory of Discriminative Category Matching (DCM) (Fung et al., 2002), the overall *distinctiveness* of a pattern given a cluster is a combination of *Significance* and *Clarity* accordingly; with a normalization factor of $\sqrt{2}$:

$$Distinctiveness_{i,R} = \frac{Significance_{i,R}^2 \cdot Clarity_i^2}{\sqrt{Significance_{i,R}^2 + Clarity_i^2}} \cdot \sqrt{2} \tag{3}$$

We use this $Distinctiveness$ measure to re-weigh the pattern-cluster assignments and produce a ranked list of patterns for each cluster.

4 Evaluation

We quantify the proposed approach in two tasks: a *clustering task* to measure the impact of our feature generation method on overall clustering performance and evaluate the ability to discover relations. And an *information extraction task* to examine clustered patterns with respect to their usefulness to information extraction. Since ground truth is not usually readily available for large amounts of text, assessing the quality of large scale clustering results has proven to be difficult. We therefore use distant supervision based on the YAGO knowledge base to automatically construct various ground truth data sets. Details of the set-up, advantages and drawbacks of such an evaluation approach as well as measures used to analyze clustering quality against such ground truth are discussed in Section 4.1. Details on clustering evaluation and the information extraction task are discussed in Sections 4.2 and 4.3 respectively.

4.1 Experimental Setup

4.1.1 Datasets

The extrinsic evaluation of URE is problematic as it requires a document collection with exact knowledge regarding its content in the form of labeled relation triples. Several projects have constructed such a ground truth manually, which has a number of drawbacks: Firstly, there is a high cost involved in manually annotating sentences with relations, limiting the size of the ground truth as well as the ability to quickly generate new evaluation sets. Secondly, much care must be taken to ensure that no URE-specific assumptions are modeled into the ground truth, i.e. "overfitting" the ground truth to the capabilities of the algorithm that is to be evaluated. The inherent risk in manual annotation is the creation of a ground truth that does not realistically reflect the application scenario the URE approach is intended for.

We therefore choose a *distant supervision*-based approach to automatically generate a number of labeled training, test and evaluation sets. In distant supervision, an existing knowledge base of facts (triples consisting of two entities and a relation that holds between the entities) is used as support tool (Mintz et al., 2009). We use YAGO, a semantic knowledge base derived from Wikipedia, WordNet and GeoNames with knowledge of more than 10 million entities and around 447 million facts (Hoffart et al., 2011). The relations in YAGO are semantic labels, such as WasBornIn, ActedIn and DiedIn and are therefore different from a textual representation of these relations in a sentence.

The approach randomly selects a number of entity pairs from the knowledge base and retrieves from the Web[6] a set of sentences containing each entity pair. The assumption is that a sentence that contains an entity pair for which the knowledge base specifies a relation is likely to express it, either explicitly or implicitly. Accordingly, this allows the method to automatically label all retrieved sentences with relations, enabling the generation of a ground truth of arbitrary size. In order to assess the quality of the ground truth we manually examine 200 sentences with a total of 209 relations and 29 distinct relations[7]. We find that in 159 cases the relation is either explicitly or implicitly represented in the sentence, whereas in 50 cases the entity pair is present in the sentence but the YAGO relation between them could not be inferred from the text.

Examples for explicit, implicit and false sentences are given in Table 2. While imperfect, the

[6] We use the Bing API (*http://www.bing.com/developers/*) to retrieve sentences.

[7] In 9 cases, one entity pair has more than one relation in YAGO. Some persons, for example, both ActedIn and Produced a movie.

assumption therefore holds for approximately 76% of the generated ground truth. For our evaluation purposes we find this satisfactory, as this realistically simulates noise while reliably indicating the relational content of generated evaluation sets.

sentence retrieved	relation expressed
***Mystery Men** (1999) stars **Ben Stiller** as Mr. Furious.*	explicit
***Mystery Men** brought on board a talented cast from William H. Macy to **Ben Stiller**.*	explicit
*What was **Ben Stiller**'s character's super quality in **Mystery Men**?*	implicit
***Ben Stiller** does not think **Mystery Men** should be remade.*	false

Table 2: Sentences retrieved for the entity pair "*Ben Stiller*" and "*Mystery Men*", labeled in YAGO with relation ACTEDIN and the degree of explicitness: explicit, implicit or not at all.

We use this approach to generate 5 different ground truth datasets for the two evaluation tasks. For the clustering task, we generate 3 datasets of approximately 200.000 sentences, each with a different number of distinct relations. For the information extraction task, we generate two small gold standard corpora that are manually checked for correctness, with all falsely labeled sentences filtered out: **GOLD**, a corpus of 300 sentences that explicitly express the labeled relation, and **SILVER**, a corpus of 400 sentences that either explicitly or implicitly express the labeled relation. Refer to Table 3 for a list of all datasets.

dataset	# sentences	# relations	# entity pairs	manually cleansed
R10	200.000	10	12.000	false
R20	200.000	20	9.000	false
R30	200.000	30	6.000	false
GOLD	300	20	300	true
SILVER	400	20	400	true

Table 3: Datasets created using YAGO and distant supervision. The three large datasets differ in number of distinct relations and contained entity pairs. **GOLD** and **SILVER** are smaller, manually cleaned datasets.

4.1.2 Measures

We use **BCubed** for extrinsic clustering evaluation (Amigó et al., 2009), an effective measure extendable to overlapping clustering, which satisfies the following essential criteria for measuring cluster quality [8]:

- *Cluster homogeneity*, which rewards clusterings with pure clusters.
- *Cluster completeness*, which promotes "same label, same cluster" policy.
- *Rag bag*, which rewards introducing a garbage cluster over polluting pure clusters.
- *Small cluster preservation*, which penalizes spreading data points of a rare label across various clusters.

General BCubed precision and recall are computed based on *Multiplicity*, a measure of the minimum intersection between two data points o_i and o_j regarding their labels and cluster assignments. In our case this intersection contains 1 element at most, since we performed

[8] Cf. (Han et al., 2011, Ch. 1, p. 6) for a more verbose elaboration on these quality criteria and BCubed in general.

non-overlapping clustering. Depending on whether precision or recall is computed, Multiplicity is normalized with the amount of shared cluster assignments or shared labels respectively:

$$Multiplicity_{precision}(o_i, o_j) = \frac{min(|C(o_i) \cap C(o_j)|, |L(o_i) \cap L(o_j)|)}{|C(o_i) \cap C(o_j)|} \tag{4}$$

$$Multiplicity_{recall}(o_i, o_j) = \frac{min(|C(o_i) \cap C(o_j)|, |L(o_i) \cap L(o_j)|)}{|L(o_i) \cap L(o_j)|} \tag{5}$$

Here $C(o_i)$ denotes the set of cluster assignments of a data point o_i given a clustering and $L(o_i)$ the set of labels for a given data point o_i according to ground truth. Precision and recall are then calculated by averaging Multiplicity over all data points [9]:

$$Precision_{BCubed} = \frac{\sum_{i=1}^{n} \frac{\sum_{o_j : C(o_i) \cap C(o_j) \neq \emptyset} Multiplicity_{precision}(o_i, o_j)}{\| \{ o_j | C(o_i) \cap C(o_j) \neq \emptyset \} \|}}{n} \tag{6}$$

$$Recall_{BCubed} = \frac{\sum_{i=1}^{n} \frac{\sum_{o_j : L(o_i) \cap L(o_j) \neq \emptyset} Multiplicity_{recall}(o_i, o_j)}{\| \{ o_j | L(o_i) \cap L(o_j) \neq \emptyset \} \|}}{n} \tag{7}$$

In a final step $Precision_{BCubed}$ and $Recall_{BCubed}$ are combined to give the F_1-score.

4.2 Clustering Task

We evaluate our method's ability to identify relations by comparing it on ground truth datasets of different composition with several baselines. We use BCubed F-measure to judge overall clustering performance.

4.2.1 Baselines

We compare our feature generation method (referred to as **PROP**) to the three baseline approaches using shallow analysis that were introduced in Section 2: The first is based on (Turney, 2011) and (Rosenfeld and Feldman, 2007) and uses a lexical feature generation technique with arbitrary word skips. We refer to this approach as **TUR**. A second baseline is modeled after (Bollegala et al., 2010), uses shallow lexico-syntactic patterns including pre- and postfix spans, and is referred to as **BOL**. The third baseline, after (Wang et al., 2011), uses lexical patterns without word skips and incorporates named entity class information. Patterns containing the verbs *to say* or *to tell* are filtered. This method is referred to as **WAN**.

[9] Note that self-relation is not excluded. And that Multiplicity is defined only when the two data points share at least 1 cluster assignment or label respectively.

4.2.2 Results

The results of the comparative evaluation are visualized in Figure 2. It clearly shows the impact of using an informed feature generation method. On the R30 dataset, we note overall increases in F-measure of 65% over the next best approach. Overall F-measure is highest around a k of 30 at **0.445**. The next best approach is WAN at $k = 12$ with **0.288**. This shows that our feature generation algorithm is capable of finding patterns for many expressions that shallow feature generation methods miss. Also, as F-measure peaks around $k = 30$, the results indicate that the clustering mechanism can effectively model the relations contained in the corpus.

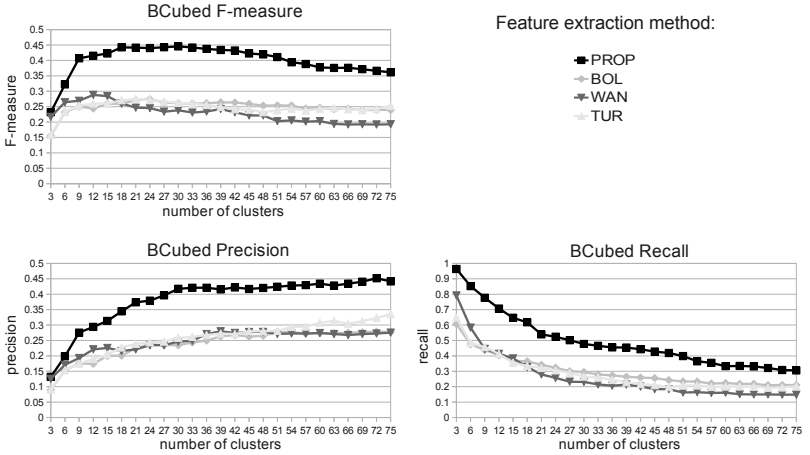


Figure 2: Clustering quality in terms of precision, recall and F_1-measure on the R30 dataset. The proposed feature generation approach (black line, square data points) outperforms all baseline approaches.

In order to examine this observation more closely, we use our approach on the R20 and R10 datasets, which contain 20 and 10 distinct relations respectively in the same amount of sentences as R30. The results are shown in Figure 3. Compared to results on the R30 dataset, we measure strong increases in F-measure on the R10 dataset which may be due to a much larger amount of examples per relation. However, we note that the results on the R20 and R30 are roughly similar, even though they are of different relational composition.

To gain more insight into these results, we inspect the data manually by randomly selecting clusters at different k. We make a number of observations. Firstly, when increasing k, the resulting clusters represent finer granularities of relations. The YAGO relation ChildOf, for example, is split into two clusters at higher k one representing the relation SonOf, the other DaugherOf. Similarly, the relation Created is split into multiple clusters, representing CreatedFilm, CreatedMusic and CreatedNovel respectively. The YAGO relation LocatedIn is split at various k into clusters of finer granularities, first into CityLocatedIn and VillageLocatedIn, then at an even higher k also into RiverLocatedIn. The YAGO relation IsAffiliatedTo is split at higher k into AffiliatedToSportsTeam and AffiliatedToPoliticalParty. These observations

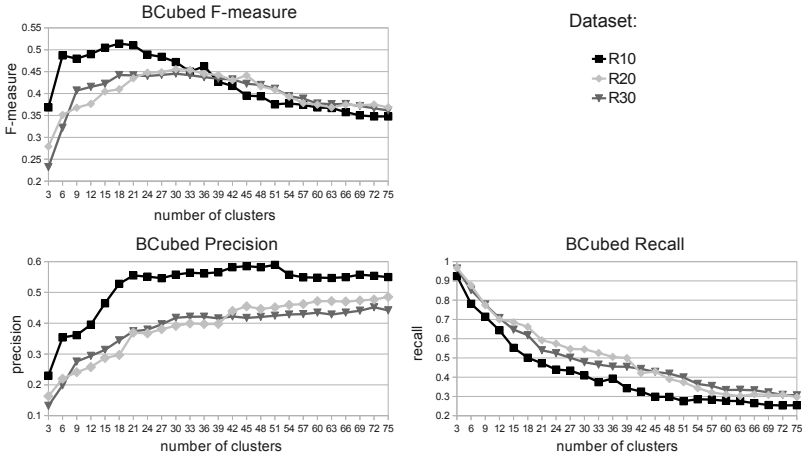


Figure 3: Clustering quality for the proposed approach on different datasets. Performance is best on dataset R10 which consists of relations typically expressed explicitly. R20 and R30 datasets each contain various more implicit or difficult to detect relations, causing lower F-measure.

indicate that the approach can be directed through parameterization to discover relations of varying granularity. However, we also note that some clusters split for difficult to interpret reasons, such as MARRIEDTO which splits into two clusters at higher k.

Generally, we observe that some relations are easier to identify by URE than others; relations like HASWONPRIZE or MARRIEDTO are often explicitly expressed and therefore easier to cluster. Other relations, like LIVESIN are often, if at all, very implicitly expressed causing difficulties to the algorithm. Other relations, such as DEALSWITH, which signifies trade relations between two countries, are almost impossible to find as very few sentences express this relation. Again other relations, such as KNOWNFOR are semantically very broad as there are any number of accomplishments (expressed in any number of ways) a person may be known for. This causes problems for a clustering approach. However, our approach is capable of finding a subset of all KNOWNFOR relations in a cluster that resembles the INVENTORINVENTED relation. This indicates that not only the amount of distinct relations in a corpus is important, but also how explicitly they are expressed and whether one relation dominates a given entity pair. We also find that the most highly ranked patterns usually characterize clusters very well. Example clustering results are shown in Table 4. Here, we find examples of explicit patterns, entailment and implicit patterns. We examine the patterns more closely in the information extraction task.

4.3 Information Extraction Task

We evaluate the ranked patterns on the GOLD and SILVER datasets using the generated patterns as classifiers. If the classifier finds a known pattern in a sentence it extracts and labels a relation, but only if the distinctiveness of the pattern is above the classifier's *threshold* setting. We compute a precision-recall curve for our proposed approach over a range of threshold values

ID	example entity pairs	example patterns	YAGO label
(1)	- Media General; WJAR - CBS Radio; WPHT - News Corporation; Fox	*1.* **Y** owned by **X** *3.* **X** operate **Y** *5.* **X** acquire **Y** *6.* **X** parent company of **Y** *32.* **X** gain for buying **Y**	[OWNS]
(2)	- Ronny Yu; Fearless - John Madden; Proof - Dana Brown; Highwater	*1.* **Y** film directed by **X** *2.* **Y** directed by **X** *7.* **X** 's film **Y** *14.* find trailer info for **Y** by **X**	[DIRECTED]
(3)	- Alan Turing; Turing test - Carlos Chagas; Chagas disease - Hans Geiger; Geiger counter	*1.* **Y** invented by **X** *2.* **X** creator of **Y** *3.* **Y** discovered by **X** *7.* **Y** named after **X** *19.* **X** inventor known for invention of **Y**	[KNOWNFOR]

Table 4: Example of clustering output. Each cluster contains a set of entity pairs and is defined via a ranked list of patterns. The rank is given in italics before each pattern. The YAGO relation labels are not part of the clustering output and added for evaluation purposes only.

from 0 to 1, see Figure 4. The results show that the threshold can be used to control the tradeoff between precision and recall. If the threshold is set high, the extractor only uses patterns with high distinctiveness and finds relations at high precision and lower recall. At lower threshold settings, recall gradually increases while precision decreases. This indicates that the proposed method computes a valid ranking of patterns. The ability to use such a threshold setting to influence the precision-recall tradeoff is a valuable feature for information extraction.

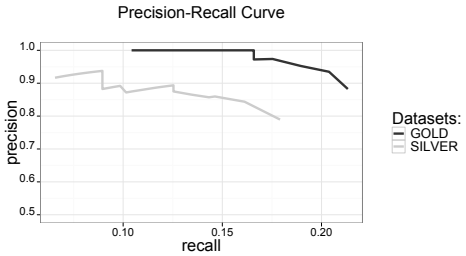


Figure 4: Precision-recall curves on the GOLD and SILVER gold standards for a range of threshold values. By lowering the confidence threshold, we trade high precision for an increase in recall.

We compare the approach against a reimplementation of (Lin and Pantel, 2001) in which patterns are directly clustered according to the distribution of entity pairs as introduced in (Sun and Grishman, 2010). This baseline (referred to as **PAN**) produces a hard clustering of patterns, without a distinctiveness value that can be used as threshold. We therefore compare the proposed approach against this baseline at two threshold levels: A threshold of 0 (PROP-0)

and 1 (PROP-1). For completeness, we also compare the proposed approach using **TUR** and **WAN** instead of our proposed feature generation method. This is denoted as **PROP-TUR** and **PROP-WAN** respectively, again at threshold settings of 0 and 1.

	GOLD			SILVER		
approach	**precision**	**recall**	**F_1-measure**	**precision**	**recall**	**F_1-measure**
PROP-0	0.88	**0.21**	**0.34**	0.79	**0.18**	**0.29**
PROP-1	**1**	0.1	0.18	**0.92**	0.07	0.13
PAN	0.58	0.15	0.24	0.51	0.1	0.17
PROP-WAN-0	0.6	0.03	0.06	0.56	0.05	0.09
PROP-WAN-1	0.19	0.1	0.13	0.21	0.11	0.14
PROP-TUR-0	0.28	0.14	0.19	0.26	0.14	0.18
PROP-TUR-1	0.31	0.18	0.23	0.29	0.17	0.21

Table 5: Results of the information extraction task on the GOLD and SILVER ground truths. The proposed approach outperforms the baseline in precision, recall and F_1-measure.

The results in Table 5 show that the proposed method outperforms the baseline. Shallow patterns, as used in **TUR** and **WAN**, are hardly usable for information extraction. Especially on the GOLD dataset, in which all relations are explicitly expressed, we note very high precision of the proposed approach. As expected, precision is lower on the SILVER dataset, which also includes implicit expressions of relations, but still higher than the baseline. Overall, the results show that the pattern-relation assignment of the proposed approach yields valuable results for the task of information extraction.

5 Conclusion

In this paper we have presented a method for Unsupervised Relation Extraction that discovers relations from unstructured text as well as finding a list of discriminative patterns for each discovered relation. We introduced a feature generation algorithm that utilizes dependency parse information and demonstrated that using an informed feature generation technique significantly improves overall clustering F-measure. We interpreted clustering results to produce a set of pattern-relation assignments weighted according to the distinctiveness of each assignment using the theory of Discriminative Category Matching. We demonstrated that the strength of an assignment indicates how reliably a pattern evokes a relation by using the patterns for information extraction at different confidence thresholds. We presented a thorough evaluation of both relation discovery and pattern ranking on 5 datasets of different composition. We believe our approach to be a promising step towards achieving the goals of URE.

Future work will focus on further evaluation on a range of different clustering algorithms in order to find an optimal approach. Specifically, we believe that using overlapping or fuzzy clustering algorithms may counterbalance problems of entity pair ambiguities. Furthermore, since using more samples positively affected clustering quality, we aim to scale up the method to large corpora and more broadly inspect the results at different levels of granularity.

Acknowledgements

We would like to thank our student Tuan Anh Do for his help with gathering training data. Alan Akbik and Priska Herger received funding from the European Union's Seventh Framework Programme (FP7/2007-2013) under grant agreement no ICT-2009-4-1 270137 'Scalable Preservation Environments' (SCAPE). Larysa Visengeriyeva, Holmer Hemsen and Alexander Löser received funding from the Federal Ministry of Economics and Technology (BMWi) under grant agreement "01MD11014A, 'MIA-Marktplatz für Informationen und Analysen' (MIA)".

References

Akbik, A. and Broß, J. (2009). Wanderlust: Extracting semantic relations from natural language text using dependency grammar patterns. In *1st. Workshop on Semantic Search at 18th. WWW Conference*.

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

Bollegala, D., Matsuo, Y., and Ishizuka, M. (2010). Relational duality: unsupervised extraction of semantic relations between entities on the web. In *WWW*, pages 151–160.

Bullinaria, J. and Levy, J. (2007). Extracting semantic representations from word co-occurrence statistics: A computational study. *Behavior Research Methods*, 39(3):510–526.

De Marneffe, M., MacCartney, B., and Manning, C. (2006). Generating typed dependency parses from phrase structure parses. In *Proceedings of LREC*, volume 6, pages 449–454.

Etzioni, O., Fader, A., Christensen, J., Soderland, S., and Mausam (2011). Open information extraction: The second generation. In *IJCAI*, pages 3–10.

Fung, G. P. C., Yu, J. X., and Lu, H. (2002). Discriminative category matching: Efficient text classification for huge document collections. In *ICDM*, pages 187–194.

Han, J., Kamber, M., and Pei, J. (2011). *Data Mining: Concepts and Techniques*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 3rd edition.

Hoffart, J., Suchanek, F. M., Berberich, K., Lewis-Kelham, E., de Melo, G., and Weikum, G. (2011). Yago2: exploring and querying world knowledge in time, space, context, and many languages. In *Proceedings of the 20th international conference companion on World wide web*, WWW '11, pages 229–232, New York, NY, USA. ACM.

Klein, D. and Manning, C. (2003). Accurate unlexicalized parsing. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics-Volume 1*, pages 423–430. Association for Computational Linguistics.

Lin, D. and Pantel, P. (2001). Dirt: discovery of inference rules from text. In *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 323–328. ACM.

Mintz, M., Bills, S., Snow, R., and Jurafsky, D. (2009). Distant supervision for relation extraction without labeled data. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 2-Volume 2*, pages 1003–1011. Association for Computational Linguistics.

Rosenfeld, B. and Feldman, R. (2007). Clustering for unsupervised relation identification. In *Proceedings of the sixteenth ACM conference on Conference on information and knowledge management*, pages 411–418. ACM.

Rush, A. and Petrov, S. (2012). Vine pruning for efficient multi-pass dependency parsing. In *NAACL '12*.

Sun, A. and Grishman, R. (2010). Semi-supervised semantic pattern discovery with guidance from unsupervised pattern clusters. In *Proceedings of the 23rd International Conference on Computational Linguistics: Posters*, pages 1194–1202. Association for Computational Linguistics.

Turney, P. (2008). The latent relation mapping engine: Algorithm and experiments. *Journal of Artificial Intelligence Research*, 33(1):615–655.

Turney, P. (2011). Analogy perception applied to seven tests of word comprehension. *Journal of Experimental & Theoretical Artificial Intelligence*, 23(3):343–362.

Turney, P. D. (2006). Expressing implicit semantic relations without supervision. In *ACL*.

Wang, W., Besançon, R., Ferret, O., and Grau, B. (2011). Filtering and clustering relations for unsupervised information extraction in open domain. In *CIKM*, pages 1405–1414.

Zhang, Y. and Nivre, J. (2011). Transition-based dependency parsing with rich non-local features. In *ACL (Short Papers)*, pages 188–193.

Automatic Detection of Point of View Differences in Wikipedia

Khalid Al Khatib[1] Hinrich Schütze[1] Cathleen Kantner[2]
(1) Institute for Natural Language Processing
University of Stuttgart
(2) Institute for Social Sciences
University of Stuttgart
khalid@ims.uni-stuttgart.de

ABSTRACT
We investigate differences in point of view (POV) between two objective documents, where one is describing the subject matter in a more positive/negative way than the other, and present an automatic method for detecting such POV differences. We use Amazon Mechanical Turk (AMT) to annotate sentences as positive, negative or neutral based on their POV towards a given target. A statistical classifier is trained to predict the *POV score* of a document, which reflects how positive/negative the document's POV towards its target is. The results of our experiments on a set of articles in the Arabic and English Wikipedias from the *people* category show that our method successfully detects POV differences.

KEYWORDS: sentiment analysis, content analysis, natural language processing.

Proceedings of COLING 2012: Technical Papers, pages 33–50,
COLING 2012, Mumbai, December 2012.

1 Introduction

In many areas of public discourse, content creators strive for objectivity. Reporters try to report the facts without bias; judges are expected to write opinions that are not influenced by personal views; encyclopedias are committed to what Wikipedia calls a "neutral point of view" (NPOV), defined as "... representing fairly, proportionately, and as far as possible without bias, all significant views that have been published by reliable sources".[1]

Even though objectivity is an important ideal, content creators cannot avoid being influenced by their background and context. There are two main reasons for this (Scheufele, 1999; Habermas, 2006; Littlejohn and Foss, 2010; D'Alessio and Allen, 2000). First, there are always many different non-equivalent ways of conveying a given piece of information. By choosing one vs. the other, the content creator introduces part of his/her point of view (POV) into the discourse. For example, "his wars caused the death of more than a million civilians" (a translation of a sentence in the French Wikipedia) puts Napoleon in a more negative light than "more than a million civilians died in his wars", which in turn is more negative than "more than a million civilians died in the wars fought between him and his enemies". This is so because the chain of causality between Napoleon and people being killed is more explicit in the first sentence than in the third. None of these sentences is a violation of Wikipedia's neutral point of view policy.

The second reason for different objective points of view is that the selection of what information to present is also influenced by background and context. If for space reasons only one of two equally relevant facts about a politician – one positive, one negative – can be added to a newspaper article, then this choice impacts how positive/negative the article is. This is an unavoidable dilemma journalists face on a daily basis. It is usually impossible to include all available information.

We call the difference between two objective documents, where one describes the subject in a more positive/negative way than the other, a *point of view difference* or *POV difference*. This paper develops a method that detects POV differences and quantifies their magnitude.

The automatic identification of POV and POV differences is of high potential for content analysis in the social sciences – which we take to include the humanities in this paper. Early content analysis was motivated by concerns about the declining quality of public debate in modern mass societies and tried to answer empirically questions such as: Do the media live up to their own quality standards of factually accurate and ideologically unbiased reporting? Are no relevant facts omitted? Are all relevant POVs equally represented? (Krippendorff, 2004, 55ff.)

There are also systematic reasons for the widespread empirical investigation of the evaluative positions taken by various speakers in the media. Our social world is permanently produced and reproduced, interpreted and criticized in social interactions of actors with their – often conflicting – intentions, values and reasons for actions (Berger and Luckmann, 1966; Habermas, 1984). This activity leaves traces – interpretable symbols, text and images. Evaluative aspects are almost always of central importance for the social science research question that motivates a content analysis. Typical questions that scholars, readers, the public and practitioners ask are: How favorably are the objects (e.g., social groups, politicians, policies, countries, corporations) perceived? Do different groups of people (e.g., migrants, citizens of different countries) view a certain object differently in a systematic fashion? How can this be explained? Which effect will this have?

[1] `http://en.wikipedia.org/wiki/Wikipedia:Neutral_point_of_view`

In most content analysis today, positive and negative presentation of subject are annotated – or coded – *manually*, an expensive and time-consuming method. To maximize intercoder agreement, the coding usually is restricted to explicit evaluative claims although some authors have also looked at stylistic means – such as irony and emotional language – for weakening or strengthening evaluations (Früh, 2011, 241–260). The result of this type of manual analysis is usually an aggregate value of the variable of interest that can then be used to compare groups (e.g., French vs. U.S. newspapers) or to analyze changes over time.

These efforts have not resulted in a widely shared research methodology. This may be due to the problem that there will never be an "objective" standard of what would constitute a pure, neutral way of reporting (Krippendorff, 2004, 55–57). Moreover, the subtle differences in evaluative tone and connotation that are of so much interest to social scientists are difficult to operationalize in a reliable way.

The approach presented in this paper overcomes these two problems of current content analysis. First, we develop a fully automatic method that is suitable for the analysis of large amounts of text and thereby reduces the obstacles that manual analysis and reliance on human coders present.

Second, while we acknowledge that the *absolute* assessment of POV is an important problem, we do not pretend to define an objective neutral standard in this paper. Instead, we cast the problem of assessing POVs as a *relative* problem and thereby avoid the difficulties inherent in attempting to define objective standards.

POV differences are also related to work on sentiment analysis in natural language processing (NLP). In contrast to most prior work in sentiment analysis, we are concerned only with *objective language* in this paper. For example, we do not address the analysis of editorials (which are intentionally opinionated) or of badly written Wikipedia articles (which violate the NPOV principle). The question as to how to automatically assess whether a piece of text written in objective language represents a positive or negative POV of the subject matter and to what extent has not been addressed before.

In addition to defining the problem of POV difference detection and proposing a method for solving it, we also provide an *evaluation gold standard*. It consists of articles and sentences from the Arabic and English Wikipedias that were annotated for POV and for POV differences using a combination of Amazon Mechanical Turk and student annotators. We chose the Arabic and English Wikipedias as the basis for our data set because we found that it contains many POV differences.

This paper is organized as follows. Section 2 introduces and motivates the concept of POV difference. Section 3 presents data acquisition and preparation. We describe our approach to the detection of POV differences in Section 4. Section 5 presents experiments and evaluates results. In Section 6, we discuss our results. Section 7 covers related work. Finally, conclusions and a brief summary of planned future work are given in Section 8.

2 Point of view (POV) differences

As a concrete example for a POV difference consider the French and Spanish Wikipedia articles about Napoleon.[2] Both articles are objective and meet the neutral POV criteria of Wikipedia. However, there is a POV difference between them: the French article is more positive than

[2]Based on the versions available online on 2012-04-01.

the Spanish article. We can find instances of the two types of reasons for POV differences that we discussed above: (i) different ways of describing a certain fact and (ii) different ways of selecting subsets of facts.

An example of different descriptions of the same facts is the phrase "sus agresivas guerras de conquista" 'his aggressive wars of conquest' in the Spanish Wikipedia. This amounts to a negative evaluation of Napoleon. Nowhere in the French article are Napoleon's wars called aggressive. Instead, his readiness to attack and the speed of his campaigns are referred to in more positive words: "offensive immédiate" 'immediate offensive', "marche forcée" 'forced march', "impressionante de rapidité" 'impressive for its rapidity'.

An example of a different selection of facts is the number of casualties during the Peninsular war. Only the Spanish article gives an estimate (300,000), which potentially casts a negative light on Napoleon as someone who is responsible for the loss of many lives.

Our goal in this paper is to develop a method that detects and quantifies such POV differences. Our approach is to first train a classifier that detects absolute POV. We then calculate POV difference between two articles as the difference of the absolute POV scores.

3 Data acquisition and preparation

Our approach to estimating POV differences for a pair of documents is to *estimate the absolute POV score* for each of the two articles of the pair and then *calculate the difference*. This approach will be described in detail in Section 4. To build a POV difference detector and evaluate it, we need a gold standard. Our gold standard consists of two parts, one for absolute POV and one for POV differences. While we could limit ourselves to only evaluating our main task, the estimation of POV differences, we decided instead to also evalute the quality of absolute POV scores. In this section, we describe the two gold standards we need for executing this plan: the gold standard for absolute POV scores and the gold standard for POV differences.

Gold standard for absolute POV. We first must decide which unit of text to create the gold standard for. Even though we are interested in the evaluation of entire documents, we do not annotate documents for two reasons. First, most documents will contain a mix of different POVs, so that a single label gives a statistical classifier noisy information. Second, reading and evaluating an entire document takes a long time for an annotator and would make gold standard creation expensive.

On the other hand, our units cannot be too small – e.g., words or phrases – because POV is a complex phenomenon that cannot be judged reliably at such a low level; the sentences about Napoleon in the introduction are examples for this.

Based on this reasoning we choose *the sentence* as our annotation unit. We annotate two sets of 1200 sentences, one for Arabic and one for English. The Arabic (resp. English) set consists of the first 20 sentences of 60 Arabic (resp. English) Wikipedia articles from the category *people*. We selected articles about people that are well known in both Western culture and Arabic culture because they are more likely to have been written by experienced authors and therefore to have a high quality.

The second major decision concerns the classification scheme for POV. We define three *POV classes*: positive, neutral and negative. These classes cover the potential cases of POV. We need a neutral class since many sentences do not contain any information that implies positive or negative POV.

Target: Mel Gibson

Paragraph: ... some audio recordings alleged to be of Gibson were posted on the internet. The same day Gibson was dropped by his agency , William Morris Endeavor. Civil rights activists alleged that Gibson had shown patterns of racism ... and called for a boycott of Gibson's movies.

Answer: Positive _____, Negative _____, Neutral _____

Figure 1: Interface of the AMT task.

The final decision concerns the annotators. We use Amazon Mechanical Turk (AMT). AMT has become a standard method for gold standard creation in NLP because annotations are of reasonable quality and comparatively low in cost (cf. (Alonso and Lease, 2011)).

For many objective statements, it is clear which POV class – positive, neutral or negative – applies to them. However, there is a certain subset of statements for which the decision is difficult. The different degrees of explicitness in describing a causal chain from Napoleon's actions to people dying in the introduction are a good example. As the statements become more explicit about the causal relation, at some point the sentence acquires a negative POV, but people differ as to when that point is reached.

Figure 1 shows the interface of the AMT task.[3] We ask non-expert workers to provide annotations for POV, a difficult decision for a subset of sentences. Thus, the design of the HIT (Human Intelligence Task) in AMT is crucial: in order to get acceptable agreement, the AMT task must be simple and easy to understand; definitions must be clear and the annotation interface well-structured.

Definitions of the three POV classes are provided in the instructions: the sentence has a positive (resp. negative, neutral) POV toward the target if it states that the target did something positive (resp. negative, neutral) or is described in a positive (resp. negative, neutral) way. No direct information about the target is also rated as neutral. Four examples from Wikipedia articles are given to help workers understand the task: one for positive, one for negative, and two for the neutral POV class. One neutral example shows a sentence that is directly relevant to the target, but is neither positive nor negative. The other neutral example is a negative sentence that is not relevant to assessing the POV of the article towards the target – e.g., because it talks about historical background that the target is not involved in. Because we found almost no sentences that had a mix of positive/negative elements, we did not explicitly include this case in the instructions.

The instructions are appropriately adapted for Arabic and English. They state that the Arabic (resp. English) task is only for Arabic (resp. English) native speakers. Even though the workers of the Arabic task have to be Arabic native speakers, the language of the instructions is English. All AMT workers know English since the AMT platform has only an English interface; so English as instruction language does not impose any additional restrictions on eligibility.

Each task includes one of the 1200 selected sentences (in blue color), the target that the article the sentence is extracted from is about (top line in Figure 1: "Mel Gibson"), and the surrounding paragraph (in black). To ensure sufficient context, we show to workers the entire paragraph

[3] We have reformatted the output that annotators see for space reasons and better legibility. E.g., we have omitted some text (marked "..."). Annotators see the entire paragraph without omissions.

	Arabic	English
agreement – all workers	.30	.48
agreement – majority of workers	.90	.95
Fleiss' κ	.215	.419

	Arabic	English
positive	.435	.47
neutral	.42	.34
negative	.145	.19

Table 1: Absolute POV gold standard: agreement (left); sentence label distribution (right).

containing the sentence to be annotated. Even though reading only the sentence is sufficient for the annotation task in most cases, sometimes it is difficult to determine the correct POV without reading some preceding or following sentences. For example, if the target sentence contains a pronoun, the annotator needs the context of the paragraph to resolve the reference.

We select one word from the sentence to be annotated randomly and render it in green. In the figure, the word is *agency*. The worker has to type this word in the corresponding answer field instead of using radio buttons or check boxes. We have found that this simple copying operation improves AMT annotation quality (Laws et al., 2011).

Workers are asked to label the sentence with one of three labels: positive, neutral and negative, based on the POV of the sentence toward the target. In Figure 1, the sentence to be annotated shows a negative POV towards the target (Mel Gibson), so we would expect the worker to annotate it as negative.

Incomplete assignments where the worker submits the task without giving all the required information and suspicious assignments where the worker spends only a few seconds on the task are rejected and republished to a different worker.

We use Fleiss' κ (Fleiss, 1971) (instead of Cohen's κ) to compute intercoder agreement because it can be applied when there are more than two raters and different items are rated by different raters (which is the case when using AMT). κ is .215 for Arabic, which is considered *fair* agreement; and .419 for English, which is considered *moderate* agreement (Landis and Koch, 1977).

We assign a sentence to the class chosen by at least two of the three annotators if there is such a class. If the three annotators assign three different labels (positive, neutral and negative), we assign the sentence to the neutral class. The proportion of sentences that have agreement between two or more workers is .90 for Arabic and .95 for English. Table 1 (left) summarizes agreement statistics for the absolute POV gold standard.

The difference between the agreement among the three workers and the agreement of the majority of workers (two workers) indicates problems with the quality of the AMT results. A number of workers did not follow the instructions very carefully. For example, some workers labeled sentences that are not directly relevant to the target as positive/negative, in violation of the instructions. Our impression is that one cause of such incorrect annotations is inexperience with AMT; in general, Arabic workers seem to have less experience than English workers.

Also, as we discussed above while there are many sentences that clearly belong to a particular POV class, other sentences are in the grey area between the two. One of the authors[4] assigned labels to a subset of Arabic sentences and compared them with the labels that were assigned to the sentences based on the majority-based gold standard label. We found that gold standard labels generally agree with our own judgments.

[4]Khalid Al Khatib, a native speaker of Arabic.

Table 1 (right) shows the distribution of labels. The positive class is more frequent than the negative class in both Arabic and English. The reason seems to be that the majority of people deemed worthy of a Wikipedia article are people like inventors, poets and athletes who are generally described in a positive way.

Gold standard for POV differences. As for the gold standard for absolute POV, we have to make decisions about three aspects for the gold standard for POV differences: unit of annotation, classification scheme and type of annotator.

For POV differences, our unit of annotation is a *pair of Wikipedia articles*. We need a pair of articles because a difference can only be annotated if the two things that we want to compare are represented. We have to go up to the level of documents because Wikipedias of different languages are not aligned on the sentence/paragraph level. We use Interwiki links to establish which articles in Arabic and English correspond to each other.

We use JWPL[5] to download articles that are in the *people* category and present in both the 20120114 Arabic and the 20111115 English Wikipedia. There are 16,000 such pairs.

We selected four categories that we sampled pairs of articles from. These categories are: Arab nationalists (5 pairs), Israeli nationalists (5 pairs), hand picked (5 pairs), and random (15 pairs). The motivation for the first two categories is that we expect strong POV differences for Arab and Israeli nationalists based on our personal knowledge of the two Wikipedias. Including these ten pairs ensures that a wide spectrum of POV differences is represented in the evaluation set. For the hand picked category, we selected people who are internationally well known both in the West and the Arab world. The motivation for this category is that we want to be able to present some results to the reader that are easy to interpret – without having to look up obscure personalities in Wikipedia. The random subset (15 pairs) is a standard random sample.

The length range of downloaded articles is 1–1128 sentences for English and 1–1050 sentences for Arabic. Short articles have many problems concerning quality and completeness and are often marked as stubs that require further work. We therefore impose the constraint that both articles must contain at least 50 sentences. We also exclude very long articles because they would make the annotation task too time-consuming and expensive.

The second design decision concerns the classification scheme. Here we propose a scheme with five different classes: much more positive, more positive, equal, more negative and much more negative. This scheme is more fine-grained than for absolute POV because a document pair is a rich source of information compared to a single sentence. There is sufficient information available to make more subtle distinctions such as between “more positive” and “much more positive”.

The final design decision concerns the annotators. Here we decided against AMT because reading, understanding and evaluating a pair of documents is a complex and time-consuming task that does not correspond to the typical HIT on AMT. More importantly, we need annotators for the task that are highly proficient in both Arabic and English. This type of annotator is difficult to find on AMT; and it is difficult to verify a high level of proficiency in a language on AMT.

For these reasons, we decided to hire engineering master students at our university for the annotation task. They are all students in an information technology master’s program, native

[5] `http://www.ukp.tu-darmstadt.de/software/jwpl/`

Target: Michael Faraday

Q1. The attitude of the English article toward the target compared to the Arabic article is:

1. Much more positive
2. More positive
3. Equal
4. More negative **X**
5. Much more negative

Q2. Briefly justify your answer to question 1.
Both Articles have a very positive attitude towards Faraday but I sensed it more in the Arabic one. For example in the marriage section of the Arabic article compared with the English one ,the attitude was much more positive and it mentioned that he was a loved , devoted, humble person which isn't mentioned in the English article. Also the controversy with Davy was only mentioned in the English article not the Arabic one. The Arabic article didn't mention anything negative towards Faraday.

Figure 2: Annotation setup and example of a completed annotation for POV differences. The annotator chose "More negative" ("**X**") and wrote an explanation (in italics).

speakers of Arabic and have an excellent command of English.

The annotation setup is shown in Figure 2. The annotator reads the Arabic and English Wikipedia articles and compares the two articles based on the articles' POV toward the target. The annotation guidelines state that information that is not directly related to the target must be ignored in the annotation decision and that the decision must be based solely on the contents of the two articles. Annotators are also instructed to not be influenced by their personal opinion, emotion or POV toward the target. The annotators have to justify their answers. In our experience, this helps the annotators to provide consistent and objective annotations.

Each pair of articles is annotated by three different annotators. We map the five point rating scale to $[-2,-1,0,1,2]$; e.g., "much more positive" is mapped to 2. The gold standard score Δ^{g}_{POV} for a pair of articles is then the average of the three scores given by the annotators (where the superscript g indicates "gold standard").

Intercoder agreement is $\alpha = .585$ (Krippendorff, 2004). This agreement is not as good as we would like it to be, but it is sufficient to evaluate our method; several other studies have published evaluation results based on gold standards with similar agreement (Bhardwaj et al., 2010; Brusk et al., 2010; Becker et al., 2012; Chen et al., 2012).[6]

4 Method

Absolute POV classification. For the task of determining the absolute POV – positive, neutral, negative – of a sentence, we adopt a statistical classification approach and use the Stanford MaxEnt classifier (Manning and Klein, 2003) with default parameters.

We refer to the probability of the positive (resp. negative) class for a sentence s as PosScore (resp. NegScore):

$$\text{PosScore}(s) = P(\text{positive}|s) \qquad \text{NegScore}(s) = P(\text{negative}|s)$$

Our features are bag of words (BOW) and letter k-grams (n-grams) where $2 \leq k \leq 6$.

[6]The two gold standards are available at `ifnlp.org/~schuetze/pov`.

For English, BOW and n-gram features are directly computed from text (as tokenized by the Stanford classifier) without any further linguistic preprocessing like lemmatization.

For Arabic, we investigate a number of different options for linguistic preprocessing. Arabic is a clitic language and highly inflectional. Normalization and lemmatization of Arabic text are beneficial preprocessing steps in many NLP applications. Lemmatization has been used widely in classification problems due to its ability to generate one form that matches many other related forms (Al Ameed et al., 2005). Therefore, in addition to the non-lemmatized surface forms, we used two lemmatization types: stem and root. We use *light stemming* to extract the stem: only frequent suffixes/prefixes are removed. In contrast, a word is reduced to its corresponding root by removing *all* affixes, not just frequent affixes (Al Ameed et al., 2005). We use the Arabic Text Mining tool for computing stems and roots.[7]

We use the term "bag of words" to refer to all word-level features, including "bag of stems" and "bag of roots".

Estimation of POV differences. To estimate POV differences we first need an aggregate measure of absolute POV on the document level. For this purpose, we define a document's POVScore as follows:

$$\text{POVScore}(d) = 1/|d|[\textstyle\sum_{s \in d}(\text{PosScore}(s) - \text{NegScore}(s))]$$

where $|d|$ is the number of sentences in the document. The POVScore is simply the difference of the averages of the PosScores and NegScores of the sentences of the article. This scoring method takes into consideration how positive or negative each sentence in the article is while ignoring any neutral meaning components. The higher POVScore(d), the more positive the article is. POVScore ranges from -1 to 1.

Our assumption is that most sentences of a Wikipedia article describe the target directly. This assumption can result in errors as we will discuss in Section 6.

We can now define the POV difference Δ_{POV} of a pair of articles as the difference of the POVScore of the English article and the POVScore of the Arabic article:

$$\Delta_{\text{POV}}(\{d_e, d_a\}) = \text{POVScore}(d_e) - \text{POVScore}(d_a)$$

where d_e is the English article of the pair and d_a is the Arabic article of the pair.

5 Experiments and results

Absolute POV classification. We train MaxEnt in tenfold cross validation on the gold standard described in Section 3 using the BOW and letter n-gram representations described in Section 4. Folds were constructed in a way that ensures that all sentences from a particular Wikipedia article are in same fold. The baseline in our experiment is to assign all sentences to the positive class, the most frequent class in both Arabic and English.

Table 2 gives evaluation results. The best result in each column is in bold. For Arabic, the classifier is better than the baseline for all six representations, both in accuracy and F_1. The overall best results are achieved using the stem representation with letter n-grams: accuracy is .584, F_1 is .474. The problem of root-based lemmatization is that many words with the same root have different meanings (Al Ameed et al., 2005). BOW without lemmatization ("BOW, tokens") performs less well because Arabic is highly inflected.

For English, accuracy is .608 and F_1 .533 using n-grams. Using BOW, accuracy is .587 and F_1

[7] `http://sourceforge.net/projects/ar-text-mining`

		Arabic		English	
		acc	F_1	acc	F_1
	baseline	.437	.206	.478	.214
BOW	tokens	.569†	.447†	.587†	.506†
BOW	roots	.555†	.464†		
BOW	stems	.561†	.460†		
n-grams	tokens	.574†	.453†	**.608**†	**.533**†
n-grams	roots	.580†	.470†		
n-grams	stems	**.584**†	**.474**†		

Table 2: Accuracy (acc) and F_1 of absolute POV classification. BOW = bag of words. †: significantly better than the baseline ($p < .01$).

	POVScore		
personality	English	Arabic	Δ_{POV}
Baruch Goldstein	.244	-.199	.443
Tzipi Livni	.549	.009	.540
Ariel Sharon	.259	-.113	.372
Wael Ghonim	.398	.528	-.130
Gamal Abdel Nasser	.296	.389	-.093
Saladin	.201	.320	-.119
Michael Jackson	.475	.617	-.142
Maria Sharapova	.579	.683	-.104
Steven Spielberg	.570	.744	-.174

Table 3: POV differences: Israeli, Islamic/Arabic and international personalities.

.506. The classifier outperforms the baseline by a fair margin in this case too. The best results are achieved using n-gram features.

All differences in accuracy and F_1 between the classifier and the baseline are statistically significant at $p < .01$.[8] However, the differences in accuracy and F_1 between using BOW and n-grams features are not significant.

Estimation of POV differences. We use the classifiers that achieved the best results in the previous section for computing POV differences: n-grams on tokens for the English absolute POV classifier and n-grams on stems for the Arabic absolute POV classifier.

For each of the 30 gold standard pairs (Section 3), we run the Arabic (resp. English) classifier on all sentences of the Arabic (resp. English) article. We then compute the two document scores and the difference $\Delta_{POV}(\{d_e, d_a\})$ (Section 4).

Table 3, shows examples for three Israeli, three Islamic/Arabic and three international personalities. Israeli personalities generally have more negative articles in Arabic than English. Islamic and Arabic personalities generally have more positive articles in Arabic than in English. International personalities also have more positive articles in Arabic than in English; we attribute this to our impression that Arabic Wikipedia authors tend to be more enthusiastic about the achievements of artists and athletes even if they are held in high regard in both the Arabic-speaking and the English-speaking world.

[8] Approximate randomization test (Noreen, 1989)

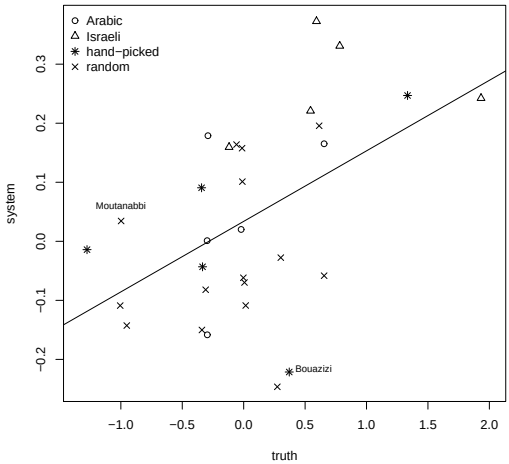


Figure 3: Estimated POV difference Δ_{POV} as a function of true POV difference $\Delta^{\text{g}}_{\text{POV}}$. The different symbols are used to show the category of each gold standard pair.

Figure 3 plots estimated POV difference Δ_{POV} as a function of true POV difference $\Delta^{\text{g}}_{\text{POV}}$ for the 30 pairs in the evaluation set. The correlation between human annotations and system scores is statistically significant for Spearman's ρ and Kendall's τ (both at $p < .005$).

We performed an error analysis for large divergences between true and predicted POV difference. The reason for divergences mostly seems to be that we use a simple BOW representation. We will illustrate this problem with two pairs of articles – Bouazizi and Moutanabbi (see marked points in Figure 3) – in the analysis we present in the next section.

6 Discussion

In our analysis of the errors we found that most of the cases with large divergences between automatically calculated POV difference and gold standard POV difference were due to the simple BOW representation we use – where we will use *BOW* in this section as a short hand for both bag of words and n-gram representations. This type of classifier is often not capable of detecting the subtle semantic nuances that are necessary to accurately assess absolute POV and POV differences. There are two main subcases of this general problem.

First, our assumption that all sentences in the Wikipedia article are describing the target is incorrect. (As in the rest of the paper we refer to the subject of the Wikipedia article as the target in this section.) There are sentences that describe people or events that have an impact on the life of the target, but are not directly about the target. These sentences can affect the system POVScore even though they are not communicating information relevant to the absolute

POV of the article towards the target. Identifying the subject of a sentence (target vs. something else) is not possible using BOW.

A special case of this are passages in Wikipedia about artists that contain titles and descriptions of movies, novels and other works of art. Again, this information can affect the system POVScore even though the fact that – for example – an actor played a murderer does not contribute to a negative POV about him.

The second subcase concerns parts of articles that *are* directly about the target, but not relevant for POV. The article may describe negative events that happened to the target, e.g., "Roosevelt contracted ... polio which resulted in permanent paralysis." Again, this will decrease the POVScore of the article even though the information reports something negative about the circumstances of the person's life that will not affect a reader's POV towards the person in a negative way.

An even subtler problem occurs if positive or negative words occur in a sentence that is directly relevant for POV towards the target, but these positive or negative words are in the scope of another word that reverses their meaning (cf. (Kessler and Schütze, 2012)). For example, the statement: *John started a war on violence against women* supports a positive POV towards John even though most of the words in the statement are negative words.

The immediate effect of the shortcomings of a BOW-based feature representation is an incorrect estimation of absolute POV. However, since these effects are somewhat random and will in most cases not affect Arabic and English to the same extent, the BOW problem can also give rise to incorrect POV differences.

In our data set, this is the reason that our system does not correctly predict the POV difference for Mohamed Bouazizi, the Tunisian who is credited with starting the Arab Spring (see data point marked "Bouazizi" in Figure 3). The system prediction is -.217 whereas the true score is .333. The problem with this pair of articles is that Bouazizi is described as a mostly positive person in both languages, but the circumstances of his life are described as tragic. Since our system does not distinguish between sentences that are directly relevant about the target vs. those that are not, this causes an incorrectly estimated POV difference.

A second example is Moutanabbi, a famous Arabic poet (system score: .034, truth: −1.00, data point marked "Moutanabbi" in Figure 3). His English article is positive, but his Arabic article is even more positive, hence the truth score of -1.00. The Arabic article is not handled well by a BOW representation for similar reasons as for Bouazizi. In particular, it contains poems about negative phenomena like mudslinging and sadness; and it describes the negative behavior of fellow poets towards Moutanabbi – this is negative, but will not create a negative impression of Moutanabbi in the mind of the reader.

There is one positive aspect of simplistic BOW representations. A potential concern is that the annotation of POV could be affected by annotator bias. Annotators have their own POV and even though we explicitly ask them to base their annotations solely on the content provided, there is a danger that they will be influenced by their personal views. However, in a BOW model this is not a problem: potentially incorrect annotations may contribute noise, but no systematic biases will be introduced. For example, even if an annotator is sympathetic with a murderer and the resulting annotation could mislead a classifier into believing that "murderer" is a positive word, there will be other annotations containing "murderer" that will counterbalance the incorrect annotation.

Note that for POV *differences*, personal annotator POV is less of a problem. We can expect a good annotator to provide a high-quality assessement of POV differences because a relative judgment about two articles is not in conflict with one's own personal views.

7 Related Work

Sentiment analysis (Pang and Lee, 2008) is mostly concerned with subjective language. However, the classification of objective language into positive vs. negative might also be considered sentiment analysis since the two tasks have a similar structure and face similar challenges.

Many papers have studied sentiment analysis in the news domain. Although this domain mainly contains objective content, subjectivity is also found to some extent, e.g., in editorials. Most previous work on sentiment analysis of news has ignored objective content. For example, Wiebe et al. (2005) released the Multi-Perspective Question Answering (MPQA) corpus. This corpus has detailed manual annotations of a set of 535 news articles. The corpus separates subjective and objective expressions. It has some information about objective content (such as the source and the target of the objective speech), but only has sentiment information about the subjective content. Our annotated corpus is different because it is concerned with objective language. Our task requires different annotation guidelines and, in general, a different setup for the annotation process compared to work on sentiment analysis.

Balahur et al. (2010), Abdul-Mageed and Diab (2011) and Balahur and Steinberger (2009) try to distinguish between positive and negative sentiment vs. good and bad news. Good and bad news are considered objective information and excluded from the classification process. In contrast, our method deals with good and bad objective information in the classification step.

Some prior work has classified financial news according to polarity. Some papers limit their classification to the subjective content of the news (e.g., (Agic et al., 2010)); other papers have classified objective content as well (Ahmad, 2006; Devitt and Ahmad, 2007; Shtrimberg, 2004). For example, Shtrimberg (2004) proposed an approach to classify news stories about companies as positive or negative. His classifier learned from a corpus where every news story about a company is labeled based on its impact on the future price of its stock. However, impact on price is different from positive/negative. For example, bad economic news can have a positive impact on stock prices if investors think it will make the Federal Reserve more likely to launch another round of quantitative easing. Our approach generates a score that indicates the POV of the article toward the subject matter and that is not directly related to the impact such information might have on the financial markets.

Another topic related to POV is media bias (Gentzkow and Shapiro, 2005, 2006). Some studies on this topic investigate bias in Wikipedia. Herzig et al. (2011) propose a novel annotation scheme as a basic step towards an automatic machine learning system to detect biased language in English Wikipedia. The scheme has multiple levels of bias tagging: the intra-sentential level, which includes polar-phrase, weasel, repetition, and personal-tone, and the sentence and entry level. The proposed scheme was applied to a set of articles from the service providers category in Wikipedia. Annotation categories distinguished between biased language and unbiased language. The authors conducted their annotation scheme based on the articles which explicitly *violate* the NPOV principle. Our approach studies POV differences under the assumption that Wikipedia articles mostly *adhere* to the NPOV principle and do not use biased linguistic expressions.

A line of research related to POV is work on perspectives and viewpoints (Lin et al., 2006; Paul et al., 2010). Most of this work uses Bitterlemons, a corpus of 594 articles each of which is written either from an Israeli or from a Palestinian perspective. Perspective classification (Lin and Hauptmann, 2006; Greene and Resnik, 2009; Klebanov et al., 2010) and modeling (Ahmed and Xing, 2010; Hardisty et al., 2010) then attempts to automatically detect the perspective of an article. Perspective and positive/negative POV are related, but different concepts; e.g., the sentences "political prisoners are released in Hamas deal" and "parties discuss new construction in Judea and Samaria" are both neutral or positive, but indicate different – Palestinian vs. Israeli – perspectives. In addition, much of the content of the Bitterlemons corpus is subjective – 66% of sentences according to Lin et al. (2006). In contrast, we address the problem of identifying positive/negative POV in objective language. Finally, the computational work on Bitterlemons is mostly on the document level whereas the measures we propose are based on sentences.

Massa and Scrinzi (2011) describe Manypedia, a web tool that supports comparing articles on the same subject in Wikipedia versions of different languages. They define the linguistic point of view as the potential difference in POV between Wikipedia articles from different languages due to the isolation of editor communities of these language versions. The tool provides users with multiple options such as translating the articles using Google Translate, extracting the most frequent words and showing information about editing of the article (e.g., total number of edits and editors). In contrast to our approach, Manypedia does not aim to provide automatic NLP analysis functionalities.

8 Conclusion and Future Work

The comparative analysis of differences – be they subtle or conspicuous – in the evaluation of particular subject matters is of great importance in the social sciences. The method we propose in this paper has two advantages. By choosing a relative instead of an absolute approach we avoid the old and still unsettled problem of defining an objective, neutral standard; and by taking a statistical classification approach, we provide an automatic method suitable for the analysis of large amounts of text.

Future work. We would like to address two problems in future work. First, our error analysis showed that most errors in predicting POV difference were due to our simple representation of sentences: bag of words. We would like to use more sophisticated representations that take into account the scope of positive and negative words; and also language understanding methods that can detect what a statement is about – the target itself or something not directly related to the target.

Second, we pointed out that there seem to be cultural differences in the magnitude of absolute POV. In particular, we found that international personalities are viewed in more positive light in the Arabic Wikipedia than in English. This means that there are at least two possible reasons for a POV difference: it can be due to a generally lower or higher level of absolute POV in one language; or it can be due to a genuinely different evaluation of a personality in two Wikipedias. We plan to distinguish these two different kinds of POV difference in future work.

Acknowledgments. We thank Deutsche Forschungsgemeinschaft for funding this research (DFG, SFB 732, D7); the anonymous reviewers, Christian Scheible, Wiltrud Kessler, Charles Jochim and Andrea Glaser for their valuable comments; and Amr Yassin, Ghada Dessouky, Mariam Hassib and Mirna Bouchra for performing the annotation.

References

Abdul-Mageed, M. and Diab, M. T. (2011). Subjectivity and sentiment annotation of modern standard Arabic newswire. In *Linguistic Annotation Workshop*, pages 110–118.

Agic, Z., Ljubesic, N., and Tadic, M. (2010). Towards sentiment analysis of financial texts in croatian. In *The 7th International Conference on Language Resources and Evaluation*, LREC '10.

Ahmad, K. (2006). Multi-lingual sentiment analysis of financial news streams. *PoS*, GRID '06:001.

Ahmed, A. and Xing, E. P. (2010). Staying informed: Supervised and semi-supervised multi-view topical analysis of ideological perspective. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, EMNLP '10, pages 1140–1150.

Al Ameed, H. K., Al Ketbi, S. O., Al Kaabi, A. A., Al Shebli, K. S., Al Shamsi, N. F., Al Nuaimi, N. H., and Al Muhairi, S. S. (2005). Arabic light stemmer: A new enhanced approach. In *Proceedings of the 2nd International Conference on Innovations in Information Technology*, IIT '05.

Alonso, O. and Lease, M. (2011). Crowdsourcing for information retrieval: Principles, methods, and applications. In *Proceedings of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '11, pages 1299–1300.

Balahur, A. and Steinberger, R. (2009). Rethinking sentiment analysis in the news: From theory to practice and back. In *Proceedings of the 1st Workshop On Opinion Mining and Sentiment Analysis*.

Balahur, A., Steinberger, R., Kabadjov, M., Zavarella, V., van der Goot, E., Halkia, M., Pouliquen, B., and Belyaeva, J. (2010). Sentiment analysis in the news. In *Proceedings of the 7th International Conference on Language Resources and Evaluation*, LREC '10.

Becker, L., Basu, S., and Vanderwende, L. (2012). Mind the gap: Learning to choose gaps for question generation. In *The 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, NAACL-HLT '12, pages 742–751.

Berger, P. L. and Luckmann, T. (1966). *The Social Construction of Reality: A Treatise in the Sociology of Knowledge*. Anchor books. Doubleday.

Bhardwaj, V., Passonneau, R. J., Salleb-Aouissi, A., and Ide, N. (2010). Anveshan: A framework for analysis of multiple annotators' labeling behavior. In *Proceedings of the 4th Linguistic Annotation Workshop*, LAW IV '10, pages 47–55.

Brusk, J., Artstein, R., and Traum, D. R. (2010). Don't tell anyone! Two experiments on gossip conversations. In *Proceedings of The 11th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, SIGDIAL '10, pages 193–200.

Chen, J., Ding, R., Jiang, S., and Knudson, R. (2012). A preliminary evaluation of metadata records machine translation. *The Electronic Library*, 30(2).

D'Alessio, D. and Allen, M. (2000). Media bias in presidential elections: A meta-analysis. *Journal of Communication*, 50:133–156.

Devitt, A. and Ahmad, K. (2007). Sentiment polarity identification in financial news: A cohesion-based approach. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, ACL '07.

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

Früh, W. (2011). *Inhaltsanalyse. Theorie und Praxis*. UTB.

Gentzkow, M. and Shapiro, J. (2005). Media bias and reputation. Working Paper 11664, National Bureau of Economic Research.

Gentzkow, M. and Shapiro, J. M. (2006). What drives media slant? Evidence from U.S. daily newspapers. Working Paper 12707, National Bureau of Economic Research.

Greene, S. and Resnik, P. (2009). More than words: Syntactic packaging and implicit sentiment. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, NAACL '09, pages 503–511.

Habermas, J. (1984). *The Theory of Communicative Action. Reason and the Rationalization of Society*, volume 1. Heinemann.

Habermas, J. (2006). Political communication in media society: Does democracy still enjoy an epistemic dimension? The impact of normative theory on empirical research. *Communication Theory*, 16:411–426.

Hardisty, E. A., Boyd-Graber, J., and Resnik, P. (2010). Modeling perspective using adaptor grammars. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, EMNLP '10, pages 284–292.

Herzig, L., Nunes, A., and Snir, B. (2011). An annotation scheme for automated bias detection in Wikipedia. In *Proceedings of the 5th Linguistic Annotation Workshop*, LAW V '11, pages 47–55.

Kessler, W. and Schütze, H. (2012). Classification of inconsistent sentiment words using syntactic constructions. In *Proceedings of the 24th International Conference on Computational Linguistics*, COLING 24.

Klebanov, B. B., Beigman, E., and Diermeier, D. (2010). Vocabulary choice as an indicator of perspective. In *Proceedings of the The 48th Annual Meeting of the Association for Computational Linguistics*, pages 253–257.

Krippendorff, K. (2004). *Content Analysis: An Introduction to Its Methodology*. Sage Publications.

Landis, J. R. and Koch, G. G. (1977). The measurement of observer agreement for categorical data. *Biometrics*, 33(1):159–174.

Laws, F., Scheible, C., and Schütze, H. (2011). Active learning with Amazon Mechanical Turk. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, EMNLP '11, pages 1546–1556.

Lin, W.-H. and Hauptmann, A. (2006). Are these documents written from different perspectives? A test of different perspectives based on statistical distribution divergence. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, ACL '06, pages 1057–1064.

Lin, W.-H., Wilson, T., Wiebe, J., and Hauptmann, A. (2006). Which side are you on? Identifying perspectives at the document and sentence levels. In *Proceedings of the 10th Conference on Computational Natural Language Learning*, CoNLL-X '06, pages 109–116.

Littlejohn, S. W. and Foss, K. A. (2010). *Theories of Human Communication*. Waveland.

Manning, C. and Klein, D. (2003). Optimization, maxent models, and conditional estimation without magic. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology: Tutorials - Volume 5*, NAACL-Tutorials '03.

Massa, P. and Scrinzi, F. (2011). Exploring linguistic points of view of Wikipedia. In *Proceedings of the 7th International Symposium on Wikis and Open Collaboration*, WikiSym '11, pages 213–214.

Noreen, E. (1989). *Computer-Intensive Methods for Testing Hypotheses: An Introduction*. Wiley.

Pang, B. and Lee, L. (2008). Opinion mining and sentiment analysis. *Found. Trends Inf. Retr.*, 2(1-2):1–135.

Paul, M. J., Zhai, C., and Girju, R. (2010). Summarizing contrastive viewpoints in opinionated text. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, EMNLP '10, pages 66–76.

Scheufele, D. A. (1999). Framing as a theory of media effects. *Journal of Communication*, 49(1):103–122.

Shtrimberg, I. (2004). Good news or bad news? Let the market decide. In *AAAI Spring Symposium on Exploring Attitude and Affect in Text*, pages 86–88.

Wiebe, J., Wilson, T., and Cardie, C. (2005). Annotating expressions of opinions and emotions in language. *Language Resources and Evaluation*, 39(2-3):165–210.

SpeedRead: A Fast Named Entity Recognition Pipeline

Rami Al-Rfou' *Steven Skiena*
Department of Computer Science
Stony Brook University
NY 11794, USA
{ralrfou, skiena}@cs.stonybrook.edu

ABSTRACT
Online content analysis employs algorithmic methods to identify entities in unstructured text. Both machine learning and knowledge-base approaches lie at the foundation of contemporary named entities extraction systems. However, the progress in deploying these approaches on web-scale has been been hampered by the computational cost of NLP over massive text corpora. We present SpeedRead (SR), a named entity recognition pipeline that runs at least 10 times faster than Stanford NLP pipeline. This pipeline consists of a high performance Penn Treebank-compliant tokenizer, close to state-of-art part-of-speech (POS) tagger and knowledge-based named entity recognizer.

KEYWORDS: Tokenization, Part Of Speech, Named Entity Recognition, NLP pipelines.

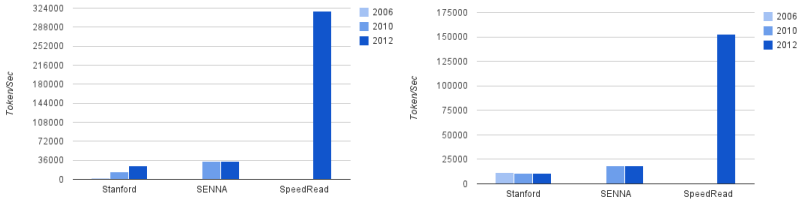
Proceedings of COLING 2012: Technical Papers, pages 51–66,
COLING 2012, Mumbai, December 2012.

1 Introduction

Information retrieval (IR) systems rely on text as a main source of data, which is processed using natural language processing (NLP) techniques to extract information and relations. Named entity recognition is essential in information and event-extraction tasks. Since NLP algorithms require computationally expensive operations, the NLP stages of an IR system become the bottleneck with regards to scalability (Pauls and Klein, 2011). Most of the relevant work, conducted by researchers, was limited to small corpora of news and blogs because of the limitation of the available algorithms in terms of speed. Most of the NLP pipelines use previously computed features that are generated by other NLP tasks, which adds computational cost to the overall NLP pipeline. For example, named entity recognition and parsing need POS tags; co-reference resolution requires named entities. In effect, we anticipate lower speed for future tasks.

A conservative estimate of a sample of the web news and articles can add up to terabytes of text. On such scale, speed makes a huge difference. For example, considering the task of annotating 10 TiBs of text with POS tags and named entities using a 20 CPU cores computer cluster would take at least 4 months using the fastest NLP pipeline available for researchers, our calculations show. Using our proposed NLP pipeline the time is reduced to a week.

Several projects have tried to improve the speed by using code optimization. Figure 1a shows that Stanford POS tagger has improved throughout the years, increasing its speed by more than 10 times between 2006 and 2012. However, the current speed is twice slower than the SENNA POS tagger.



(a) POS taggers performance.

(b) NER taggers performance.

Figure 1: Performance of NLP pipelines through the years over POS and NER tagging. Stanford POS tagger uses L3W model, its speed in 2006 is slow to be apparent in the graph. Stanford tagger uses CONLL 4 classes model. SENNA pipeline was first released in 2008

In this paper, we present a new NLP pipeline, SpeedRead, where we integrate global knowledge extracted from large corpora with machine learning algorithms to achieve high performance. Figures 1a and 1b show that our pipeline is 10 times faster than Stanford pipeline in both tasks: POS tagging and NER tagging. Our design is built on two principles: (1) majority of the words have unique annotations and tagging them is an easy task; (2) the features extracted for the frequent words should be cached for later use by the classifier. Both principles are simple and they show how to bridge the large gap in performance between current systems and what can be achieved.

Our work makes the following contributions:

Phase	SpeedRead Relative Speed
Tokenization	11.8
POS	11.1
NER	13.9
TOK+POS+NER	18.0

Table 1: SpeedRead relative speed to Stanford pipeline.

- *Exposing the performance limitations of the current NLP systems:* We show that there is an algorithmic room for improving performance, rather than relying solely on optimizing the code.
- *High performance NLP pipeline that supports English tokenization, POS tagging and named entity recognition:* Novel design decisions that are not taken by most of the available tools to explore new area of the accuracy-performance space. SpeedRead is available under an open-source license. The code's organization is simple and it is written in Python for its readability benefits. This makes it easier for others to contribute and hack.
- *Techniques to reduce computation needed for sequence tagging tasks:* We distinguish between ambiguous and non-ambiguous words. We use the larger copora to calculate the frequent words and their frequent tags. We cache the extracted features of the most frequent words to avoid unnecessary calculations and boost performance.

Figure 2 shows the design of the SpeedRead pipeline. The first stage is tokenization followed by POS tagging that is used as an essential feature to decide the boundaries of the named entities' phrases. Once the phrases are detected, a classifier decides to which category these named entities belong to.

This paper is structured as follows. In Section 2, we discuss the current NLP pipelines, available to researchers. Section 3 discusses SpeedRead tokenizer's architecture, speed and accuracy. In Section 4, we discuss the status of the current state-of-art POS taggers and describe SpeedRead new POS tagger. Section 5 describes the architecture SpeedRead's named entity recognition phase. Finally, in Section 5.2, we discuss the status of the pipeline and the future improvements.

1.1 Experimental Setup

All the experiments presented in this paper were conducted on a single machine that has i7 intel 920 processor running on 2.67GHz, the operating system used is Ubuntu 11.10. The time of execution is the sum of *{sys, user}* periods calculated by the Linux command `time`. The speeds that are reported are calculated by averaging the execution time of five runs without considering any initialization times.

2 Related Work

There are many available natural language processing packages available for researchers under open source licenses or non-commercial ones. However, this section is not meant to review the literature of named entity recognition research as this is already available in (Nadeau and Sekine, 2007). We are trying to discuss the most popular solutions and the ones we think are interesting to present.

Stanford NLP pipeline (Toutanova and Manning, 2000; Toutanova et al., 2003; Klein et al., 2003; Finkel et al., 2005; Lee et al., 2011) is one of the most popular and used NLP packages.

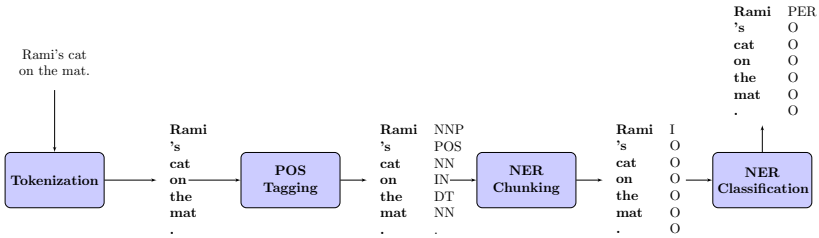


Figure 2: SpeadRead named entity recognition pipeline. First, tokenization split the words into basic units to be processed in the later phases. POS tagging identifies to which speech categories words belong to. There are 45 part of speech category, we are mainly interested in nouns. Chunking identifies the borders of phrases that make up the named entities. In the above sentence, the named entity, Rami, is one word phrase. The last stage classifies each phrase to one of four categories; Person, Location, Organization or Miscellaneous.

The pipeline is rich in features, flexible for tweaking and supports many natural languages. Despite being written in Java, there are many other programming language bindings that are maintained by the community. The pipeline offers a tokenization, POS tagging, named entity recognition, parsing and co-referencing resolution. The pipeline requirements of memory and computation are non-trivial. To accommodate the various computational resources, the pipeline offers several models for each task that vary in speed, memory consumption and accuracy. In general, to achieve good performance in terms of speed, the user has to increase the memory available to the pipeline to 1-3 GiBs and choose the faster but less accurate models.

More recent efforts include SENNA pipeline. Even though it lacks a proper tokenizer, it offers POS tagging, named entity recognition, chunking, semantic role labeling(Collobert and Weston, 2008) and parsing (Collobert, 2011). The pipeline has simple interface, high speed and small memory footprint (less than 190MiB).

SENNA builds on the idea of deep learning of extracting useful features from unlabeled text. This unsupervised learning phase is done using auto-encoders and neural networks language models. It allows the pipeline to map words into another space of representation that has lower dimensionality. SENNA maps every word available in its 130 thousand word dictionary to a vector of 50 floating numbers. These vectors are then merged into a sentence structure using convolutional networks. The same architecture is then trained on different tasks using annotated text to generate different classifiers. The big advantage of taking this approach is the lesser amount of engineering that it requires to solve multiple problems.

NLTK (Bird et al., 2009) is a set of tools and interfaces to other NLP packages. Its simple APIs and good documentation makes it a favorable option for students and researchers. Written in Python, NLTK does not offer great speed or close to state-of-art accuracy with its tools. On the other hand, it is well maintained and has great community support.

WikipediaMiner (Milne and Witten, 2008) detects conceptual words and named entities; it also disambiguates the word senses. This approach can be modified to detect only the words that represent entities, then using the disambiguated sense, it can decide which class the entity belongs to. Its use of the Wikipedia interlinking information is a good example of the power

of using knowledge-based systems. Our basic investigation shows that the current system needs large chunks of memory to load all the interlinking graph of Wikipedia and it would be hard to optimize for speed. TAGME (Ferragina and Scaiella, 2010) is extending the work of WikipediaMiner to annotate short snippets of text. They are presenting a new disambiguation system that is faster and more accurate. Their system is much simpler and takes into account the sparseness of the senses and the possible lack of unambiguous senses in short texts.

Stanford and SENNA performed the best in terms of speed and quality in our early investigation. Therefore, we will focus on both of them from now on as good representatives of a wide range of NLP packages.

3 Tokenizer

The first task that an NLP pipeline has to deal with is tokenization and sentence segmentation (Webster and Kit, 1992). Tokenization target is to identify tokens in the text. Tokens are the basic units which need not to be processed in the subsequent stages. Part of the complexity of tokenization comes from the fact that the definition of what a token is, depends on the application that is being developed. Punctuation brings another level of ambiguity; commas and periods can play different roles in the text. For example, we do not need to split a number like $1,000.54$ into more units whereas we need to split a comma-separated list of words. On the other hand, tokenization is important as it reduces the size of the vocabulary and improves the accuracy of the taggers by producing similar vocabulary to the one used for training.

As many NLP tasks' gold standards are dependent on Penn Treebank(PTB), a corpus of annotated text and parsed sentences taken from Wall Street Journal (WSJ), we opted for their tokenization scheme.

Searching for good tokenizers, we limited our options to the ones that support Unicode. We believe that Unicode support is essential to any applications that depends on the pipeline. Stanford tokenizer and Ucto (Gompel, 2012) projects offer almost Penn Treebank (PTB) compliant tokenizers plus other variations that are richer in terms of features.

Table 2 shows that there is a substantial gap in performance between basic white space tokenizer (words are delimited by spaces or tabs and sentences are split by new line characters) and more sophisticated tokenizers as Stanford tokenizer and Ucto. We observed that the Stanford tokenizer is 50 times slower than the baseline (WhiteSpace tokenizer), which motivated us to look at the problem again.

The Stanford tokenizer is implemented using JFlex, a Java alternative to Flex. The tokenizer matured over the years by adding more features and modes of operation which makes it harder for us to modify. Ucto uses C++ to compile a list of regular expressions that passes over the text multiple times.

SpeedRead, like the Stanford tokenizer, uses a lexical analyzer to construct the tokenizer. However, we use different generating engine than the (F)lex family. SpeedRead depends on Quex (Schafer, 2012), a lexical analyzer generator, to generate our tokenizer. Quex makes different trade-off decisions than the usual lex tools when it comes to the tokenizer's generation time. Quex spends more time optimizing its internal NFA to produce a faster engine. While generating a tokenizer from a normal lex file can take few minutes, Quex takes hours for the same task. However, Quex supports Unicode in multiple ways and has similar description language to lex, but is cleaner and more powerful. The extensive multiple mode support makes

Tokenizer	Word/Second	Relative Speed
Ucto	185,500	0.8
PTB Sed Script	214220	0.96
Stanford	222,176	1.0
SpeedRead	2,626,183	**11.8**
WhiteSpace	11,130,048	50.0

Table 2: Speed of different tokenizers measured as word/second; Every tokenizer generates different number of tokens. For consistency, the original words count before tokenization used to calculate the speed. Words count is calculated using linux command *wc*. Execution time includes both tokenization and sentence segmentation times with the exception that the original PTB Sed Script does not do sentence segmentation. Ucto's default configuration is used. Stanford tokenizer runs with strict PTB flag turned on.

it easy to write the lexical rules in understandable and organized way. All of that results in a fast C implementation of a Penn Treebank compliant tokenizer as Table 2 shows.

As a design decision, we did not support some features which we believe will not affect the accuracy of the tokenizer. Table 3 shows the features which are not implemented. While some of the features are easy to add as supporting contractions, others, involving abbreviations especially *U.S.*, prove to be complex (Gillick, 2009).

Feature	Text	PTB	SpeedRead
Reordering	Japan. ...	Japan	Japan . ...
Punctuation addition	U.S."	U.S. . "	U.S. "
Contractions	gimme	gim me	gimme

Table 3: Some features that are not implemented in SpeedRead Tokenizer. Contractions that involves apstrophes are implemented in SpeedRead. For instance, *can't* will be tokenized to *ca n't*.

Table 4 shows that the accuracy of our tokenizer is Penn Treebank compliant, despite the missing features. Moreover, running SpeedRead and Stanford tokenizers over Reuters RCV1 corpus results in approximately 214, 215 million tokens consecutively.

3.1 Sentence Segmentation

While PTB offers a set of rules for tokenization, their tokenizer assumes that the sentences are already segmented, which is done manually. SpeedRead's sentence segmentation uses the same rules that Stanford tokenizer uses. For instance, a period is an end of a sentence unless it is part of an acronym or abbreviation. The list of rules to detect those acronyms and abbreviations are taken from the Stanford tokenizer. Any quotations or brackets, that follow the end of the sentence, will be part of that sentence. Running SpeedRead's sentence segmentation on Reuters RCV1 generated 7.8 million sentences, while Stanford tokenizer generated 8.2 million sentences.

Tokenizer	Accuracy
PTB Sed Script	100.0%
Stanford tokenizer	99.7%
SpeedRead	**99.0**%
White Space	0.0%

Table 4: Accuracy of the tokenizers over the first 1000 sentence in the Penn Treebank. The gold standard was created by getting the tokenized text from the parse trees and manually segment the original text into sentences according to the parse trees. Errors in differentiating between starting and ending quotations are not considered. Not supporting MXPOST convention, replacing brackets with special tokens, is not considered necessary.

4 Part of Speech Tagger (POS)

Earlier work to solve the POS tagging problem relied on lexical and local features using maximum entropy models (Toutanova and Manning, 2000). Later, more advanced models took advantage of the context words and their predicted tags (Toutanova et al., 2003) to achieve higher accuracy. As POS tagging is a sequence tagging problem, modeling the sequence into a Maximum Entropy Markov Model (MEMM) or Conditional Random Fields (CRF) model (to infer the probability of the tags' sequences) seems to be the preferred option. The probability of each tag is computed using log-linear model with features that include large enough context words and their already-computed tags. This transforms every instance of the problem into a large vector of features that is expensive to compute. Then the sequence of vectors are fed to graphical model to compute the probability of each class, using the inference rules. The size of features' vector and the inference computation are the same regardless of the complexity of the problem.

Although the previous algorithms are sufficient to achieve satisfying accuracy, their computation requirements are overkill for most of the cases faced by the algorithm. For example, *the* has a unique POS tag that never changes depending on its position in the sentence. Moreover, *more* and *that* are frequent enough in the English text that there is a need to cache their extracted features.

4.1 Algorithm

SpeedRead takes advantage of the previous observations and tries to distinguish between ambiguous and certain words. To understand such influences, we ran a Stanford POS tagger (left 3 words Model (L3W); trained on Wall Street Journal(WSJ), Sections 1-18) over a 1 GiB of news text to calculate the following dictionaries:

- The most frequent POS tag of each token (Uni).
- The most frequent POS tag of each token, given the previous POS tag (Bi).
- The most frequent POS tag of each token, given the previous and next POS tags (Tri).

Using the above dictionaries to calculate the POS tag of a word, leads to various precision/recall scores. (Lee et al., 2011) shows that using sieves is the solution to combine several rules/dictionaries. In a sieve algorithm, there is a set of rules that are cascaded after each other. The algorithm runs the rules from the highest in precision to the lowest. The first rule, matching the problem instance, returns its computed tag immediately. SpeedRead

implements few sieves in the following order:

1. **Certain tokens**: Given a sentence, if the percentage frequency of the most frequent tag of a token is more than a threshold (in our work, 95%) then return that tag.
2. **Left and Right tags** (Tri): For each token with unknown tag, return the most frequent tag, given the left and right POS tags if they are known.
3. **Left tags** (Bi): For each token with unknown tag, return the most frequent tag, given the left POS tag if it is known from the previous stages.
4. **Token tag** (Uni) : For each token with unknown tag up to this stage, return the most frequent tag.
5. **Backoff tag**: If the token is unknown, use regular expression tagger to deduce the tag; the regular expression tagger relies heavily on matching suffixes.

4.2 Results

Table 5 shows the performance of different algorithms running on different sections of PTB. Stanford and SENNA models use sections 1-18, 19-21, 22-24 for training, development and testing datasets, respectively. Despite the simplicity of our algorithm, it achieves relatively high accuracy on the various datasets available.

Applying more context-aware rules, SpeedRead with sieves 1-5 (SR[Tri/Bi/Uni]) implemented, shows improvement in accuracy by around 2.85% compared to just using unigrams, SpeedRead with sieves 1,4-5 (SR[Uni]). To be sure that our algorithm is robust enough and not overfitting the dataset, we calculated the dictionaries again by running SENNA POS tagger(Collobert et al., 2011) over Reuters RCV1 corpus and the results were similar.

POS Tagger \ Sections	19-21	22-24	1-24
Stanford Bidirectional	97.27	97.32	98.16
Stanford L3W	96.97	96.89	97.90
SENNA	97.81	96.99	97.68
SR[Tri/Bi/Uni]	96.73	**96.39**	96.66
SR[Bi/Uni]	96.06	95.82	96.03
SR[Uni]	93.73	93.56	93.70

Table 5: Accuracy of different taggers on different sections of Penn Treebank. The first column corresponds to the development set and the second to the testing set.

Tables 5 and 6 show the tradeoff between accuracy and speed. Stanford pipeline offers two models with different speeds and accuracies. Since Left 3 Words model (L3W) is the preferred tagger to use in practice, we chose it to be our reference in terms of speed. L3W model runs 18 times faster than the state-of-art Bidirectional model and is only 0.4% less accurate. SpeedRead pushes the speed by another factor of 11 with only 0.5% drop in accuracy. Since the speed of some algorithms vary with the memory used, every algorithm was given enough memory that adding more memory will not affect its speed. The memory footprint is reported in the fourth column of Table 6.

POS Tagger	Speed Token/Sec	Relative Speed	Memory in MiB
Stanford Bi	1389	0.04	900
Stanford L3W	28,646	1.00	450
SENNA	34,385	1.20	150
SR [Tri/Bi/Uni]	318,368	**11.11**	600
SR [Bi/Uni]	397,501	13.87	250
SR [Uni]	564,977	19.72	120

Table 6: Speed of different POS taggers. The first two taggers are Stanford taggers. The first tagger runs the Bidirectional(Bi) model and the second runs the Left 3 Words (L3w) model. SpeedRead has three variations

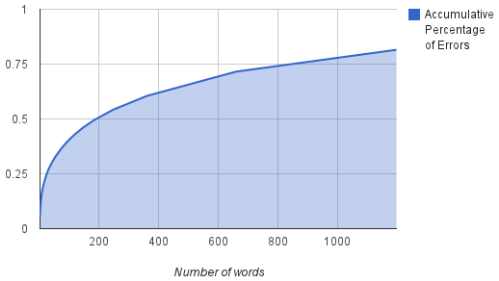


Figure 3: Accumulative percentage of errors made by the most frequent mistagged words. The total number of words is around 2000, the graph lists only the most frequent 1000.

4.3 Error Analysis

The most common errors are functional words, such as *that, more, ..* which have multiple roles in speech. This confirms some of the conclusions reported by (Manning, 2011). Figure 3 shows that less than 10% of mistagged words are responsible for slightly more than 50% of the errors. Regarding unknown words, the only part of the tagger that generalizes over unseen tokens is the regular expression tagger. Regular expressions are not extensive enough to achieve high accuracy. Therefore, we are planning to implement another backoff phase for the frequent unseen words where we accumulate the sentences, containing these words, after sufficient amount of text is processed and then run Stanford/SENNA tagger over those sentences to calculate the most common tag.

Table 7 shows the confusion matrix of the most ambiguous tags; the less ambiguous tags are clustered into one category, `0`. One of the biggest sources of confusion in tagging is between adjectives (JJ) and nouns (NN). Proper nouns are the second source of errors as most of the capitalized words will be mistakenly tagged as proper nouns while they are either adjectives or nouns. Such errors are the result of the weak logic implemented in the backoff tagger in SpeedRead, where regular expressions are applied in sequence returning the first match. Other types of errors are adverbs (RB) and propositions (IN). These errors are mainly because of the

Ref \ Test	DT	IN	JJ	NN	NNP	NNPS	NNS	RB	VBD
DT	11094	62	3	7	3	0	0	1	0
IN	15	13329	9	1	0	0	0	88	0
JJ	1	11	7461	**257**	130	2	10	65	38
NN	1	5	**288**	17196	111	0	18	11	2
NNP	8	13	118	109	12585	264	31	8	0
NNPS	0	0	0	0	70	81	16	0	0
NNS	0	0	1	23	20	42	7922	0	0
RB	17	**281**	103	23	8	0	0	3892	0
VBD	0	0	8	5	4	0	0	0	4311
VBG	0	0	25	104	5	0	0	0	0
O	26	163	154	172	47	4	107	67	174

Table 7: Confusion Matrix of the POS tags assigned by SpeedRead over the words of sections 22-24 of PTB. O represents all the other not mentioned tags.

ambiguity of the functional words. Functional words need deeper understanding of discourse, semantic and syntactic nature of the text. Taking into consideration the contexts around the words improves the accuracy of tagging. However, trigrams are still small to be considered sufficient context for resolving all the ambiguities.

5 Named Entity Recognition (NER)

Named entity recognition is essential to understand and extract information from text. Many efforts and several shared tasks, aiming to improve named entity recognition and classification, had been made; CONLL 2000/2003 (Tjong Kim Sang and De Meulder, 2003) are some of the shared tasks that addressed the named entity recognition task. We use CONLL 2003's definition of named entity recognition and classification task. CONLL 2003 defines the chunk borders of an entity by using IOB tags, where `I-TYPE` means that the word is inside an entity, `B-TYPE` means a beginning of a new entity if the previous token is part of an entity of the same type and `O` for anything that is not part of an entity. For classification, the task defines four different types: Person(PER), Organization(ORG), Location(LOC) and Miscellaneous(MISC) (See Figure 4).

We split the task into two phases. The first is to detect the borders of the entity phrase. After the entity chunk is detected, the second phase will classify each entity phrase to either a Person, Location, Organization or Miscellaneous.

Columbia/ORG *is an* **American**/Misc *university located in* **New**/LOC **York**/LOC.

Figure 4: Annotated text after NER.

5.1 Chunking

We rely on the POS tags of the phrase words to detect the phrase that constitute an entity. A word is considered to be a part of an entity: (1) if it is a demonym (our compiled list contains 320 nationalities), (2) if one of the following conjunction words {`&`, `de`, `of`} appearing in middle of an entity phrase or, (3) if its POS tag is `NNP(S)` except if it belongs to one of these sets:

- Week days and months and their abbreviations.
- Sports (our compiled list contains 182 names).
- Job and profession titles (our compiled list contains 314 title).
- Single Capital letters.

These sets are compiled using freebase.

CONLL dataset shows a strong correlation between POS tags `NNP(S)` and the words that are part of entities' phrases; 86% of the words that appear in entities' phrases have `NNP(S)` POS tags. The remaining words are distributed among different POS tags; 6.3% are demonyms. Adding the demonyms and proper nouns guarantee 92.3% coverage of the entities' words that appear in the dataset.

Using POS tags as main criteria to detect the entity phrases is expected, given the importance of the POS tags for the NER task. 14 out of 16 submitted paper to CONLL 2003 used POS tags as part of their feature set.

The behavior of the chunking algorithm is greedy as it tries to concatenate as many consecutive words as possible into one entity phrase. A technical issue appears in detecting the borders of phrases when multiple entities appear after each other without non-entity separator. This situation can be divided into two cases. Firstly, if the two consecutive entities are of the same type. In this case, the chunking tag should be `B-TYPE`. Looking at the dataset, such tag appears less than 0.2% out of all the entities' tags. For example, in the original Stanford MEMM implementation, the classifier (Klein et al., 2003) generates `IOB` chunking tags while in the later CRF models (Finkel et al., 2005) only `IO` chunking tags are generated. The second case is when the phrases are of different types. In the dataset, this case appears 248 times over 34834 entities. Since both cases are not frequent enough to harm the performance of the classifiers, SpeedRead does not recognize them.

5.1.1 Results

Table 8 shows F1 score of the chunking phase using different taggers to generate the POS tags. This score is calculated over the chunking tags of the words. `I` and `B` tags are considered as one class while `O` is left as it is. It is clear from Table 8 that using better POS taggers does not necessarily produce better results. The quality of SpeedRead POS tagging is sufficient for the chunking stage. SENNA and SpeedRead POS taggers work better for the detection phase because they are more aggressive, assigning the `NNP` tag to any capitalized word. On the other hand, Stanford tagger prefers to assign the tag of the lowered case shape of the word, if it is a common word.

Phase \ Dataset	Train	Dev	Test
SR+SR POS	94.24	94.49	**93.12**
SR+Stanford POS L3W	92.98	93.37	92.05
SR+CONLL POS	90.88	90.82	89.43
SR+SENNA POS	94.73	95.07	**93.80**

Table 8: F1 scores of the chunking phase using different POS tags. F1 score is calculated over tokens and not entities.

5.1.2 Error Analysis

Table 9 shows the error cases that appears in the chunking phase. The most common class of errors in the chunking phase is titles, such as *{RESULTS, DIVISION, CONFERENCE, PTS, PCT}*. These words seem to confuse the POS tagger. Another source of confusion for the POS tagger is the words *{Women, Men}*; such words appear in the name of sports so they get assigned `NNP` tag. As expected, all numbers that are part of entities are not detected. Conjunction words are the second important class of errors. (Pawel and Robert, 2007) shows that conjunction words that appear in middle of entities' phrases are hard to detect and need special classification task. As most of *of* occurrences are part of entities and the converse is true for *and*, we decided to include the former and exclude the later.

Word	Percentage	Type of error
Titles	22.7%	Detected
Titles	4.9%	Missed
of	2.6%	Detected
96, 95, 1000 ...	2.6%	Missed
Men	1.3%	Detected
Women	1.3%	Detected
and	1.1%	Missed
central	1.1%	Detected

Table 9: Most frequent errors in the chunking stage.

5.2 Classification

Classification is a harder problem than just detecting an entity. For example, "West Bank" can belong to two classes, location and organization. Disambiguating the sense of an entity depends on the context. For instance, "Mr. Green" indicates that "Green" is a person, while "around Green" points to a location. To classify an entity, we used a logistic regression classifier, sklearn (Scikit, 2011). The features we feed to the classifier are two factors per type: $\phi_{ij}(Type_i, phrase_j)$ and $\psi_{ij}(Type_i, context_j)$. Context consists of two words that precede and follow an entity phrase. To calculate these factors:

$$\phi_{ij}(Type_i, phrase) = \prod_{k}^{n} P(Type_i|w_k) \tag{1}$$

$$\psi_{ij}(Type_i, context = \{w_{before}, w_{after}\}) = P(Type_i|w_{before}) \times P(Type_i|w_{after}) \tag{2}$$

The conditional probabilities of the types, given a specific word, are calculated using the distribution of tags frequencies over words, retrieved from the annotated Reuters RCV1 corpus. SENNA NER tagger has been used to annotate the corpus.

Table 10 indicates the importance of the classification phase. First row shows that, given chunked input, the classification phase is able to achieve close scores to the state-of- art classifiers. However, given the chunks generated by SpeedRead, the scores drop around 9.5% in F1 scores.

Phase \ Dataset	Training	Dev	Test
SR+Gold Chunks	90.80	91.98	87.87
SpeeRead	82.05	83.35	78.28
Stanford	99.28	92.98	89.03
SENNA	96.75	97.24	89.58

Table 10: F1 scores calculated using conlleval.pl script for NER taggers. The table shows that SpeedRead F1 score is 10% below the sate-of-art achieved by SENNA.

To analyze the scores of the classification phase further, Table 11 shows a confusion matrix over the tags generated by SpeedRead. The errors that involve `O` are signs of chunking errors; there are 1158 chunking errors which exceed the total number of classification errors, 849.

Ref \ Test	LOC	MISC	ORG	PER	O
LOC	1737	34	95	36	23
MISC	36	660	57	52	113
ORG	**323**	73	1954	37	109
PER	26	8	72	2632	35
O	66	248	**412**	152	37445

Table 11: Confusion matrix of the SpeadRead NER tags over the CONLL test dataset tokens.

The chunking errors contain more false positives than false negatives. The chunking algorithm is aggressive in considering every `NNP(S)` as part of an entity. That would be fine if we had a perfect POS tagger. The reality that the POS tagger has hard time classifying uppercased words in titles and camel cased words that appear at the beginning of the sentence.

Once non-entity is considered part of an entity phrase, the classifier has higher chance of classifying it as an *ORG* than any other tag. The names of the organizations contain a mix of locations and persons' names, forcing the classifier to consider any long or mix of words as an organization entity. That appears more clearly in the second most frequent category of errors. 323 words in organizations entities' names were classified as locations. This could be explained by the fact that many companies and banks name themselves after country names and their locations. For example, "Bank of England" could be classified as a location because of the strong association between England and the tag location.

Table 12 shows that Stanford pipeline has a high cost for the accuracy achieved by the classifier. SENNA achieves close accuracy with twice the speed and less memory usage. SpeedRead takes another approach by focusing on speed. We are able to speed up the pipeline to the factor of 13. SpeedRead's memory footprint is half the memory consumed by the Stanford pipeline. Even though SpeedRead's accuracy is not close to the state-of-art, it still achieves 18% increase over the CONLL 2003 baseline. Moreover, adapting the pipeline to new domains could be easily done by integrating other knowledge base sources as freebase or Wikipedia. SENNA and SpeedRead are able to calculate POS tags at the end of the NER phase without extra computation while that is not true of Stanford pipeline standalone NER application. Using Stanford corenlp pipeline does not guarantee better execution time.

NER Tagger	Token/Sec	Relative Speed	Memory MiB
Stanford	11,612	1.00	1900
SENNA	18,579	2.13	150
SpeedRead	153,194	**13.9**	950

Table 12: Speed of different NER taggers. SpeedRead is faster by 13.9 times using half the memory consumed by Stanford.

Conclusion and Future Work

Our success in implementing a high performance tokenizer and POS tagger shows that it is possible to use simple algorithms and conditional probabilities, accumulated from a large corpora, to achieve good classification and chunking accuracies.

This could lead to a general technique of approximating any sequence tagging problem using sufficiently large dictionaries of conditional probabilities of contexts and inputs. This approximation has the advantage of speeding up the calculations and opens the horizon for new applications where scalability matters.

Expanding this approach to other languages depends on the availability of other high accurate taggers in these languages. We are looking to infer these conditional probabilities from a global knowledge base as freebase or the interlinking graph of Wikipedia.

SpeedRead is available under GPLv3 license and it is available to download from `www.textmap.org/speedread`. We anticipate that it will be useful to large spectrum of named entity recognition applications.

References

Bird, S., Klein, E., and Loper, E. (2009). *Natural Language Processing with Python*. O'Reilly Media.

Collobert, R. (2011). Deep learning for efficient discriminative parsing. In *International Conference on Artificial Intelligence and Statistics*.

Collobert, R. and Weston, J. (2008). A unified architecture for natural language processing: deep neural networks with multitask learning. In *Proceedings of the 25th international conference on Machine learning*, ICML '08, pages 160–167, New York, NY, USA. ACM.

Collobert, R., Weston, J., Bottou, L., Karlen, M., Kavukcuoglu, K., and Kuksa, P. (2011). Natural language processing (almost) from scratch. *Journal of Machine Learning Research*, 12:2493–2537.

Ferragina, P. and Scaiella, U. (2010). Tagme: on-the-fly annotation of short text fragments (by wikipedia entities). In *Proceedings of the 19th ACM international conference on Information and knowledge management*, CIKM '10, pages 1625–1628, New York, NY, USA. ACM.

Finkel, J. R., Grenager, T., and Manning, C. (2005). Incorporating non-local information into information extraction systems by gibbs sampling. In *In ACL*, pages 363–370.

Gillick, D. (2009). Sentence boundary detection and the problem with the u.s. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics, Companion Volume: Short Papers*, NAACL-Short '09, pages 241–244, Stroudsburg, PA, USA. Association for Computational Linguistics.

Gompel, M. V. (2012). Ucto: Unicode tokenizer. `http://ilk.uvt.nl/ucto`.

Klein, D., Smarr, J., Nguyen, H., and Manning, C. D. (2003). Named entity recognition with character-level models. In Daelemans, W. and Osborne, M., editors, *Proceedings of CoNLL-2003*, pages 180–183. Edmonton, Canada.

Lee, H., Peirsman, Y., Chang, A., Chambers, N., Surdeanu, M., and Jurafsky, D. (2011). Stanford's multi-pass sieve coreference resolution system at the conll-2011 shared task. In *Proceedings of the CoNLL-2011 Shared Task*.

Manning, C. D. (2011). Part-of-speech tagging from 97% to 100%: Is it time for some linguistics? In Gelbukh, A. F., editor, *CICLing (1)*, volume 6608 of *Lecture Notes in Computer Science*, pages 171–189. Springer.

Milne, D. and Witten, I. H. (2008). An effective, low-cost measure of semantic relatedness obtained from wikipedia links. In *In Proceedings of AAAI 2008*.

Nadeau, D. and Sekine, S. (2007). A survey of named entity recognition and classification. *Lingvisticae Investigationes*, 30(1):3–26.

Pauls, A. and Klein, D. (2011). Faster and smaller n-gram language models. In Lin, D., Matsumoto, Y., and Mihalcea, R., editors, *ACL*, pages 258–267. The Association for Computer Linguistics.

Pawel, M. and Robert, D. (2007). Handling conjunctions in named entities. *Lingvisticae Investigationes*, 30(1):49–68.

Schafer, F.-R. (2012). Quex - fast universal lexical analyzer generator. `http://quex.sourceforge.net`.

Scikit, S.-l. D. (2011). Scikit-learn: Machine learning in python. *Journal of Machine Learning Research*, 12:2825–2830.

Tjong Kim Sang, E. F. and De Meulder, F. (2003). Introduction to the conll-2003 shared task: Language-independent named entity recognition. In Daelemans, W. and Osborne, M., editors, *Proceedings of CoNLL-2003*, pages 142–147. Edmonton, Canada.

Toutanova, K., Klein, D., Manning, C. D., and Singer, Y. (2003). Feature-rich part-of-speech tagging with a cyclic dependency network. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology - Volume 1*, NAACL '03, pages 173–180, Stroudsburg, PA, USA. Association for Computational Linguistics.

Toutanova, K. and Manning, C. D. (2000). Enriching the knowledge sources used in a maximum entropy part-of-speech tagger. In *Proceedings of the 2000 Joint SIGDAT conference on Empirical methods in natural language processing and very large corpora: held in conjunction with the 38th Annual Meeting of the Association for Computational Linguistics - Volume 13*, EMNLP '00, pages 63–70, Stroudsburg, PA, USA. Association for Computational Linguistics.

Webster, J. J. and Kit, C. (1992). Tokenization as the initial phase in nlp. In *Proceedings of the 14th conference on Computational linguistics - Volume 4*, COLING '92, pages 1106–1110, Stroudsburg, PA, USA. Association for Computational Linguistics.

Experiments with Term Translation

Mihael ARCAN[1] *Christian FEDERMANN*[2] *Paul BUITELAAR*[1]
(1) Unit for Natural Language Processing, Digital Enterprise Research Institute
National University of Ireland Galway
Galway, Ireland
(2) Language Technology Lab, German Research Center for AI
Saarbrücken, Germany
mihael.arcan@deri.org, paul.buitelaer@deri.org,
cfedermann@dfki.de

Abstract

In this article we investigate the translation of financial terms from English into German in the isolation of an ontology vocabulary. For this study we automatically built new domain-specific resources from the translation search engine Linguee and from the online encyclopaedia Wikipedia. Due to the fact that we performed the translation approach on a monolingual ontology, we ran several sub-experiments to find the most appropriate model to translate the financial vocabulary. The findings from these experiments lead to the conclusion that a hybrid translation system, a combination of bilingual terminological resources and statistical machine translation, can help to improve translation of domain-specific terms. Finally we undertook a manual cross-lingual evaluation on the monolingual ontology to get a better understanding on this specific short text translation task.

Keywords: Ontologies and terminology, Empirical machine translation.

Proceedings of COLING 2012: Technical Papers, pages 67–82,
COLING 2012, Mumbai, December 2012.

1 Introduction

Our research on the translation of ontology vocabularies is motivated by the challenge of translating domain-specific terms with restricted or no additional textual context that in other cases may be used to improve the translation. For our experiment we started by translating financial terms with the baseline systems trained on the JRC-Acquis (Steinberger et al., 2006) corpus and the European Central Bank Corpus (Tiedemann, 2009). Although both resources contain a large amount of parallel data, the translations were not satisfactory. To improve the translations of the financial ontology vocabulary we built a new parallel resource, which was generated using Linguee, an online translation query service. With this data, we could train a small model, which produced better translations than the baseline model using only general resources.

Since the manual development of terminological resources is a time intensive and expensive task, we used Wikipedia as a background knowledge base and examined the articles tagged with domain-specific categories. With this extracted domain-specific data we built a specialised English-German lexicon to store translations of domain-specific terms. These terms were then used in a pre-processing method in the decoding approach. This approach incorporates the work by (Aggarwal et al., 2011), where the authors use the ontology structure to calculate the similarity between the labels. They combine the semantic, terminological and linguistic information for monolingual ontology matching, which can be extended to the multilingual scenario. We split the financial terms into n-grams and queried for financial sub-terms in Wikipedia, which we used to query Wikipedia.

The remainder of the paper is organised as follows: In Section 2 we give an overview on the related work. In Section 3 we describe the ontology and the existing parallel resources, which were used for generating the translation and language model. Section 4 presents the new resources which were used for improving the term translation. Furthermore we discuss the results of exploiting the different resources. We conclude with a summary and give an outlook on future work.

2 Related Work

The related research focusses on different aspects relevant to our work: domain-specific term translation. Firstly we have to understand the structure of these specific terms and the variations which come when dealing with these terms. Kerremans (2010) discusses in detail the issue of terminological variation in the context of specialised translation on a parallel corpus of biodiversity texts. He shows that a term often cannot be aligned to any term in the target language. As a result, he proposes that specialised translation dictionaries should store different translation possibilities or term variants. In addition to that, Weller et al. (2011) describe methods for terminology extraction and bilingual term alignment from comparable corpora. In their compound translation task, they use a dictionary to avoid out-of-domain translation. In contrast, to address this problem, which frequently arises in domain-specific translation we decided to generate our own customised lexicon; which we constructed from the multilingual Wikipedia and its dense inter-article link structure.

Erdmann et al. (2008) also extracted terms from Wikipedia articles; however, they assumed that two articles connected by an Interlanguage link are likely to have the same content and thus an equivalent title. We likewise build a lexicon from Wikipedia, but instead of collecting all of the titles from Wikipedia, we target only the domain-specific titles and their translated equivalents. Vivaldi and Rodriguez (2010) proposed a methodology for term extraction in the biomedical domain with the help of Wikipedia. As a starting point, they manually selected a set of seed words for a domain, which were then used to find the corresponding nodes in this resource. For cleaning their collected data, they used thresholds to avoid storing undesirable categories. Müller and Gurevych (2008) used

Wikipedia and Wiktionary as knowledge bases to integrate semantic knowledge into Information Retrieval. Their models, text semantic relatedness (for Wikipedia) and word semantic relatedness (for Wiktionary), are compared to a statistical model implemented in Lucene. In their approach to bilingual retrieval, they use the cross-language links in Wikipedia, which improved the retrieval performance in their experiment, especially when the machine translation system generated incorrect translations. Zesch et al. (2008) address the issues in accessing the largest collaborative resources: Wikipedia and Wiktionary. They describe several modules and APIs for converting a Wikipedia XML Dump into a more suitable format. Instead of parsing the large Wikipedia XML Dump, they suggest to store the Dump into a database, which significantly increases the performance in retrieval time of queries.

3 Experimental Data

We are investigating the problem of translating a domain-specific vocabulary, therefore our experiments started with an analysis of the financial terms stored in the investigated ontology. With these extracted terms we built different multilingual resources, which were used for financial term translation. Firstly, we used the encyclopaedia Wikipedia, where we extracted the titles from domain-specific Wikipedia articles. Secondly, we used the same financial labels to build a parallel resource for the financial domain. For this approach we used the Linguee Web service.

In this section, we present several types of data. Section 3.1 gives an overview of the data that was used in translation. In Sections 3.2 and 3.3 we describe existing multilingual resources, which were used to train the translation and language model. For our current research we used JRC-Acquis and the European Central Bank (ECB) corpus, respectively. In the end we describe the procedure to obtain domains-specific resources by Linguee 3.4 and Wikipedia 3.5.

3.1 The Financial Ontology

For our study we used the UK GAAP[1] financial ontology, prepared by the XBRL[2] European Business Registers (xEBR) Working Group. This financial ontology is a framework for describing financial accounting and profile information of business entities across Europe; see also Declerck et al. (2010). The ontology holds 142 concepts and is partially aligned into German, Dutch, Spanish, French and Italian. We identified only 16 English financial terms and their German equivalents, which were used as reference translations for automatic evaluation.

The financial terms are not really terms from a linguistic point of view, but they are used in financial or accounting reports as unique financial expressions or tags to organize and retrieve automatically reported information. Therefore it is important to translate these financial terms exactly. Table 1 illustrates the structure of xEBR terms.

It is obvious that they are not comparable to general language, but instead are more like headlines in newspapers, which are often short, very informative, and written in a telegraphic style. xEBR terms are often only noun phrases without any determiner. The length of the financial terms varies, e.g. the longest financial term considered for translation has a length of 11 tokens, while others may consist of 1 or 2 (Figure 1).

[1] GAAP - Generally Accepted Accounting Practice
[2] XBRL - eXtensible Business Reporting Language, `http://www.xbrl.org/`

Term Length	Term Examples
11	Taxes Remuneration And Social Security Payable After More Than One Year
10	Amounts Owed To Credit Institutions After More Than One Year ...
	...
2	Net Turnover, Liquid Assets, Income Taxes, Financial Charges ...
1	Assets, Capital, Equity, Securities, Charges, Balance, Capital, Reserves ...

Table 1: Examples for financial labels in the UK GAAP

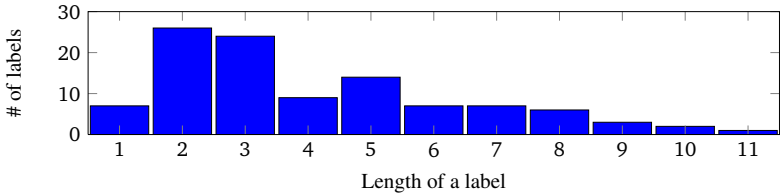


Figure 1: Label length of the UK GAAP ontology

3.2 JRC-Acquis

The general parallel corpus JRC-Acquis[3] was used as baseline training data. This corpus is available in almost every EU official language (except Irish), and is a collection of legislative texts written between 1950 and now.

Although previous research showed, that a training model built by using a general resource cannot be used to translate domain-specific terms (Wu et al., 2008), we decided to evaluate the translations on these resources to illustrate any improvement steps from a general resource to specialised domain resources.

3.3 European Central Bank Corpus

For comparison with JRC-Acquis, we also did experiments using the European Central Bank Corpus[4], which contains a financial vocabulary. The multilingual corpus is generated by extracting the website and documentation from the European Central Bank and is aligned among 19 European languages. For our research we used the English-German language pair, which consists of 113,171 sentence pairs or 2.8 million English and 2.5 million German tokens.

3.4 Linguee - Dictionary and Translation Search Engine

Alongside these existing resources, we built a new parallel resource based on the ontology vocabulary that we want to translate. Therefore we used Linguee,[5] a combination of a dictionary and a search engine, which indexes around 100 million bilingual texts on words and expressions. The search results show example sentences that depict how the searched expression has been translated in context. The bilingual dataset was gathered from the web, particularly from multilingual websites of companies, organisations or universities. Other sources include EU documents and patent

[3] http://langtech.jrc.it/JRC-Acquis.html
[4] http://opus.lingfil.uu.se/ECB.php
[5] http://www.linguee.com/

specifications. Since Linguee includes EU documents, they also use parallel sentences from JRC-Acquis, whereby the proportion of sentences returned by Linguee is very low, only 131 sentences or 0.54% overlap with the corpus.

In contrast to translation engines like Google Translate and Bing Translator, which give you the most probable translation of a source text, every entry in the Linguee database was translated manually.

Domain-specific parallel corpus generation

To build a new training model that is specialised for our xEBR ontology, we used the Linguee search engine. This resource can be queried on single words and on word expressions with or without quotation marks. We stored the HTML output of the Linguee queries of our financial terms and parsed these files to extract plain parallel text. From this, we built a financial parallel corpus with 24,247 translation pairs, including single words, multi-word expressions and sentences (Table 2). The English part of the parallel resource contained 1,032,676 tokens and the German part 865,460.

Single terms	Enterprise, share, reserve, debtor, expenses, ...
Multi-words	at a specific amount, credit institute, in the amount of, doubled over the last year
Sentences	Finally, the European Parliament called for social and cultural aspects of immigration to receive equal treatment than economic and security aspects of the issue.

Table 2: Examples of extracted text from the translation search engine Linguee

3.5 Wikipedia

Wikipedia[6] is a multilingual, freely available encyclopaedia that was built by a collaborative effort of voluntary contributors. All combined Wikipedias hold approximately 19 million articles or more than 8 billion words in more than 270 languages, making it the largest collection of freely available knowledge.[7]

With the heavily interlinked information base, Wikipedia forms a rich lexical and semantic resource. Besides a large number of articles, it also holds a hierarchy of categories that Wikipedia articles are tagged with. It includes knowledge about named entities, domain-specific terms and word senses. Furthermore, the redirect system of Wikipedia articles can be used as a dictionary for synonyms, spelling variations and abbreviations.

Domain-specific lexicon generation

To improve translations, based on the domain-specific parallel corpus, we built a cross-lingual terminological lexicon. From the Wikipedia articles we used different information units: the title, the category (or categories) of the title and the internal Interwiki\Interlanguage links of the title. The concept of Interwiki links can be used to make links to other Wikipedia articles in the same language or to another Wikipedia language i.e. Interlanguage links. The domain-specific lexicon was generated by two approaches:

a) domain detection of the ontology (bottom-up approach);

b) extraction of cross-lingual terminology (top-down approach).

[6] http://www.wikipedia.org

[7] http://en.wikipedia.org/wiki/Wikipedia:Size_comparison

In our first approach, we used Wikipedia to determine the domain (or several domains) of the ontology. The bottom-up approach (a) is to represent this domain by the most frequent categories associated with the vocabulary we want to translate. For this approach, the financial terms, which were extracted from the ontology, were used to query the Wikipedia knowledge base.[8] Initially a Wikipedia article was considered for further examination if its title is equivalent to our financial terms. In this first step, 7 terms from our ontology were identified in the Wikipedia knowledge base, i.e.:

Income tax, Earnings before interest and taxes, Asset, Stocks, Debtor, Gross profit, Income

We then collected the categories of the articles associated with these titles. Since a category can appear with different financial term, we also stored the frequency of these categories.[9] In a second round, we split our financial terms into all possible n-grams and repeated the query again to find additional categories based on the split n-grams. Table 3 shows the collected categories of the first approach and how often they appeared with respect to the extracted financial terms.

Collected Wikipedia Categories	
Frequency	Name
8	Generally Accepted Accounting Principles
4	Debt
	...
1	Political science terms
1	Physical punishments

Table 3: Collected Wikipedia Categories based on the extracted financial terms

After storing all categories, the only categories considered were the ones that had a frequency value more than the calculated arithmetic mean of all the frequencies (> 3.15). For the calculation of the arithmetic mean only the categories that had a frequency larger than 1 were considered, since 2,262 of 3,615 collected categories (62.6%) had a frequency of 1. Using this threshold we avoided the extraction of a vocabulary that is not related to the ontology. Without this threshold, out-of-domain categories would be stored, which would extend the lexicon with vocabulary that would not benefit the ontology translation, e.g. *Physical punishments*, which was a category associated with the financial term *Stocks*.

In the next step, we further extended the list of the previous collected categories with the use of full and split terms. This was done by storing new categories based on the Wikipedia Interwiki links of each article which was tagged with a category from Table 3. For example, we collected all categories of the Wikipedia article *Balance sheet*.[10] In addition to that, we examined all Interwiki links of the article *Balance sheet* and also stored the categories of articles which have an incoming link from this article.[11] For example, we stored all categories of the 106 articles which are linked with the article *Balance sheet*. The frequencies of these categories were summed up again to re-calculate the geometric mean. Finally a new category was added to the final category list, if the new category frequency exceeds the arithmetic mean threshold (> 18.40).

[8] For the Wikipedia Query we used the Wikipedia XML dump; `enwiki-20120702-pages-articles`

[9] The Wikipedia titles *Operating Income*, *Income*, *Gross profit*, *Income statement*, *Debtor* ... are tagged with the category *Generally Accepted Accounting Principles*

[10] *Financial statements*, *Accounting terminology*

[11] *Balance sheet*

Final Category List	
Frequency	Name
95	Economics terminology
62	Generally Accepted Accounting Principles
61	Macroeconomics
	...

Table 4: Most frequent Categories based on the xEBR terms and their Interwiki links

The final category list contained 33 financial Wikipedia categories (Table 4), which were used to extract the financial terms and their translations.

With the final list of categories, we started an investigation of all Wikipedia articles tagged with these financial categories. Each Wikipedia title was considered as a useful domain-specific term and was stored in our lexicon if a German title in the Wikipedia knowledge base also existed. As an example, we examined the category *Accounting terminology* and stored the English Wikipedia title *Balance sheet* with the German equivalent Wikipedia title *Bilanz*.

At the end of the lexicon generation we examined 5,228 Wikipedia articles that were tagged with one or more financial categories. From this set of articles we were able to generate a terminological lexicon with 3,228 English-German entities. The difference between the number of examined titles and the lexicon items is attributed to the fact that not all English Wikipedia titles are linked to a German one. These translation pairs were used to suggest the SMT system to choose the extracted translations by annotating the decoder input using the XML input markup scheme.

4 Experiments and Evaluation

Since the UK GAAP is a monolingual ontology, it holds no reference translation needed for automatic evaluation. Therefore we performed several experiments to find the best approach to translate this financial ontology. For decoding, we used the Moses Toolkit, with its standard settings (Section 4.1). If reference translations were available, we undertook an automatic evaluation using the BLEU (Papineni et al., 2002), NIST (Doddington, 2002), TER (Snover et al., 2006), and Meteor[12] (Denkowski and Lavie, 2011) algorithms.

With the first evaluation experiment we translated 16 aligned English-German labels with different translation models (Section 4.2). Furthermore, we translated the bilingual German GAAP to see which translation model performs best regarding the 2794 financial labels that are stored in this ontology (Section 4.3). We also compared the perplexity between several language models and the vocabulary stored in the UK GAAP ontology (Section 4.4). Finally we applied the best translation model to the monolingual ontology and undertook a manual, cross-lingual evaluation with six annotators (Section 4.5).

4.1 Translation System: Moses Toolkit

For generating the translations from English into German, we used the statistical translation toolkit Moses (Koehn et al., 2007). Furthermore, we aimed to improve the translations only on the surface level, and therefore no part-of-speech information was taken into account. Word and phrase alignments were built with the GIZA++ toolkit (Och and Ney, 2003), where the 5-gram language

[12]Meteor configuration: exact, stem, paraphrase

model was built by SRILM with Kneser-Ney smoothing (Stolcke, 2002).

4.2 Translating aligned UK – German GAAP labels

The UK GAAP is a monolingual ontology which holds 142 financial labels. With the help of the German equivalent, i.e. German GAAP, we aligned 16 German labels with the English ones, stored in the UK GAAP. This allowed us to do a small automatic evaluation, regardless of the low number of labels to be translated.

		Scoring Metric				
Source	# correct	BLEU-2	BLEU-4	NIST	TER	Meteor
JRC-Acquis	3	0.2629	0.2747	1.8112	0.6969	0.1579
ECB	3	0.2572	0.2725	1.5282	0.7878	0.1707
Linguee+Wikipedia	5	0.3623	0.2922	2.3259	0.6363	0.4085

Table 5: Evaluation scores for aligned UK–German GAAP translations

Despite the small amount of translations Table 5 shows the Linguee + Wikipedia resource produces the best BLEU score.

#	Source Label	Linguee+Wikipedia Model	Reference Translation
1	Fixed assets	Anlagevermögen	Anlagevermögen
2	Tangible fixed assets	Sachanlagen	Sachanlagen
3	Other tangible fixed assets	sonstige Sachanlagen	sonstige Sachanlagen
4	Equity	Eigenkapital	Eigenkapital
5	Income statement	Gewinn- und Verlustrechnung	Gewinn- und Verlustrechnung
6	Intangible fixed assets	immaterielle Vermögenswerte	Immaterielle Vermögensgegenstände
7	Other intangible fixed	sonstige immaterielle Vermögenswerte	sonstige immaterielle Vermögensgegenstände
8	Social security cost	Sozialbeiträge	soziale Abgaben
9	Other provisions	die sonstigen Rückstellungen	sonstige Rückstellungen
10	Other operating income	die sonstigen betrieblichen Erträge	sonstige betriebliche Erträge
11	Wages and salaries	die Löhne und Gehälter	Löhne und Gehälter
12	Current assets	kurzfristige Vermögenswerte	Umlaufvermögen
13	Work in progress	angefangene Arbeiten	unfertige Erzeugnisse
14	Work in progress	angefangene Arbeiten	unfertige Leistungen
15	Extraordinary income	das außerordentliche Ergebnis	außerordentliche Erträge
16	Equity and Liabilities	Eigenkapital und Zur	Bilanzsumme, Summe Passiva

Table 6: Results of financial translations generated by Linguee+Wikipedia translation model

Table 6 shows the translations of the 16 financial labels which were aligned between the UK and the German GAAP. The first part of the table, examples 1 to 5, represents the correct translations, which match exactly with the reference provided by the xEBR Working Group.

The next block represents translations which do not match completely with the reference translations. Examples 6 and 7 illustrate the problem of translating the label *fixed assets*[13] that can be translated into near synonyms *Vermögenswerte* or *Vermögensgegenstände*. Example 8 shows where the translation model generated a compound, but the reference translation consists of two separate tokens. If we de-compound the translation *Sozialbeträge* into *soziale Beträge*, we get a synonym to

[13] *Fixed assets* and *Other fixed assets*

the reference translation. Examples 9 to 11 represent translations with over-specification, since the ontology labels do not require the German article[14] at the beginning of a label.

The last part of the table illustrates incorrect translations. Examples 12 to 14 are translated into idiomatic expressions, whereby example 15 shows a wrong lexical choice. The word *Income* was translated into*Ergebnis*, whereas it should have been translated into *Erträge*. In example 16 a part of the source label, i.e. *Liabilities* is missed in the target translation.

4.3 Translating the German GAAP with different models

Since we built a financial parallel resource (see Section 3.4 and 3.5) and generated a translation model based on this financial vocabulary, we tested how well the model performs on a similar ontology. Therefore we translated the aforementioned German GAAP ontology, which holds 2,794 labels[15].

		Scoring Metric				
Source	# correct	BLEU-2	BLEU-4	NIST	TER	Meteor
JRC-Acquis	47	0.2276	0.1122	2.7022	0.9337	0.1761
ECB	24	0.1715	0.0596	2.1921	0.9834	0.1321
Linguee+Wikipedia	79	0.3397	0.2292	3.9383	0.8291	0.2917

Table 7: Evaluation scores for German GAAP term translations

Table 7 illustrates the automatic metrics used to evaluate the translation of the German GAAP, where the best BLEU results are generated by the Linguee+Wikipedia translation model. We can deduce from this experiment that even though JRC-Acquis has a larger number of tokens than the Linguee+Wikipedia corpus, it does not generate better translations of financial labels. The ECB corpus also does not generate better translations, although it is considered a domain-specific corpus.

4.4 Perplexity of different language models

The automatic evaluation with the small amount of translation and their references cannot demonstrate the quality of the translation model with regard to the whole UK GAAP ontology. Therefore we compared the perplexity[16] of different language models and the vocabulary of the UK GAAP ontology. Since a better language model should assign a higher probability to its test set, we tested which generated language model gives the highest probability on the UK GAAP vocabulary.

The perplexity (1) is a reformulation of cross-entropy (2).

$$PP = 2^{H(p_{LM})} \tag{1}$$

$$H(p_{LM}) = -\frac{1}{n}\sum_{i=1}^{n} log p_{LM}(w_i|, \ldots, w_{i-1}) \tag{2}$$

Table 8 illustrates that the ECB language model generates the worst perplexity on the UK GAAP vocabulary. On the other hand, the best probability is calculated by the Linguee+Wikipedia language

[14]German articles: die, der, das

[15]For comparison, the monolingual UK GAAP holds only 142 financial labels

[16]The perplexity was calculated with the SRILM ngram tool

model, which is not a surprise, since the resource is generated from the same vocabulary. Besides that, the best perplexity is generated by the German GAAP language model, which indicates that the vocabulary is most similar to the UK GAAP in comparison to other languages models.

	logprob	Perplexity
JRC-Acquis LM	-1,656.39	243.625
ECB LM	-1,871.33	497.098
German GAAP LM	-1,528.92	159.608
Linguee + Wikipedia LM	-1,277.15	69.226

Table 8: Perplexity of the language models

4.5 Manual Evaluation of Translation Quality - UK GAAP

We have undertaken a manual evaluation campaign to assess the translation quality of our terminology translation system, which was performed with the Appraise Toolkit.(Federmann, 2012)

In this section, we will a) describe the annotation setup and task presented to the human annotators, b) report on the translation quality achieved by the Linguee+Wikipedia approach, and c) present inter-annotator agreement scores that allow us to judge the reliability of the human rankings.

4.5.1 Annotation Setup

In order to manually assess the translation quality of the different systems under investigation, we designed a simple classification scheme consisting of three distinct classes:

1. *Acceptable (A)*: terms classified as acceptable are either fully identical to the reference term or semantically equivalent;
2. *Can easily be fixed (C)*: terms in this class require some minor correction (such as fixing of typos, removal of punctuation, etc.) but are nearly acceptable. The general semantics of the reference term are correctly conveyed to the reader.
3. *None of both (N)*: the translation of the term does not match the intended semantics or it is plain wrong. Items in this class are considered severe errors which cannot easily be fixed and hence should be avoided wherever possible.

4.5.2 Annotation Data

We set up an evaluation task containing 142 term translations and the corresponding source term. The set was then given to a total of six human annotators who classified the observed translation output according to the classification scheme described above. The human annotators were lay users without in-depth knowledge of the terms' domain.

In total, we collected 852 classification items from six annotators. Table 9 shows the results from the manual evaluation for term translations into German. We report the distribution of classes per evaluation task which are displayed in *best-to-worst* order.

	Classes		
System	A	C	N
Linguee+Wikipedia Model	59.15%	29.34%	11.50%

Table 9: Results from the manual evaluation for German

In order to better be able to interpret these rankings, we computed the inter-annotator agreement between human annotators. We report the scores generated with the following agreement metrics:

- S (Bennett et al., 1954);
- π (Scott, 1955);
- κ (Fleiss, 1971);
- α (Krippendorff, 2004).

Table 10 presents the aforementioned metrics' scores for German term translations.

	Agreement Metric			
System	S	π	κ	α
Linguee+Wikipedia Model	0.467	0.355	0.357	0.355

Table 10: Annotator agreement scores for German

Overall, we achieve an average κ score of 0.357, which can be interpreted as *fair agreement* following (Landis and Koch, 1977). Given the observed inter-annotator agreement, we expect the reported ranking results to be meaningful. The inclusion of domain experts into the manual evaluation campaign will be an interesting extension of the work presented.

4.6 Manual error analysis of UK GAAP

In addition to the manual evaluation we performed with six annotators on the UK GAAP ontology monolingual (Section 4.5), we also performed a closer analysis of each label.

In the first step, we extracted 36 labels from the manual evaluation campaign, where all evaluators annotated the translation as "Acceptable". Examples 1 to 7 (Table 11) depict a small set of the acceptable translations.

#	Source label	Target label
1	Equity	Eigenkapital
2	Stocks	Wertpapiere
3	Key Balance Sheet Figures	Bilanzkennzahlen
4	Revaluation Reserve	Neubewertungsrücklage
5	Interest And Similar Charges	Zinsen und ähnliche Aufwendungen
6	Debenture Loans After More Than One Year	Schuldscheindarlehen nach mehr als einem Jahr
7	Profit Or Loss On Ordinary Activities Before Taxes	Gewinn oder Verlust aus der gewöhnlichen Geschäftstätigkeit vor Steuern
8	Net Operating Income	Ergebnis aus der
9	Equity And Liabilities	Und Passiva
10	Profit Loss For The Period	Ergebnis der

Table 11: Translations which all annotators considered as "Acceptable" (1-7) and "None of both" (8-10)

We also extracted financial labels where all evaluators annotated the translations of the labels as "None of both", which indicates a low quality of the translations. These labels are shown in the

last part of Table 11, examples 8 to 10. The reason for the low quality of the translations is that the target label omits part of the source label. In example 8 we miss the translation for the segment *Net operating*, in 9 *Equity* is not translated and in example 10 *Loss for the period* is missing.

4.7 Interpretation of the evaluation time and the quality of translations

In addition to the evaluation of the quality of financial label translation, we also measured the evaluation time regarding different criteria, i.e. regarding the length of the label, the quality of the translation and the evaluation time for all labels.

Evaluation time regarding the length of the source labels

Figure 2 illustrates the evaluation time regarding the length of a source label. We learned that, on average, the evaluation time increased with the length of the source label, e.g. the evaluators spent more than 9 seconds to evaluate unigram label.[17] On the other hand, it took more than 26 seconds to evaluate the longest financial label.[18]

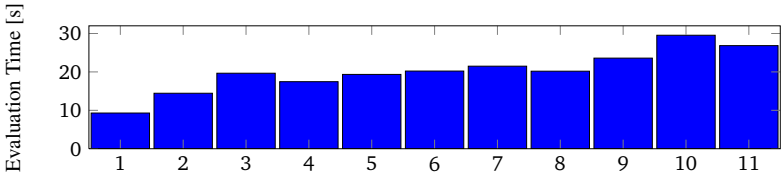


Figure 2: Evaluation time per length of the source labels

Evaluation time with respect to the quality of the translations

The evaluation task asked the evaluators to evaluate the translation quality based on three classes, "Acceptable", "Can easily be fixed" and "None of both" (cf. Section 4.5). To get a more fine-grained classification with a broader span of data, we gave each label a numeric value regarding the translation quality set by the six evaluators, e.g. the financial label *Charges* and its translation *Kosten* was annotated by all evaluators as "Acceptable"; analogously, the financial label *Financial Charges* and its translation *finanziellen Belastungen* was annotated by four evaluators with "Acceptable", whereas two evaluators annotated it as "Can easily be fixed". Since we know how each evaluator annotated a translation, we interpret the three evaluation classes into a numerical value evaluation score, i.e. if an translation was annotated with "Acceptable" we add the value 3 to the evaluation score, if it was annotated with "Can easily be fixed" we add 2, and if it was annotated with "None of both", we do not add any value to the score. With this reformulation, the financial label *Charges-Kosten* gets an evaluation score of 18,[19] and the *Financial Charges-finanziellen Belastungen* gets an evaluation score of 16.[20] With this additional classification we get a broader variety with 18 different quality classes, compared to the three classes set by the evaluators.

Figure 3 depicts the evaluation time regarding the translation quality of the financial labels. For

[17] *Assets*, *Reserves*, *Equity*, *Stocks* ...
[18] *Taxes Remuneration And Social Security Payable After More Than One Year*
[19] 3+3+3+3+3+3 = 18
[20] 3+3+3+3+2+2 = 16

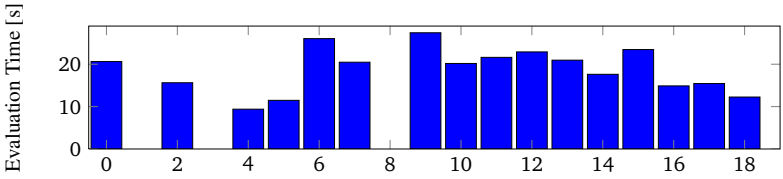


Figure 3: Evaluation time per quality of the translation

labels, which have an evaluation score of zero[21] the evaluation time is more than 20 seconds. The evaluation time decreases for labels with an evaluation score between two and five, but starts to increase when the evaluation score is equal six or more. For labels that have an evaluation score between six and thirteen, the evaluation time is higher than for labels with a lower or higher score. At the end the evaluation time decreases again. We can deduce from this experiment that it is easier to evaluate good and weak translations, but on the other hand it is harder to evaluate translations that do not belong to these two evaluation classes.

Evaluation time for the financial 142 labels

Figure 4 shows the evaluation time for all 142 labels stored in the UK GAAP ontology. We can see that the longest evaluation time to evaluate one term was more than 62 seconds, namely for the label *Operating Bach Ratios*. On the other hand, the fastest time to evaluate a label was less than 3 second for the label *Staff Costs* which was translated into *Personalkosten*.

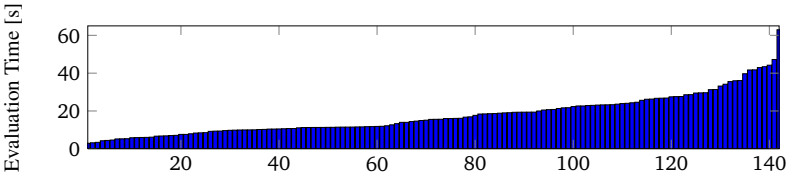


Figure 4: Evaluation time for the financial 142 labels

#	Source label	Time [s]
1	Staff Costs	2.946
2	Capital	3.177
3	Extraordinary Charges	3.421
⋮	⋮	⋮
140	Deferred Charges And Accrued Income	44.213
141	Depreciation On Intangible And Tangible Fixed Assets	47.211
142	Operating Bach Ratios	62.943

Table 12: Financial labels with the fastest (above) and the slowest (below) evaluation time

Table 12 shows the five fastest and slowest evaluation for the financial labels.

[21] *Net Operating Income*, *Equity And Liabilities*, *Profit Loss For The Period*

Conclusion and Future Work

We presented our work on the translation of a monolingual financial ontology. We performed smaller sub-experiments to determine the most appropriate translation model to translate financial labels in isolation. Hence we evaluated the translations on a small subset of aligned labels between different financial ontologies. Furthermore, we evaluated different translation models on a comparable ontology from the financial domain and compared the perplexity of the ontology to be translated with different resources. All these sub-experiments proved that the approach of building new, specific resources showed a large impact on the translation quality. Therefore, generating specialised resources for different specific domains will be the focus of our future work. On the one hand, building appropriate translation models is important, but our experiment also highlighted the importance of additional non-parallel resources, like Wikipedia, Wiktionary,[22] and DBpedia.[23] In addition to extracting Wikipedia articles with their multilingual equivalents, Wikipedia holds much more information in the articles themselves. Therefore, exploiting these non-parallel resources, as shown by (Fišer et al., 2011), would clearly help to improve the performance of the translation system. Future work needs to include the Wikipedia redirect system, which would allow a better understanding of the synonymy and spelling variations of specific terms.

In addition to exploiting new resources for statistical machine translation, the manual evaluation for monolingual resources needs to become the focus of our future work. The manual evaluation campaign was time consuming, but provided a closer look into the translation errors. It indicates that the evaluation classes for manual evaluation have to be reformulated into more fine-grained decisions. We learned that we may distinguish between translations with "one grammatical error" or "several grammatical errors". It might also be interesting to classify the types of grammatical error, e.g. number, gender or case, e.g. *Betriebsstoffen* vs. *Betriebsstoffe*. During the evaluation we also observed over-specification, where the translation into German *Die Forderungen ...*,[24] does not require the German article *die* at the beginning. Specifically to the German language we further observed some compound errors, e.g. *Ergebnis Verlust* should be merged into a compound expression. Another major issue were errors of omissions, where we miss some information from the source side, e.g. the translation *Und Passiva* omits the source part *Equity*. Further to the linguistic error classification, the type of the translation mismatch might be interesting to investigate, i.e. cultural, linguistic or domain-specific. Also it is important to know if a translation is too broad or too narrow. Especially for GAAP national differences are important as financial concepts largely depend on the legal system of the country.

In summary, the work presented in this paper outlines an initial approach to domain-specific ontology translation. It provides an indication that external resources are useful for overcoming the sparsity of data, as well as a wealth of challenges to fuel future work on this task.

Acknowledgments

This work has been funded under the Seventh Framework Programme for Research and Technological Development of the European Commission through the T4ME contract (grant agreement no.: 249119) and in part by the European Union under Grant No. 248458 for the Monnet project as well as by the Science Foundation Ireland under Grant No. SFI/08/CE/I1380 (Lion-2). The authors would like to thank Noëmi Aepli, Tobias Wunner and Thierry Declerck for their help with the manual evaluation. We are grateful to the anonymous reviewers for their valuable feedback.

[22] `en.wiktionary.org/wiki/Wiktionary:Main_Page`
[23] `dbpedia.org/About`
[24] generated from *Trade Debtors*

References

Aggarwal, N., Wunner, T., Arcan, M., Buitelaar, P., and O'Riain, S. (2011). A similarity measure based on semantic, terminological and linguistic information. In *The Sixth International Workshop on Ontology Matching collocated with the 10th International Semantic Web Conference (ISWC'11)*.

Bennett, E. M., Alpert, R., and Goldstein, A. C. (1954). Communications Through Limited-response Questioning. *Public Opinion Quarterly*, 18(3):303–308.

Declerck, T., Krieger, H.-U., Thomas, S. M., Buitelaar, P., O'Riain, S., Wunner, T., Maguet, G., McCrae, J., Spohr, D., and Montiel-Ponsoda, E. (2010). Ontology-based multilingual access to financial reports for sharing business knowledge across europe. In *Internal Financial Control Assessment Applying Multilingual Ontology Framework*.

Denkowski, M. and Lavie, A. (2011). Meteor 1.3: Automatic Metric for Reliable Optimization and Evaluation of Machine Translation Systems. In *Proceedings of the Sixth Workshop on Statistical Machine Translation*, pages 85–91, Edinburgh, Scotland. Association for Computational Linguistics.

Doddington, G. (2002). Automatic evaluation of machine translation quality using n-gram co-occurrence statistics. In *Proceedings of the second international conference on Human Language Technology Research*, HLT '02, pages 138–145.

Erdmann, M., Nakayama, K., Hara, T., and Nishio, S. (2008). An approach for extracting bilingual terminology from wikipedia. *Lecture Notes in Computer Science*, (4947):380–392. Springer.

Federmann, C. (2012). Appraise: An open-source toolkit for manual evaluation of machine translation output. *The Prague Bulletin of Mathematical Linguistics*, 98:25–35.

Fišer, D., Vintar, v., Ljubešić, N., and Pollak, S. (2011). Building and using comparable corpora for domain-specific bilingual lexicon extraction. In *Proceedings of the 4th Workshop on Building and Using Comparable Corpora: Comparable Corpora and the Web*, BUCC '11, pages 19–26.

Fleiss, J. (1971). Measuring Nominal Scale Agreement among Many Raters. *Psychological Bulletin*, 76(5):378–382.

Kerremans, K. (2010). A comparative study of terminological variation in specialised translation. In *Reconceptualizing LSP Online proceedings of the XVII European LSP Symposium 2009*, pages 1–14.

Koehn, P., Hoang, H., Birch, A., Callison-Burch, C., Federico, M., Bertoldi, N., Cowan, B., Shen, W., Moran, C., Zens, R., Dyer, C., Bojar, O., Constantin, A., and Herbst, E. (2007). Moses: Open Source Toolkit for Statistical Machine Translation. In *Proceedings of the 45th Annual Meeting of the ACL*, ACL '07, pages 177–180.

Krippendorff, K. (2004). Reliability in Content Analysis. Some Common Misconceptions and Recommendations. *Human Communication Research*, 30(3):411–433.

Landis, J. and Koch, G. (1977). Measurement of Observer Agreement for Categorical Data. *Biometrics*, 33(1):159–174.

Müller, C. and Gurevych, I. (2008). Using wikipedia and wiktionary in domain-specific information retrieval. In *Working Notes for the CLEF 2008 Workshop*.

Och, F. J. and Ney, H. (2003). A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29.

Papineni, K., Roukos, S., Ward, T., and Zhu, W.-J. (2002). BLEU: A Method for Automatic Evaluation of Machine Translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, ACL '02, pages 311–318, Stroudsburg, PA, USA. Association for Computational Linguistics.

Scott, W. A. (1955). Reliability of Content Analysis: The Case of Nominal Scale Coding. *The Public Opinion Quarterly*, 19(3):321–325.

Snover, M., Dorr, B., Schwartz, R., Micciulla, L., and Makhoul, J. (2006). A study of translation edit rate with targeted human annotation. In *Proceedings of Association for Machine Translation in the Americas*, pages 223–231.

Steinberger, R., Pouliquen, B., Widiger, A., Ignat, C., Erjavec, T., Tufis, D., and Varga, D. (2006). The jrc-acquis: A multilingual aligned parallel corpus with 20+ languages. In *Proceedings of the 5th International Conference on Language Resources and Evaluation (LREC'2006)*.

Stolcke, A. (2002). Srilm-an extensible language modeling toolkit. In *Proceedings International Conference on Spoken Language Processing*, pages 257–286.

Tiedemann, J. (2009). News from OPUS - A collection of multilingual parallel corpora with tools and interfaces. In Nicolov, N., Bontcheva, K., Angelova, G., and Mitkov, R., editors, *Recent Advances in Natural Language Processing*, volume V, pages 237–248. John Benjamins, Amsterdam/Philadelphia, Borovets, Bulgaria.

Vivaldi, J. and Rodriguez, H. (2010). Using wikipedia for term extraction in the biomedical domain: first experiences. *Procesamiento del Lenguaje Natural*, 45:251–254.

Weller, M., Gojun, A., Heid, U., Daille, B., and Harastani, R. (2011). Simple methods for dealing with term variation and term alignment. In *Proceedings of the 9th International Conference on Terminology and Artificial Intelligence*, pages 87–93.

Wu, H., Wang, H., and Zong, C. (2008). Domain adaptation for statistical machine translation with domain dictionary and monolingual corpora. In *Proceedings of the 22nd International Conference on Computational Linguistics - Volume 1*, COLING '08, pages 993–1000.

Zesch, T., Müller, C., and Gurevych, I. (2008). Extracting lexical semantic knowledge from wikipedia and wiktionary. In *Proceedings of the Sixth International Conference on Language Resources and Evaluation (LREC'08)*.

The Floating Arabic Dictionary: An Automatic Method for Updating a Lexical Database through the Detection and Lemmatization of Unknown Words

Mohammed Attia[1,3] *Younes Samih*[2] *Khaled Shaalan*[1] *Josef van Genabith*[3]
(1) The British University in Dubai, UAE
(2) Heinrich-Heine-Universität, Germany
(3) School of Computing, Dublin City University, Ireland
{mattia,josef}@computing.dcu.ie,
samih@phil.uni-duesseldorf.de,
khaled.shaalan@buid.ac.ae

ABSTRACT

Unknown words, or out of vocabulary words (OOV), cause a significant problem to morphological analysers, syntactic parses, MT systems and other NLP applications. Unknown words make up 29 % of the word types in in a large Arabic corpus used in this study. With today's corpus sizes exceeding 10^9 words, it becomes impossible to manually check corpora for new words to be included in a lexicon. We develop a finite-state morphological guesser and integrate it with a machine-learning-based pre-annotation tool in a pipeline architecture for extracting unknown words, lemmatizing them, and giving them a priority weight for inclusion in a lexical database. The processing is performed on a corpus of contemporary Arabic of 1,089,111,204 words. Our method is tested on a manually-annotated gold standard and yields encouraging results despite the complexity of the task. Our work shows the usability of a highly non-deterministic morphological guesser in a practical and complex application.

TITLE IN ARABIC

القاموس العائم للغة العربية: طريقة آلية لتحديث قاعدة البيانات المعجمية من خلال اكتشاف الكلمات الغير معروفة وردها إلى أصلها

ABSTRACT IN ARABIC

تسبب الكلمات الغير معروفة أو الكلمات الغير مدونة في القواميس مشكلة كبيرة في التحليل الصرفي والإعراب الآلي والترجمة الآلية وغيرها من تطبيقات المعالجة الآلية للغات الطبيعية. فتشكل الكلمات الغير معروفة نسبة 29 % من الكلمات الموجودة في ذخيرة النصوص المستخدمة في هذا البحث. ومع الزيادة الهائلة في حجم ذخائر النصوص التي تتجاوز اليوم مليار كلمة يصبح من المستحيل إجراء أي بحث يدوي عن الكلمات الجديدة لإدراجها في المعاجم الحديثة. ولذلك قمنا بتطوير أداة للتخمين الصرفي قائمة على تقنية آلات الحالة المحدودة وتم دمجها مع أداة قائمة على التعلم الآلي واستخدمناهما معا في عملية تشبه خط الأنابيب تكون مخرجات بعض أجزائه مدخلات لأجزائه الأخرى بحيث نتمكن من استخراج الكلمات الغير معروفة وردها إلى أصلها وإعطائها وزنا يعبر عن الأولوية في الإدراج في قاعدة البيانات المعجمية. ويعتمد هذا البحث على ذخيرة نصوص حجمها 1,089,111,204 كلمة. وقد قمنا باختبار الطريقة التي طورناها باستخدام معيار تم بناؤه يدويا ويقدم نتائج مرضية بالرغم من تعقيد المهمة. وتبين الطريقة التي استخدمناها فائدة أداة التخمين الصرفي الذي يعطي نتائج بها درجة كبيرة من الغموض في تطبيقات عملية ومعقدة.

KEYWORDS : Arabic, unknown words, out of vocabulary words, floating dictionary, lexical enrichment, lexical extension

KEYWORDS IN ARABIC:
اللغة العربية، الكلمات الغير معروفة، الكلمات الغير مدرجة في القواميس، القاموس العائم، الإثراء المعجمي، التوسع المعجمي

Proceedings of COLING 2012: Technical Papers, pages 83–96,
COLING 2012, Mumbai, December 2012.

1 Introduction

Due to the complexity and semi-algorithmic nature of Arabic morphology (that employs numerous rules and constraints on inflection, derivation and cliticization), it has been a challenge for computational processing and analysis (Kiraz, 2001; Beesley 2003). A lexicon is an indispensable part of a morphological analyser (Dichy and Farghaly, 2003; Attia, 2006; Buckwalter, 2004; Beesley, 2001), and the coverage of the lexical database is a key factor in the coverage of the morphological analyser, and limitations in the lexicon will cascade through to higher levels of processing. Moreover, out of vocabulary words (or OOVs) have impact negatively on the performance of parsers (Attia et al., 2010) and MT applications (Huang et al. 2010). This is why an automatic method for updating a lexical database and dealing with unknown words is crucially important.

We present the first attempt, to the best of our knowledge, to address the lemmatization (rather than stemming) of Arabic unknown words. The problem with lemmatizing unknown words is that they cannot be matched against a morphological lexicon. Furthermore, the specific problem with lemmatizing Arabic words is the richness and complexity of Arabic morphological derivational and inflectional processes. For the purposes of this paper, unknown words are words not found by the SAMA morphological analyser (Maamouri et al., 2010) but accepted by the Microsoft Spell Checker. We develop a rule-based finite-state morphological guesser and use a machine learning based disambiguator, MADA (Roth et al., 2008), in a pipeline-based approach to lemmatization.

We test our method against a manually created gold standard of 1,310 types (unique words) and show a significant improvement over the baseline. Furthermore, we devise a novel algorithm for weighting and prioritizing new words for inclusion in a lexicon depending on three factors: number of form variations of the lemmas, cumulative frequency of the forms, and the type of POS (part of speech) tag.

This paper is structured as follows. The remainder of the introduction provides more details on the complexity of the lemmatization process in Arabic, why dealing with unknown words is important, previous work on the topic, and the data used in our experiments. Section 2 presents the methodology we follow in extracting and analysing unknown words. Section 3 provides details on the morphological guesser we develop to help deal with the problem. Section 4 presents and discusses the evaluation results, and Section 5 concludes.

1.1 Complexity of Lemmatization in Arabic

Arabic is an inflectionally rich language with nouns specified for number, gender and case; and verbs specified for tense, number, gender, person, voice and mood. These inflectional processes entail complex alterations on base forms. Arabic is also a clitic language. Clitics are morphemes that have the syntactic characteristics of a word but are morphologically bound to other words (Crystal, 1980). In Arabic, many coordinating conjunctions, the definite article, many prepositions and particles, and a class of pronouns are all clitics that attach themselves either to the start or end of words, and subsequently change the base form according to alteration rules which include assimilation and deletion. These facts complicate the process of lemmatization, or returning the base form given the inflected form.

For English, one can reasonably assume that new words appear very often in their base forms, or the lexical look-up forms. Lindén (2008) indicates that about 86 % of the new words in English appear in their base form. However, in Arabic, which is highly inflectional in nature, only 45 % of new token types in our test set appear in their base form. Moreover, 36 % of the unknown types do not appear in their base form at all in the entire corpus.

1.2 Why Deal with Unknown Words?

Sinclair (1987) introduced the term "Floating Dictionary", a self-updating dictionary that is able to automatically monitor language change. "It would, so to speak, float on top of a corpus, rather like a jelly-fish, its tendrils constantly sensing the state of the language." We think that an electronic 'floating dictionary' should be able to perform at least three major tasks. It should be able to tell which words are not is use anymore, which words have newly appeared in a language, and which word usages or senses have changed based on contemporary data. In this paper we explain our methodology for automatically detecting new words in Arabic, lemmatizing such new words in order to relate multiple surface forms to their base underlying representations, deciding on the word POS tag, collecting statistics on the frequency of use, and modelling human decisions on whether to include the new words in a lexicon or not.

New words are constantly finding their way into any living human language. These new words are either coined or borrowed, or they can be transliterations of proper nouns from other languages. The inclusion of new words in a lexicon is a non-trivial task as it needs to address two important problems. First, there is the problem of detection, or how do we know that a new word has appeared? Second, there is the problem of reaching a decision on the new word, or how do we judge whether the new word is worth adding to the lexicon or not? This is usually done by looking at whether the word is frequent enough, whether it appears in various forms and inflections, and whether it is well-distributed in a corpus. This enables us to determine whether the word constitutes a core lexical item or the usage of the word is just accidental or idiosyncratic.

We address this issue by developing an automatic technique to recognize unknown words and reduce them to their lemmas, predict their POS, and rank them in their order of importance.

1.3 Previous Work

Lemmatization of unknown words has been addressed for Slovene in (Erjavec and Džerosk, 2004), for Hebrew in (Adler at al., 2008) and for English, Finnish, Swedish and Swahili in (Lindén, 2008). Apart from the language involved, our work is different in that we incorporate a finite state guesser in the process. Lemmatization of Arabic words has been addressed in (Roth et al., 2008; Dichy, 2001). The idea of finding and stemming unknown Arabic words has been utilized by Diab et al, (2004). While Diab et al. do not mention unknown words specifically, the fact that they use a character-based classification model and tokenization indicates that they can handle unknown words and perform stemming on them. However, they do not present any evaluation on unknown words specifically. Mohamed and Kübler (2010) handle unknown words explicitly and provide results for known and unknown words in both word segmentation (stemming) and part of speech tagging. They reach a stemming accuracy of 81.39 % on unknown words and over 99 % on known words.

Diab et al.'s and Mohammed and Kübler's work focuses on stemming rather than lemmatization, which are quite distinct albeit frequently confused. The difference between stemming and lemmatization is that stemming strips off prefixes and suffixes and leaves the bare stem, while lemmatization returns the canonical base form. To illustrate this with an example, take the Arabic verb form يقولون 'yqwlwn' "they say". Stemming will remove the present prefix 'y' and the plural suffix 'wn' and leave قول 'qwl' which is a non-word in Arabic. By contrast, full lemmatization will reveal that the word has gone through an alteration process and return the canonical قال 'qAl' "to say" as the base form.

Lemmatization reduces surface forms to their canonical base representations (or dictionary look-up form), i.e, words before undergoing any inflection, which, in Arabic, means verbs in their perfective, indicative, 3rd person, masculine, singular forms, such as شَكَرَ $akara "to thank"; and nominals (the term used for both nouns and adjectives) in their nominative, singular, masculine forms, such as طالِب TAlib "student"; and nominative plural for *pluralia tantum* nouns (or nouns that appear only in the plural form and are not derived from a singular form) , such as ناس nAs "people".

1.4 Data Used

In our work we use a large-scale corpus of 1,089,111,204 words, consisting of the Arabic Gigaword Fourth Edition (Parker et al., 2009) with 925,461,707 words, in addition to 163,649,497 words from news articles crawled from the Al-Jazeera web site. In this corpus, unknown words appear at a rate between 2 % of word tokens (when we ignore possible spelling variants) and 9 % of word tokens (when possible spelling variants are included). In this context spelling variants refer to alternative (sub-standard) spellings recognized by SAMA which are mostly related to the possible overlap between orthographically similar letters, such as the various shapes of *hamzahs* (أ إ ا آ), *taa' marboutah* and *haa'* (ة ه), and *yaa'* and *alif maqsoura* (ي ى).

2 Methodology

To deal with unknown (or out-of-vocabulary) words, we use a pipeline approach which predicts part-of-speech tags and morpho-syntactic features before lemmatization. In the first stage of the pipeline, we use MADA (Roth et al., 2008), an SVM-based tool that relies on the word context to assign POS tags and morpho-syntactic features. MADA internally uses the SAMA morphological analyser (Maamouri et al., 2010), an updated version of Buckalter morphology (Buckwalter, 2004). Second, we develop a finite-state morphological guesser that can provide all the possible interpretations of a given word. The morphological guesser first takes an Arabic surface form as a whole and then strips all possible affixes and clitics off one by one until all possible analyses are exhausted. The morphological guesser is highly non-deterministic as it outputs a large number of solutions. To counteract this non-determinism, all the solutions are matched against the POS and morpho-syntactic tag output for the full surface token by MADA and the analysis with the closest resemblance (i.e. the analysis with the largest number of matching morphological features) is selected.

Beside the complexity of lemmatization described in Section 1.1, the problem is further compounded when dealing with unknown words that cannot be matched by existing lexicons. This requires the development of a finite-state guesser to list all the possible interpretations of an unknown string of letters (explained in detail in Section 3).

To identify, extract and lemmatize unknown Arabic words we use the following sequence of processing steps (Figure 1):

- A corpus of 1,089,111,204 tokens (7,348,173 types) is analysed with MADA.
- The number of types for which MADA could not find an analysis in the Buckwalter morphological analyser is 2,116,180 (about 29 % of the types).

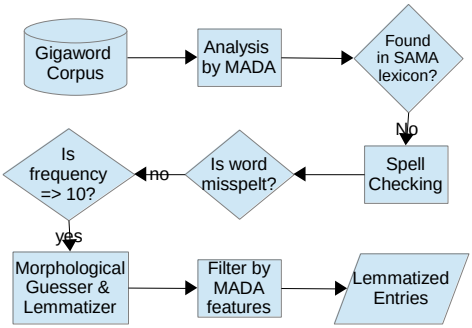


FIGURE 1 – Lemmatization process

- These unknown types were spell checked by the Microsoft Arabic spell checker using MS Office 2010. Among the unknown types of 2,116,180, the number of types accepted as correct is 208,188. The advantage of using spell checking at this stage is that it provides significant filtration of the forms (almost 90 % reduction) and retains a more compact, more manageable, and better quality list of entries to deal with in further processing. The disadvantage is that there is no guarantee that all word forms not accepted by the MS speller are actually spelling mistakes (or that all the ones accepted are correct).
- We select types with frequency of 10 or more of the types accepted by the MS spell checker. This results in a total of 40,277 types.
- We use the full POS tags and morpho-syntactic features produced by MADA.
- We use the finite-state morphological guesser to produce all possible morphological interpretations and relevant lemmatizations.
- We compare the POS tags and morphosyntactic features in MADA output with the output of the morphological guesser and choose the one with the highest matching score.

For testing and evaluation we gold annotate 1,310 words randomly selected from the 40,277 types, providing the gold lemma, the gold POS and lexicographic preference for inclusion in a dictionary. It is to be noted that working with the 2,116,180 types before filtering out possible spelling errors will require annotating a much larger gold standard.

3 Morphological Guesser

Arabic morphotactics allows words to be concatenated with a comparatively large number of clitics (Attia, 2006). Clitics themselves can be concatenated one after the other. Furthermore, clitics undergo assimilation with word stems and with each other, which makes them even harder to handle using surface features only. A verb can comprise up to four tokens (a conjunction, complementizer, verb stem and object pronoun) as illustrated in Table 1. Moreover the verb stem can be prefixed and suffixed with bound morphemes that mark the morpho-syntactic features of tense, number, gender, person, voice and mood. The lemma resides as a nucleus inside layers of proclitics, prefixes, suffixes and enclitics. A verb lemma like شكر '$akara' "to thank" can generate up to 2,552 different valid forms.

Proclitics		**Prefix**	**Lemma**	**Suffix**	**Enclitic**
Conjunction/ question article	*Comp*	*Tense/mood – number/gender*	*Verb*	*Tense/mood – number/gender*	*Object pronoun*
Conjunctions و wa 'and' or ف fa 'then'	ل li 'to'	Imperfective tense (5)	lemma	Imperfective tense (10)	First person (2)
Question word أ > 'is it true that'	س sa 'will'	Perfective tense (1)		Perfective tense (12)	Second person (5)
	ل la 'then'	Imperative (2)		Imperative (5)	Third person (5)

TABLE 1 – Proclitics, enclitics, prefixes and suffixes with Arabic verbs

Proclitics			**lemma**	**Suffix**	**Enclitic**
Conjunction/ question article	*Preposition*	*Definite article*	*Noun*	*Gender/Number*	*Genitive pronoun*
Conjunctions و wa 'and' or ف fa 'then'	ب bi 'with', ك ka 'as' or ل li 'to'	ال Al 'the'	Stem	Masculine Dual (4)	First person (2)
				Feminine Dual (4)	
Question word أ > 'is it true that'				Masculine regular plural (4)	Second person (5)
				Feminine regular plural (1)	Third person (5)
				Feminine Mark (1)	

TABLE 2 – Proclitics, enclitics, prefixes and suffixes with Arabic nouns

Similarly a noun stem can be attached to up to three clitics as shown in Table 2. Although Table 2 shows four clitics, we note that the definite article and the genitive (or possessive) pronoun are mutually exclusive. Nominal stems can also be suffixed with bound morphemes that mark the morpho-syntactic features of number, gender and case. a typical noun like معلم 'muEal~im' 'teacher', generates 519 valid forms.

We develop a finite state (Beesley and Karttunen, 2003; Hulden, 2009) morphological guesser for Arabic that can analyse unknown words with all possible clitics, morpho-syntactic affixes and all relevant alteration operations that include insertion, assimilation, and deletion. Beesley and Karttunen (2003) give some advice on how to create a basic guesser. The core idea of a guesser is to assume that a stem is composed of any arbitrary sequence of non-numeric characters, and this stem can be prefixed and/or suffixed with a predefined set of prefixes, suffixes or clitics. The guesser marks clitic boundaries and tries to return the stem to its default unmarked form, the lemma. Due to the nondeterministic nature of the guesser, there will be a multitude of possible lemmas for each form. The Arabic FST guesser consists of three parts: a lexc file, alteration rules and an XFST compilation file. First, there is the lexc file (Figure 2) with lexicons and continuation classes for the Arabic guesser. The lexc file specifies that there is an optional conjunction, followed by an optional preposition, followed by an optional definite article before the Arabic noun.

```
LEXICON Conjunctions
و-:و-+conj:                    Prepositions;
ف-:ف-+conj:                    Prepositions;
                               Prepositions;

LEXICON Prepositions
ل-:ل-+prep:                    Article;
ك-:ك-+prep:                    Article;
ب-:ب-+prep:                    Article;
                               Article;

LEXICON Article
ال-+defArt                     Nouns;
                               Nouns;

LEXICON Nouns
+noun+fem                      GuessWords;
+noun+masc                     GuessWords;
^ss^خادم^se^+noun+masc         FemMascduFemduMascplFempl;
....
LEXICON GuessWords
^ss^^GUESSNOUNSTEM^^se^        FemMascduFemduMascplFempl;
^ss^^GUESSNOUNSTEM^^se^        FemMascduFemduFempl;
^ss^^GUESSNOUNSTEM^^se^        FemMascduFemdu;
^ss^^GUESSNOUNSTEM^^se^        MascduFempl;
^ss^^GUESSNOUNSTEM^^se^        Mascdu;
^ss^^GUESSNOUNSTEM^^se^        Fempl;
^ss^^GUESSNOUNSTEM^^se^        FemduFempl;
^ss^^GUESSNOUNSTEM^^se^        Femdu;
^ss^^GUESSNOUNSTEM^^se^        NoNumber;
```

FIGURE 2 – Snapshot of the Arabic lexc file

Second, there are the alteration rules which handle the morphological processes of assimilation and deletion. In our system there are about 130 replace rules to handle alterations that affect verbs, nouns, adjectives and function words when they undergo inflections or are attached to affixes and clitics. They take the form of XFST replace rules:

```
A -> w || "+pres" Alphabet _ Alphabet
```

The example rule indicates that ‘A’ changes to ‘w’ under the condition of having the left context ‘+pres’ and a single alphabetical character and the right context of another alphabetical character. Following this rule the verb قال qAl “to say” will change to يقول yaqwl in the present tense form.

Third, there are the XFST compilation rules which bind components together. They replace the multivariable words ‘GUESSNOUNSTEM’ and ‘GUESSVERBSTEM’ with the relevant alphabet using the ‘substitute defined’ command. The XFST commands in our guesser are stated as follows.

```
define Alphabet
define PossNounStem [[Alphabet]^{2,24}] "+Guess":0;
define PossVerbStem [[Alphabet]^{2,6}] "+Guess":0;
substitute defined PossNounStem for "^GUESSNOUNSTEM^"
substitute defined PossVerbStem for "^GUESSVERBSTEM^"
```

This states that a possible noun stem is defined as any sequence of Arabic non-numeric characters of length between 2 and 24 characters. A possible verb stem is between 2 and 6 characters. This word stem is surrounded by prefixes, suffixes, proclitics and enclitics. Clitics are considered as independent tokens and are separated by the ‘@’ sign, while prefixes and suffixes are considered as morpho-syntactic features and are interpreted with tags preceded by the ‘+’ sign. Below we present the analysis of the noun والمُسَوِّقونَ wa-Al-musaw~iquwna “and-the-marketers”, and the verb سَيَأْخُذُنا sa-ya'xu*unA “will-take-us”.

MADA output for wa-Al-musaw~iquwna:

form:wAlmswqwn	num:p	gen:m	per:na	case:n	asp:na	mod:na	vox:na	pos:noun
prc0:Al_det	prc1:0	prc2:wa_conj		prc3:0	enc0:0	stt:d		

Finite-state guesser output for wa-Al-musaw~iquwna:

```
والمسوقون  +adjوالمسوقّ+Guess+masc+pl+nom@
والمسوقون  +adjوالمسوقون+Guess+sg@
والمسوقون  +nounوالمسوقّ+Guess+masc+pl+nom@
والمسوقون  +nounوالمسوقون+Guess+sg@
والمسوقون  +conj@و+defArt@ال+adjمسوقّ+Guess+masc+pl+nom@
والمسوقون  +conj@و+defArt@ال+adjمسوقون+Guess+sg@
والمسوقون  +conj@و+defArt@ال+nounمسوقّ+Guess+masc+pl+nom@      [correct match]
والمسوقون  +conj@و+defArt@ال+nounمسوقون+Guess+sg@
…
```

MADA output for wa-sa-ya'xu*unA:

form:sy>x*nA	num:s	gen:m	per:na	case:na	asp:na	mod:i	vox:a	pos:verb
prc0:0	prc1:0	prc2:0	prc3:0	enc0:1p_poss	stt:na			

Finite-state guesser output for wa-sa-ya'xu*unA:

```
سيأخذنا  +adjسيأخذن+Guess+dual+nom+compound@
سيأخذنا  +adjسيأخذنا+Guess+sg@
سيأخذنا  +nounسيأخذ+Guess+sg@نا+genpron+1pers+@
سيأخذنا  +nounسيأخذنا+Guess+sg@
سيأخذنا  +verb+impسيأخذ+Guess+2pers+masc+sg@نا+objpron+1pers+pl@
سيأخذنا  +verb+impسيأخذن+Guess+2pers+dual@
سيأخذنا  س+fut+art@+verb+pres+pass+3persأخذنا+Guess+masc+sg@
سيأخذنا  س+fut+art@+verb+pres+active+3persأخذ
          +Guess+masc+sg@نا+objpron+1pers+pl@                    [correct match]
سيأخذنا  س+fut+art@+verb+pres+active+3persأخذنا+Guess+masc+sg@
…
```

For a list of 40,277 unknown word types, the morphological guesser produces an average of 12.6 possible interpretations per word. This is highly non-deterministic when compared to the finite state morphological analyser (Attia et al., 2011) which has an average of 2.1 solutions per known word. We also note that 97 % of the gold lemmas in our test set are found among the finite-state guesser's choices, which indicates the high performance of the guesser.

4 Testing and Evaluation

To evaluate our methodology we create a manually annotated gold standard test suite of randomly selected surface form types as mentioned in Section 2. For these surface forms, the gold lemma and part of speech are manually provided. In addition, a human annotator indicates a preference on whether or not to include the entry in a dictionary, that is whether a lemmatized form makes a valid dictionary entry or not. We noticed that most of the forms marked by the annotator as not fitting for inclusion in a dictionary were proper nouns, misspelled words, colloquial words, and words that form a part of a multiword expression. By contrast, nouns, verbs adjectives, and proper nouns with significantly high frequency were marked for inclusion in the lexical database. It is to be mentioned that proper nouns in Arabic are not orthographically distinguished from other words, i.e. there is no capitalization in Arabic as is the case in European languages. This feature of lexicographic preference helps to evaluate our lemma weighting algorithm discussed in Section 4.2. The size of the test suite is 1,310 word form types.

We observe that proper nouns are the most frequent category (45 %) among the unknown words types in the data, and they also cover about 61 % of the unknown token instances in the gold annotated dataset. The POS distribution of the unknown token types of our annotated data is shown in Table 3. As expected, most unknown words are open class words: proper names, nouns, adjectives, and, to a lesser degree, verbs.

Gold POS	**Type Count**	**Ratio**
noun_prop	584	45 %
noun	264	20 %
adj	255	19 %
verb	52	4 %
noun_fem_plural (pluralia tantum)	28	2 %
noun_broken_plural	28	2 %
others: noun_masc_plural (*pluralia tantum*) (4) part (3) pron_dem (1)	8	0.6 %
Excluded		
misspelling	55	4 %
not_known	15	1 %
colloquial	19	1.5 %
Lexicographic relevance		
Include in a dictionary	671	51 %
Don't include in a dictionary	639	49 %

TABLE 3 – Gold tag annotation of the test suite

4.1 Evaluating Lemmatization

In the evaluation experiment we measure accuracy calculated as the number of correct tags divided by the count of all tags. The baseline is given by the assumption that new words appear in their base form, i.e., we do not need to lemmatize them. The baseline accuracy is 45 %. The POS tagging baseline proposes the most frequent tag (proper name) for all unknown words. In our test data accuracy stands also at 45 %. We notice that MADA POS tagging accuracy for unknown words is unexpectedly low (60 %) as shown in Table 4. We use Voted POS Tagging, that is we choose the POS tag assigned most frequently in the data to a lemma. This method has improved the tagging results significantly (Table 4).

As for the lemmatization process, our first experiment in the pipeline-based lemmatization approach obtains a higher score (54 %) than the baseline (45 %) as shown in Table 5.

		Accuracy
	POS tagging	
1	POS Tagging baseline	45 %
2	MADA POS tagging	60 %
3	Voted POS Tagging	69 %

TABLE 4 – Evaluation of POS tagging of unknown words

Examining the data further, we notice that when a proper noun is prefixed with the definite article "Al", the definite article is not stripped off in the gold annotation and is considered as part of the lemma, such as القشيري 'Al-qu$ayriy'. In MADA morpho-syntactic tagging, the definite article is considered as a clitic and not part of the lemma. When this difference is ignored in the second experiment, the lemmatization accuracy increases from 54 % to 63 %. A more detailed error analysis will help devise better heuristics to increase the accuracy of the pipeline-based lemmatization. For example, in the gold annotation some regular feminine and masculine plural forms are considered as *pluralia tantum*, while in the automatic lemmatization they are reduced to their singular forms, such as حجوزات HujuwzAt "bookings".

	Lemmatization	
1	Lemmas found among corpus forms	64 %
2	Lemmas found among fst guesser forms	97 %
3	Lemma selection baseline	45 %
4	Pipeline-based lemmatization (selection decision) with strict definite article matching	54 %
5	Pipeline-based lemmatization (selection decision) ignoring definite article matching	63 %

TABLE 5 – Evaluation of lemmatization of unknown words

The test results indicate significant improvements over the baseline. However, we expect that substantial further improvements can be obtained through further extensive error analysis and developing refined heuristics.

4.2 Evaluating Lemma Weighting

We create a weighting algorithm for ranking and prioritizing unknown words in Arabic so that important words that are valid for inclusion in a lexicon are pushed up the list and less interesting words (from a lexicographic point of view) are pushed down. This is meant to facilitate the effort of manual revision by making sure that the top part of the stack contains the words with highest priority.

In our case we have 40,277 unknown token types. After lemmatization they are reduced to 18,399 types (that is 54 % reduction of the surface forms). This number is still too big for manual validation. In order to address this issue we devise a weighting algorithm for ranking so that the top *n* number of words will include the most lexicographically relevant words. We call surface forms that share the same lemma 'sister forms', and we call the lemma that they share the 'mother lemma'. The weighting algorithm is based on three criteria: number of sister forms, cumulative frequency of the sister forms, and a POS factor. The POS factor gives 50 extra points to verbs, 30 to to nouns and adjectives, and nothing to proper nouns. This is meant to penalize proper nouns due to their high frequency which is disproportionate to other categories. The parameters of the weighting algorithm have been tuned through several rounds of experimentation.

```
Word Weight = ((number of sister forms * 800) +
      cumulative sum of frequencies of sister forms) / 2 +
    POS factor
```

We use the gold annotated data for the evaluation of the lemma weighting criteria, as shown in Table 6. We notice that the combined criteria gives the best balance between increasing the number of lexicographically-relevant words in the top 100 words and reducing the number of lexicographically-relevant words in the bottom 100 words.

Lexicographically-relevant words	In top 100	In bottom 100
relying on Frequency alone (baseline)	63	50
relying on number of sister forms * 800	87	28
relying on POS factor	58	30
using combined criteria	78	15

TABLE 6 – Evaluation of lemma weighting and ranking

Table 7 shows a sample of the entries in the unknown words lexicon. The list includes a spectrum of the different word categories such as proper nouns, adjectives, nouns, broken plural and feminine plural forms, as well as verbs.

#	FST Guessed lemma	Gloss	Weight	Forms
Proper Nouns				
1	أوباما >ubAmA	Obama	40421	لأوباما # أوباما # فأوباما # وأوباما # ولأوباما # بأوباما
2	ساركوزي sArkuwziy	Sarkozy	29361	وساركوزي # فساركوزي # بساركوزي # ساركوزي
3	توتنهام tuwtinhAm	Tottenham	08829	بتوتنهام # وتوتنهام # لتوتنهام # ولتوتنهام # توتنهام
Adjectives				
4	منخرط munxariT	involved	09302	والمنخرطة # ومنخرطة # منخرطات # منخرطة # المنخرطة # المنخرطات
5	متواطئ mutawAti}	conspiring	07016	متواطئان # ومتواطئ # متواطئين # المتواطئ # كمتواطئين # والمتواطئ # والمتواطئون # المتواطئين # ومتواطئين # متواطئون # للمتواطئين # والمتواطئين # متواطئ # ومتواطئون
6	مستتر musotatir	hidden	03329	والمستتر # المستترين # المستتر # مستتر # مستترين # والمستترين # ومستتر
Nouns				
7	اقتياد AqotiyAd	leading	08559	واقتياده # واقتيادهما # اقتياد # لاقتياده # واقتيادهم # لاقتياد # اقتيادها # الاقتياد # باقتياد # باقتياده # وباقتياده # باقتيادهم # اقتيادهم # اقتيادنا # اقتيادي # واقتياد # واقتيادها # اقتيادهما
8	محاصصة muHASaSap	sharing	07056	المحاصصة # ومحاصصة # للمحاصصة # محاصصة # بالمحاصصة # بمحاصصة # والمحاصصة # فالمحاصصة
9	ارتهان ArotihAn	dependence	06616	الارتهان # وارتهان # وارتهانها # ارتهانها # ارتهان # والارتهان # بالارتهان # وارتهانه # ارتهانه # ارتهانهم # لارتهان
Broken Plurals				
10	خصال xiSAl	features	08491	بخصالك # خصال # وخصال # خصاله # الخصال # لخصاله # خصالهم # بخصال # خصالنا # خصالك # بخصاله # بالخصال # خصالها # وخصاله # وخصالها # لخصال # والخصال # وخصالك
11	مكائد makA}id	tricks	05785	للمكائد # مكائدهم # بالمكائد # والمكائد # مكائد # مكائده # المكائد # ومكائده # ومكائدهم # بمكائد # لمكائد # ومكائد # ومكائدها
12	دفوع dufuwE	defences	04418	بالدفوع # الدفوع # دفوعه # دفوعهم # دفوع # والدفوع # ودفوع # دفوعها # بدفوع
Feminine plural forms				
13	صياغة Siyagap	formation	07168	وبصياغات # وصياغات # وصياغاتها # صياغاتهم # بصياغات # صياغتين # والصياغات # لصياغات # صياغاتها # صياغات # الصياغتين # بالصياغات # الصياغات # وصياغاته # صياغاته
14	خصومة xuSuwmap	animosity	06728	والخصومات # خصوماتهم # خصوماته # بخصومات # وخصوماته # خصومات # لخصومات # وخصوماتهم # خصوماتها # خصوماتنا # وخصوماتها # بالخصومات # الخصومات
15	مرارة marArap	bitterness	05339	مراراتها # المرارات # بمراراته # لمرارات # مراراته # والمرارات # بمرارات # ومرارات # مرارات # ومراراتها
Verbs				
16	عسكر Easokara	to militarize	05255	عسكرين # وعسكرت # ليعسكر # يعسكرون # وسيعسكر # العسكرين # سيعسكر # ويعسكر # يعسكر # عسكروا # ستعسكر
17	سيس say~asa	to politicize	04223	سيس # يسيسوا # بسيسي # سيسون # وسيس # تسيس # تسيسوا # وسيسون # يسيسون
18	هندس hanodasa	to design/ engineer	03431	هندس # هندسها # يهندسوا # هندست # يهندسون # هندسوا # يهندس # يهندسها

TABLE 7 – Sample entries selected from the unknown words lexicon

As the corpus is composed mainly of news articles, we assume that the distribution of proper nouns is artificial and arbitrary as it depends, to a large extent, on the specific date and time of an event or series of events that occupies the news for a certain (short-term or long-term) duration. For example, as Table 7 shows, *Obama* and *Sarkozy* ranked top of the list of unknown words, but

now as Sarkozy is no longer the French president and the fate of Obama will be determined in the next presidential election in America, whether these names will continue to maintain the same level of frequency is questionable. This is why verbs, adjectives and nouns constitute the core of the language lexicon, while proper nouns are, to some extent, temporal and transient and the frequency of their use tends to shift from time to time.

Conclusion

We have developed a methodology for automatically updating an Arabic dictionary by extracting unknown words from data and lemmatizing them in order to relate multiple surface forms to their canonical underlying representation using a finite-state guesser and a machine learning tool for disambiguation. We have developed a weighting mechanism for simulating a human decision on whether or not to include new words in a general-domain lexical database. We have shown the feasibility of a highly non-deterministic finite state guesser in an essential application. Out of a word list of 40,255 unknown words we created a lexicon of 18,399 lemmatized, POS-tagged and weighted entries. We have made our unknown word lexicon available as a free open source resource (http://arabic-unknowns.sourceforge.net/).

Acknowledgments

This research is funded by the Irish Research Council for Science Engineering and Technology (IRCSET), the UAE National Research Foundation (NRF) (Grant No. 0514/2011), and the Science Foundation Ireland (Grant No. 07/CE/I1142) as part of the Centre for Next Generation Localisation (www.cngl.ie) at Dublin City University.

References

Adler, M., Goldberg, Y., Gabay, D. and Elhadad, M. (2008). Unsupervised Lexicon-Based Resolution of Unknown Words for Full Morpholological Analysis. In: Proceedings of Association for Computational Linguistics (ACL), Columbus, Ohio.

Attia, M. (2006). An Ambiguity-Controlled Morphological Analyzer for Modern Standard Arabic Modelling Finite State Networks. In: Challenges of Arabic for NLP/MT Conference, The British Computer Society, London, UK.

Attia, Mohammed, Jennifer Foster, Deirdre Hogan, Joseph Le Roux, Lamia Tounsi and Josef van Genabith. (2010). 'Handling Unknown Words in Statistical Latent-Variable Parsing Models for Arabic, English and French'. First Workshop on Statistical Parsing of Morphologically Rich Languages (SPMRL 2010), NAACL HLT. Los Angeles, CA.

Attia, Mohammed, Pavel Pecina, Lamia Tounsi, Antonio Toral, Josef van Genabith. (2011). An Open-Source Finite State Morphological Transducer for Modern Standard Arabic. International Workshop on Finite State Methods and Natural Language Processing (FSMNLP). Blois, France.

Beesley, K. R. (2001). Finite-State Morphological Analysis and Generation of Arabic at Xerox Research: Status and Plans in 2001. In: The ACL 2001 Workshop on Arabic Language Processing: Status and Prospects, Toulouse, France.

Beesley, K. R., and Karttunen, L.. (2003). Finite State Morphology: CSLI studies in computational linguistics. Stanford, Calif.: Csli.

Buckwalter, T. (2004). Buckwalter Arabic Morphological Analyzer (BAMA) Version 2.0. Linguistic Data Consortium (LDC) catalogue number LDC2004L02, ISBN1-58563-324-0

Crystal, D. (1980). *A First Dictionary of Linguistics and Phonetics*. London: Deutsch.

Diab, Mona, Kadri Hacioglu and Daniel Jurafsky. (2004). Automatic Tagging of Arabic Text: From Raw Text to Base Phrase Chunks. Proceedings of Human Language Technology-North American Association for Computational Linguistics (HLT-NAACL)

Dichy, J. (2001). On lemmatization in Arabic, A formal definition of the Arabic entries of multilingual lexical databases. ACL/EACL 2001 Workshop on Arabic Language Processing: Status and Prospects. Toulouse, France.

Dichy, J., and Farghaly, A. (2003). Roots & Patterns vs. Stems plus Grammar-Lexis Specifications: on what basis should a multilingual lexical database centred on Arabic be built? In: The MT-Summit IX workshop on Machine Translation for Semitic Languages, New Orleans.

Erjavec, T., and Džerosk, S. (2004). Machine Learning of Morphosyntactic Structure: Lemmatizing Unknown Slovene Words. *Applied Artificial Intelligence*, 18:17–41.

Huang, Chung-chi, Ho-ching Yen and Jason S. Chang. (2010). Using Sublexical Translations to Handle the OOV Problem in MT. in Proceedings of The Ninth Conference of the Association for Machine Translation in the Americas(AMTA).

Hulden, M. (2009). Foma: a finite-state compiler and library. In: Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics (EACL '09). Stroudsburg, PA, USA.

Kiraz, G. A. (2001). *Computational Nonlinear Morphology: With Emphasis on Semitic Languages*. Cambridge University Press.

Lindén, K. (2008). A Probabilistic Model for Guessing Base Forms of New Words by Analogy. In CICling-2008, 9th International Conference on Intelligent Text Processing and Computational Linguistics, Haifa, Israel, pp. 106-116.

Maamouri, M., Graff, D., Bouziri, B., Krouna, S., and Kulick, S. (2010). LDC Standard Arabic Morphological Analyzer (SAMA) v. 3.1. LDC Catalog No. LDC2010L01. ISBN: 1-58563-555-3.

Mohamed, Emad; Sandra Kübler (2010). Arabic Part of Speech Tagging. Proceedings of LREC 2010, Valetta, Malta.

Parker, R., Graff, D., Chen, K., Kong, J., and Maeda, K. (2009). Arabic Gigaword Fourth Edition. LDC Catalog No. LDC2009T30. ISBN: 1-58563-532-4.

Roth, R., Rambow, O., Habash, N., Diab, M., and Rudin, C. (2008). Arabic Morphological Tagging, Diacritization, and Lemmatization Using Lexeme Models and Feature Ranking. In: Proceedings of Association for Computational Linguistics (ACL), Columbus, Ohio.

Sinclair, J. M. (ed.). (1987). *Looking Up: An Account of the COBUILD Project in Lexical Computing*. London: Collins.

Contribution of complex lexical information to solve syntactic ambiguity in Basque

Aitziber ATUTXA Eneko AGIRRE Kepa SARASOLA
Ixa Group, University of the Basque Country (UPV/EHU),
Manuel Lardizabal pasealekua, 1 · 20018 Donostia-San Sebastián
{aitziber.atucha,e.agirre,kepa.sarasola}@ehu.es

ABSTRACT
In this study, we explore the impact of complex lexical information to solve syntactic ambiguity, including verbal subcategorization in the form of verbal transitivity and verb-noun-case or verb-noun-case-auxiliary relations. The information was obtained from different sources, including a subcategorization dictionary extracted from a Basque corpus, the web as a corpus, an English corpus and a Basque dictionary. Functional ambiguity between subject and object is a widespread problem in Basque, where 22% of subjects and objects are ambiguous, and this ambiguity surfaces in 33% of the sentences. This problem is comparable to PP attachment ambiguities in other languages. Our results show that, using complex lexical information, our results are better than a state-of-the-art statistical parser, obtaining a statistically significant error reduction of 20%. The disambiguation system is independent on the actual parsing algorithm used. The analysis revealed that the most relevant information are the case carried by the noun and the transitivity of the verb.

TITLE AND ABSTRACT IN BASQUE

Informazio lexikal konplexuaren ekarpena euskarazko anbiguotasun sintaktikoen ebazpenean

Lan honetan informazio lexikal konplexua erabiltzearen garrantzia aztertzen dugu euskarazko anbiguotasun sintaktikoen ebazpenean. Aditzen iragankortasuna erakusten duen azpikategorizazioaren ekarpena aztertu dugu, baita aditz-izen-kasu eta aditz-izen-kasu-laguntzaile erlazioena ere. Informazio horiek hainbat iturritatik jaso ditugu: euskarazko corpus batetik, webetik berau corpus gisa hartuta, ingelesezko corpus batetik eta euskarazko hiztegi batetik. Subjektu eta objektuaren arteko anbiguotasun funtzionala maiz aurkitzen dugu euskarazko testuetan; subjektua edo objektua bereiztea kasuen %22an ambiguoa da, eta hori gertatzen da perpausen %33an. Horrela, arazo horren garrantzi handia konparagarria da beste hizkuntza batzuek duten PP attachment arazoarenarekin. Gure sistemaren emaitzak hobeak dira artearen egoerako analizatzaile sintaktiko estatistiko batenak baino, estatistikoki esanguratsua den %20ko errore-murrizketa lortzen baitu. Analisi sintaktikoa egiteko edozein algoritmorekin erabil daiteke desanbiguazio-sistema hau.

KEYWORDS: Syntactic ambiguity resolution, subcategorization, web as a corpus.

KEYWORDS IN BASQUE: Anbiguotasun sintaktikoaren ebazpena, azpikategorizazioa, amarauna corpus gisa.

Proceedings of COLING 2012: Technical Papers, pages 97–114,
COLING 2012, Mumbai, December 2012.

1 Introduction

Due to typological differences, ambiguities in some languages differ from ambiguities in other. For example, while prepositional phrase (PP) attachment ambiguity occurs in about 50% of the English sentences in the Penn treebank (Volk, 2006), it occurs in less than 0.1% of sentences in Basque Dependency Treebank (Aduriz et al., 2003). By contrast, the subject-object ambiguity does not pose any problem in English, but it does in Basque, where 22% of the subjects or objects are ambiguous and 33% of the sentences show this ambiguity. The pervasive nature of this ambiguity makes it worth specific analysis. This problem is relevant to processing similar ambiguities in other morphologically rich languages. For instance, Urdu and Hindi also display subject-object ambiguity due to erg-abs markings, as well as null-case markings (Dixon, 1994; Husain and Agrawal, 2012).

In the literature, we find several approaches to improve syntactic disambiguation. One of them involves focusing on solving a relevant ambiguity by using a problem specific classifier (Kübler et al., 2007; Anguiano and Candito, 2011). This allows to deeply understand the features involved in the ambiguity. The results obtained over the localized ambiguous relations are either used in a post-process by replacing those of a parser (a process also known as *parse correction*) or, after analyzing the most informative features, those are incorporated in the treebank and used to improve a statistical parser (Husain and Agrawal, 2012).

In this paper we will follow the first approach, targeting the resolution of the subject-object ambiguity in Basque, comparing our results to those obtained by the Malt parser (Nivre et al., 2007). In any case, our method is independent of the parser used, and could be incorporated to statistic or rule-based parsers (Aranzabe et al., 2004) as well.

The paper is structured as follows. We will first review the subject-object ambiguity in Basque. In Section 3 we review Basque Dependency Treebank and the methods to acquire verbal subcategorization information. Section 4 presents the features that are informative when resolving the ambiguity. In Section 5 the method to create the gold standard is presented, alongside our disambiguation method and the results. The related work is discussed in Section 6. Finally, the conclusions and future work are drawn.

2 Subject-Object ambiguity in Basque

Typologically, Basque is a highly inflectional head-final language (Ortiz de Urbina, 1989; Laka, 1996). It belongs to what has been called MoR-FWO languages, that is, morphologically rich, free word order languages (such as Czech, Turkish, Hindi etc). In most of the cases the relation between a head and its dependent gets realized through a morphological marker, neither bore by the head nor the dependent. This implies examination of elements occurring in non-local environments.

(1) [***Pertsona*** *nagusi gehien**ek***], *euren etxeetan bizitzen jarraitu nahi dute.*
[**People** aged most-**erg**], their homes live keep-on want auxiliary.
[Most of aged people] want to keep living in their own home.

Example 1 shows a noun phrase headed by *pertsona*, where the ergative marker (**ek**) is carried by the last element of the noun phrase, in this case *gehienek*.

Basque is a morphological ergative language (Dixon, 1994) as well in both case-marking and verbal auxiliary morphology. Thus, case-marking for subjects of intransitive verbs and objects of

$$absolutive = \begin{cases} subject\ \ of\ \ intransitive\ \ verbs \\ object\ \ of\ \ transtive\ \ verbs \end{cases}$$
$$ergative = subject\ \ of\ \ transitive\ \ verbs$$

Figure 1: Two cases in Basque, and their respective syntactic functions depending on the transitivity of verb.

transitives and their morphological cross-reference within the verbal agreement auxiliary are identical and different from subjects of transitive verbs. The case-marking of transitive subjects is the ergative case (-ak when singular,-ek when plural). The case-marking of intransitive subjects and objects of transitives is the absolutive case (-a when singular, -ak when plural). Figure 1 summarizes the functions for each of these case-markers. And the following examples illustrate those ambiguities.

(2) *Etiketatzaile**a*** *agertu* *da.*
Tagger-**abs-singular** showed up intransitive-auxiliary.
The tagger-**subj** showed up.

(3) *Etiketatzaileak* *erlazio**a*** *desanbiguatu* *du.*
Tagger-erg-singular relation-**abs-singular** disambiguate transitive-auxiliary.
The tagger-subj has disambiguated the relation-**obj**.

When an element is in the absolutive case, its function (subject or object) is ambiguous, that is, case-marking by itself does not tell whether the element is a subject or an object; it is the transitivity of the verb, which in finite sentences appear to be lexicalized in the auxiliary, along with the case marking which makes disambiguation possible. Examples 2 and 3 show that elements bearing the absolutive case can be either objects or subjects depending on the transitivity present in the auxiliary. In these examples, the ambiguity is resolved with the information made explicit by the auxiliary, but there are exceptions.

(4) *Erlazio**a*** *erortzean* *gertatu* *zen.*
Relation-**abs-singular** when-dropped happened transitive-auxiliary.
It happened when the relation-**subj** dropped.

(5) *Erlazio**a*** *ikustean* *gertatu* *zen.*
Relation-**abs-singular** when-seen happened transitive-auxiliary.
It happened when the relation-**obj** was seen.

In the case of infinitive verbs, which do not have auxiliaries and thus do not explicitly mark for transitivity, the syntactic function of absolutive noun phrases is ambiguous. Examples 4 and 5 show two sentences where *erlazioa* is either subject or object (respectively) depending on the subcategorization information of the verb (drop or see, respectively).

(6) *Sukaldari**ak*** *egin* *ditu.*
Cook-**erg-singular** make transitive-auxiliary.
The cook-**subj** made it.

(7) *Opil**ak*** *egin* *ditu.*
Cake-**abs-plural** make transitive-auxiliary.
(S)he made the cakes-**obj**.

(8) *Erlazio**ak*** *ikusita,* *ezin* *dut* *asmatu.*
Relation-**abs-plural** seeing, not transitive-auxiliary figure-out.
Seeing the relation-**obj**, I can't figure it out.

Another source of ambiguity arises from the ambiguous morphological marker **-ak**, which can mean absolutive plural or ergative singular. Basque is a 3-way pro-drop language (Ortiz de Urbina, 1989), and thus subjects and objects can be elided (note the difference with English, where there must be always a subject). This means that in Basque we can have sentences like 6 and 7 above, where the object (subject in example 7) does not surface. Note also that position does not help disambiguating subjects and objects, as Basque is a relatively free-word order language. These two examples are syntactically ambiguous, and can only be disambiguated using semantic information, that is, cooks tend to be the subjects of transitive verbs and cakes tend to be objects of transitive verbs.

Both sources of ambiguity can occur together. Example 8 shows an example where the verb is infinitive and there is an ambiguous marker -ak, making the dependent (*Erlazio-ak*) ambiguous between absolutive/ergative and subject/object interpretations.

These ambiguous instances are very common in Basque, making up to 22% of all objects and subjects in Basque Dependency Treebank (cf. Section 5.1).

3 Resources

This section starts by describing Basque Dependency Treebank (Aduriz et al., 2003) which we used to extract potentially ambiguous occurrences and evaluate our methods. Next, it presents the methods to learn subcategorization information for Basque verbs. We finally review Malt parser.

3.1 Basque Dependency Treebank

Basque Dependency Treebank has more than 150,000 tokens, distributed in 11,125 sentences coded in CONLL-X format (Aduriz et al., 2003). Each token (dependent) is represented by the following information: index, word form, lemma, syntactic category (part of speech) and subcategory, morphological features corresponding to the different markers attached to the lemma, index of the head and syntactic relation between the dependent and the head. Example 9 shows a sentence from the treebank. Note the first, second and fourth tokens (nouns (ize) and determiner (det) respectively), which show inesive (ine), ergative (erg) and absolutive (abs) markers in singular (sg) and plural (pl). The verb (adi) shows perfective aspect (buru) and participle (part) form, and the auxiliary (adl) appears in the past form (b1), agreeing with a 3rd person singular ergative (hark) and a 3rd person plural absolutive (haiek). The corresponding treebank file is shown in Figure 3.

index	wordform	lemma	category	subcategory	morp.	head	relation
1	Martxoan	martxo	ize	arr	ine sg	6	ncmod
2	Millarrek	Millar	izb	-	erg sg	5	ncsubj
3	gurpil	gurpil	ize	arr	-	5	ncobj
4	guztiak	guzti	det	oro	abs pl	3	detmod
5	puskatu	puskatu	adi	sin	part buru	0	root
6	zituen	*edun	adl	-	b1 hark haiek	5	auxmod
7	.	.	punt	-	_	6	punc

Figure 2: Treebank file in CONLL-X format corresponding to example 9. See text for explanations of tags.

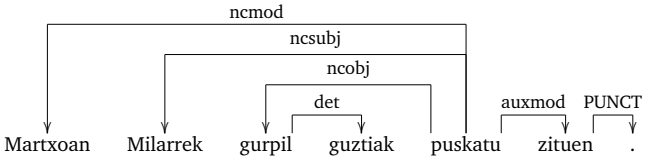


Figure 3: Dependency graph corresponding to example 9.

(9) *Martxoan* *Millarrek* *gurpil* *guztiak* *puskatu* *zituen.*
March-ine-sg Millar-erg-sg wheel all-abs-pl break auxiliary.
Millar broke all the wheels in March.

3.2 Acquisition of verbal subcategorization information

Verbal subcategorization information was extracted from 4 different sources: a subcategorization dictionary built from monolingual Basque corpus, web queries, monolingual English corpus, and a traditional monolingual Basque dictionary.

The subcategorization dictionary was obtained from Basque monolingual corpus, initially built with the purpose of making attachment decisions for a shallow parser on its way to full parsing (Aldezabal et al., 2002). For each of the 2,571 verbs this dictionary lists information about transitivity of the verb, noun[1]-case-verb triples or noun-case-verb-auxiliary quadruples and estimated frequency of each.

This dictionary was automatically built from raw corpora, comprising a compilation of 18 months of news from *Euskaldunon Egunkaria* (a journal written in Basque). The size of the corpus is around 780,000 sentences, approximately 10M words. From the 5,572 different verb lemmas in the corpus, the subcategorization dictionary was compiled for the 2,751 verbs occurring at least 10 times. The corpus was parsed by a chunker (Aduriz et al., 2004) which includes both named-entity and multiword recognition. The chunker uses a small grammar to identify heads, postpositions and verb attachments of NPs and PPs. The grammar was developed

[1]To simplify we just mention nouns, but there are also adjectives, determiner etc, anything that could be the head of a DP or CP.

based on the fact that Basque is a head-final language and it includes a distance feature as well. Phrases were correctly attached to the verb with a precision of 78%. Note that the auxiliary verb in Basque allows to unambiguously determine the transitivity of the main verb. The information captured for each verb corresponds to noun-case-verb triples and the noun-case-verb-auxiliary quadruples.

The second source is an English monolingual corpus. The assumption here is that the subject-object relation is stable when translating across languages, that is, if an element-verb relation is labeled as a subject relation in English it will also be that way in its Basque translation. This is a strong assumption, and we expect it to work better with certain verbs (e.g. activity or achievement verbs) than others (e.g. static verbs), but we did not make any distinction so far. In this approach, for each Basque ambiguous element-verb pair we collected frequencies over unambiguous English examples acquired from automatically parsed English data. We used the BNC corpus parsed with the RASP parser (Briscoe et al., 2006) containing 47,145,584 syntactic relations, where 10,447,129 are verb-noun dependency relations. The method has the following steps:

1. Translate the dependent lemma and the verb lemma using a bilingual dictionary.
2. Build all possible translation pairs.
3. Collect frequencies of each pair in the English corpus, depending on the label (subject or object) assigned by the English parser.

The third source is the result of directly querying the web. The web can be seen as a vast corpus (Bansal and Klein, 2011; Nakov, 2007; Lapata and Keller, 2005) where, in principle, we have bigger chances of finding low frequency combinations not found with the monolingual or crosslingual approaches presented so far. The following steps were pursued:

1. Obtain the lemma of the ambiguous element, create all possible subject and object unambiguous inflected forms using a language generation tool, that is, lemma+ergative+plural and lemma+absolutive+singular pairs.
2. Obtain the three different inflected forms of the verb (verb+future aspect marking, verb+perfect aspect marking and verb+ imperfect aspect marking) using the same generation tool.
3. Generate the corresponding different transitive and intransitive auxiliaries as well. It is not feasible to use all possible transitive/intransitive auxiliaries because the query number would explode so we took into account the 20 most frequent forms.
4. Construct all possible element+case+verb+auxiliary quadruples. For each element-verb candidate we get approximately 60 quadruples,
5. Search Google and collect hits.

Unfortunately Google does not recognize documents in Basque as a separate language. Some authors (Leturia et al., 2008) add certain common Basque words to the query, in order to reduces the number of texts in other languages returned by Google. We solved the problem using another heuristic, restricting to documents not in Spanish nor in English. The first one because Basque borrows vocabulary from Spanish and several times Basque texts are wrongly tagged as Spanish texts, and the other because of the same reason plus the fact that is the most common language on the web. Variability on the web does not cause a problem in this case because all the searches concerning each element-verb candidate are performed at the same time.

The last source is a traditional dictionary (EH dictionary, (Sarasola, 1996)), where each verbal entry carries information on the transitivity of the verb. As each verb usually shows more than one sense, we just considered the first sense. This dictionary uses 7 different markers to capture transitivity: du/da, du, du/dio, dio, da, zaio, da/zaio. For instance, du/da represent verbs that can appear in transitive or intransitive contexts, such as inchoative verbs like *break* that show a transitive/inchoative alternation: Leioa-subj apurtu da (The window-subj broke) or Mikelek-subj lehioa-obj apurtu zuen (Mikel-subj broke the window-obj).

4 Feature Space

In this section we will try to collect the information that we deemed was relevant to disambiguate subjects and objects. Each piece of information configures a separate feature. Our feature space F encodes heterogeneous information representing each candidate element-verb.

The features are presented grouped into sets depending on the nature and source of each one. A value close to 1 means that there is evidence for disambiguating to subject. A value close to 0 means that the feature leans for object. In some cases the feature does not predict anything.

Features related to subcategorization information

These features are based on the information mined from each source of subcategorization information, as presented in the previous section.

Subcategorization dictionary (SubcatDict)

- *TransCase(SubcatDict)*: The probability of the element to be a subject depends on the probability of the verb to be transitive $P(TransCase)$ and the actual case marking assigned by the morphological analyzer to the ambiguous element. $P(TransCase)$ is estimated from the SubcatDict, counting the occurrences of the verb as transitive and intransitive. If the case of the element is ergative and $P(TransCase)$ is bigger than 0.5, then this feature takes the value of $P(TransCase)$, that is, the feature will lean towards a subject interpretation. If the case is the absolutive, then the feature will lean towards being a subject if the verb is intransitive, or to object otherwise. The value of the feature is encoded according to the following formula.

$$\textit{TransCase(SubcatDict)} \begin{cases} P(TransCase) = \frac{\#trans}{\#trans+\#intrans} & case = erg \,\&\, P(TransCase) > 0.5 \\ 1 - P(TransCase) & case = abs \,\&\, P(TransCase) < 0.5 \\ 0 & case = abs \,\&\, P(TransCase) > 0.5 \\ none & otherwise \end{cases}$$

- *NCaseV(SubcatDict)*: The probability of the element to be a subject is related to the tendency of the element to bear the ergative case with that verb in the corpus (*P(Erg)*), independently of whether the verb shows transitive tendency. Here we do not consider the actual case marker assigned by the morphological analyzer, but the tendency of the element to bear ergative case when occurring with the verb.

$$\textit{NCaseV(SubcatDict)} \begin{cases} 1 & P(TransCase) > 0.5 \,\&\, P(Erg) > 0.5 \\ 0 & P(TransCase) < 0.5 \,\&\, P(Erg) < 0.5 \\ none & otherwise \end{cases}$$

- *NCaseVAux(SubcatDict)*: The probability of the element to be a subject is related to the tendency of the element to appear in a subject configuration, that is to say, to bear the ergative case with the verb appearing with a transitive auxiliary, or to bear absolutive case with the verb appearing with an intransitive auxiliary. This is estimated as described in the following formula.

$$NCaseVAux(SubcatDict) \begin{cases} \frac{\#(n+abs+v+intransAux)+\#(n+erg+v+transAux)}{\#(n+case+v)} & \#(n+case+v)>0 \\ none & otherwise \end{cases}$$

Web as a corpus (Web)

- Similar to the features for the SubcatDict, we can estimate the same features using the web queries explained in the previous section. This yields three new features: *TransCase(Web)*, *NCaseV(Web)* and *NCaseVAux(Web)*.

English corpus (BNC)

- *Subj(BNC)*: The value is 1 if the element shows a tendency of being a subject in the English corpus, 0 if the tendency is to object and none if it could not be translated or if it was not found in the English corpus.

Dictionary (Dict)

- *TransCase(Dict)*: This feature corresponds to the combination of two informations. The value is 1 if according to the dictionary the verb is transitive and the case is ergative, or if the verb is intransitive and the case is absolutive. The value is 0 if the verb is transitive and the case is absolutive, and none if transitivity could not be established in the dictionary.

Features related to morphological and syntactic information

Here we group weaker indications, as follows.

- *AspectCtrl*: The value is 1 if the verb is an aspectual or control verb such as *begin, end, stop, quit*. Aspectual verbs that occur a verb in the infinitive, force the verb in the infinitive to miss the surface subject. For example, *I started knowing you*, there cannot be a surface subject for *know* because *start* is an aspectual verb and thus both verbs share the same subject. Control verbs like *want* or *expect* act in Basque quite similarly to aspectual verbs In short, when the head of the verb is an aspectual or control verb, the element would tend to be the object, and thus feature will be 0, and none otherwise.
- *Preverb*: the value is 1 if the element appears in the pre-verbal position, as this is a sign of being a subject.
- *Inf*: 1 if the verb appears to be an infinitival, bare-infinitival, infinitival with a relative marker, infinitival nominalization, or in a finite form, that is to say, with an auxiliary, and finite with relative marker, respectively.
- *Erg*: The value is 1 if the case born in the noun phrase where the element is located is ergative, 0 otherwise. Remember from 2, that the head of a noun phrase does not necessarily bear the case marking, which is found in the last constituent of the noun phrase . We applied feature propagation in order to recover the case.
- *-ak*: The value is 1 if the element bears the ambiguous -ak morpheme.

- *Sing*: The value is 1 if the number of the element is singular, 0 otherwise.
- *Entity*: The value is 1 if the element starts with a capital letter and is not in the first position in the sentence. Since the morphological analyzer does not implement any entity recognition, this is an heuristic to code possible entities.

5 Experimental setup

In this section we review the method to create the gold standard, followed by our disambiguation method, the method to train malt parser, and the results of the experiments.

5.1 Creating the gold standard

The gold standard comprises ambiguous instances of noun phrases, which can either be subject or object. The syntactic structure and gold label is taken from Basque Dependency Treebank. In order to make the setting realistic the morphological tags are taken from an automatic morphological disambiguation tagger for Basque (Aduriz et al., 2000). Note that this tagger is conservative and does not always return a single tag. In those cases we use the first tag, as customary with most parsers. The accuracy of this tagger when selecting ergative and absolutive is 87%.

The procedure to detect ambiguous instances is the following. We first look up the verbs in the corpus. Depending on the finiteness of the verb, we have two cases:

1. If the verb has an auxiliary annotated as finite by the morphological analyzer, then the agreement features in the auxiliary verb resolve all ambiguities, except in some cases with -ak, which can be morphologically ambiguous between ergative singular or absolutive plural, and thus ambiguous between subject and object (respectively). To be more specific, if the auxiliary verb shows agreement with a singular ergative and a plural absolutive (both occurring with -ak) then both dependents could be either subject or object.
2. If the verb is tagged as infinitive, then if the verb has a dependent with the absolutive case, the dependent is ambiguous between subject and object.

The dependents of the verbs are looked up in the treebank, extracting the head and the case marking. The head is given as the root of the dependent, and the case marking of the dependent is given by the last token under the dependent.

Detecting dependents is a difficult task for parsers. So we filter out those dependents which could be difficult for current parsers, as follows. If the sentence contains a single verb, we then check all dependents of the verb. If the verb has multiple verbs, we then need a clause delimiter to identify which phrases are dependents of which verbs. We use a simple heuristic based on the fact that Basque is head-final: all words before the first verb are assigned to that verb, and the rest are assigned to the second verb. This is a strong baseline for any parser, as it attains 86% accuracy in a study of our own.

The above method to extract ambiguous dependents yields 4,525 ambiguous dependents in 3,617 sentences, that is, 22% of all subjects and objects in the treebank, with one ambiguous instance every 33% of sentences.

Note that if we had searched for ambiguous instances using the gold morphological tags in the treebank, we would find 4,400 subject-object ambiguous relations. This smaller amount is

due to errors by the morphological analyzer, as it incorrectly tags some ergative or absolutive cases, or certain auxiliaries as main verbs, considering the sentence as an infinitive sentence and therefore ambiguous with respect to subject-object. Note that the real ambiguity faced by a parser is closer to that of the automatic analysis.

5.2 Methods

We performed a machine learning experiment to examine the impact of the use of those lexical features to solve subject-object ambiguity in Basque. We used Support Vector Machines (Chang and Lin, 2011) with Radial kernels tuning C and G parameters over the training set using cross-validation to find their best values. The 4,525 ambiguous subject-object relations in the treebank were split into training and testing sets in a proportion of 50%. We also evaluated each feature on its own, and also, used an ablation procedure, learning the classifier with all features but one.

5.3 Malt parser

We chose Malt parser (Nivre et al., 2007) to compare our results with. This parser is a history-based deterministic dependency parser, which using the input and a stack and through 4 main actions, *shift* moving a token from the input to the stack, *left-arc* adding an arc from the token on the input to the token on the top of the stack, removing the token from stack, *right-arc* adding an arc from the token on the top of the stack to the token on the input, moving the token of the input onto top of stack and *reduce* removing a token from the top of the stack. The action is chosen according to a machine learning classifier using a variety of features including the stack, the input and past history. This way, dependency tree gets built in one single pass and in linear time. We used version 1.4, and the configuration was selected using the optimization developed by Nivre and Ballesteros Ballesteros and Nivre (2012).

Maltparser has to face a wider range of ambiguity that our classifier, as in some cases a phrase with the absolutive case can play syntactic functions other than subject or object. In order to make comparison to our classifier fair, we substituted all other tags returned by Malt parser with *ncobj*, the most common tag (around 75% compared to 25% for ncsubj).

5.4 Results

Table 1 shows the results of some features evaluated independently over the training set. For the sake of brevity, we only show those features with an accuracy over 60% are displayed. The baseline consists of assigning always the object tag to any ambiguous element, since it is the most frequent tag (75%). We measure accuracy on finding both subjects and objects, but we are also interested in measuring the performance of the system for detecting subjects since this is the most difficult task. Accuracy is the number of correctly tagged elements over all elements. Precision on subjects is the number of correctly recognized subjects divided by all elements tagged as subject by the system. Recall on subjects is the number of correctly recognized subjects divided by the total number of subjects in the gold set. F1 is the harmonic mean between precision and 1.

The table shows that the *Erg* feature performs best. This feature relies in the tag assigned by the automatic morphological analyzer. TransCase(SubcatDict) provides the second best result in terms of accuracy, still over the baseline, beating all the others in F1 and recall for subjects.

Feature	acc (sbj+obj)	prec (sbj)	rec (sbj)	F1 (sbj)
Baseline	75.29	00.00	00.00	00.00
TransCase(SubcatDic)	76.99	82.58	**74.17**	**78.15**
NCaseV(SubcatDic)	72.21	51.50	48.33	49.86
TransCase(Web)	60.10	80.94	57.47	67.21
NCaseV(Web)	69.21	22.71	19.16	20.78
TransCase(Dict)	60.31	**83.63**	50.26	62.79
Preverbal	62.09	17.93	17.93	17.93
Erg	**86.06**	50.26	50.26	50.26

Table 1: Results on ambiguous elements in the training corpus for each feature evaluated independently. Note that we only with accuracies over 0.6.

Feature	acc	prec(sbj)	rec(sbj)	F1(sbj)
Baseline	75.29	00.00	00.00	00.00
All features	**89.62**	**86.34**	**68.89**	**76.63**
¬SubcatDict	88.23	84.98	63.62	72.76
¬Web	88.32	83.94	65.20	73.39
¬BNC	88.23	84.49	64.14	72.93
¬Dict	87.66	86.25	59.57	70.47
¬SubcatInf	86.06	88.27	**50.26**	70.47
¬CaseNum	**85.28**	**77.64**	56.77	**65.58**
¬NCaseV(Aux)*	87.84	83.84	62.91	71.88

Table 2: Results using 10-fold cross-validation on ambiguous elements in the training corpus. The first lines correspond to the baseline and full classifier, and the rest present the results of feature ablation experiments. Best results and worst results in each column in bold.

Note that the performance of some features suffers from the fact that they only apply to a subset of the elements. In fact, *Erg* is the only one which applies to all.

Table 2 shows the results when training an SVM with Radial kernel and 10 fold cross-validation (All features line). Those results beat the baseline and individual features by a large margin. When compared with the best individual feature (*Erg*) the difference in accuracy is smaller, but note that the classifier is better on detecting subjects, showing that its output is better balanced.

The table also shows the results of feature ablation. Features were grouped by their source (SubcatDict, Web, BNC, Dict) or by the linguistic nature of the information they carry, independently of the source. For example, SubcatInf in Table 2 represents all subcategorization information: TransCase(SubcatDic), NCaseV(SubcatDic), NCaseVAux(SubcatDic),TransCase(Web), NcaseV(Web), NCaseVAux(Web), Subj(BNC) and TransCase(Dict). The highest loss in overall accuracy is for CaseNum features, but all features cause a performance drop when removed. This shows that the features are complementary. The loss in F1(sbj), whenever ablation of any kind is carried out, confirms this fact. The highest loss in precision and F1 also occur when information about case and number are left aside. And the highest loss in recall corresponds to

	acc	prec(sbj)	rec(sbj)	F1(sbj)
All features	89.33	82.48	71.74	76.74
MALT	86.72	76.82	65.69	70.82

Table 3: Final results for ambiguous elements on the test set.

	LAS	UAS		prec	rec	F1
MALT	83.17	83.08	sbj	71.57	75.01	73.24
			obj	76.36	73.61	74.95
MALT Post-processed	83.52	83.08	sbj	72.11	75.52	73.77
			obj	81.10	74.39	77.60

Table 4: Final results over all dependencies in test set.

the elimination of subcategorization information.

Finally we run our classifier with all features in the test set, as shown in Table 3. The performance obtained is comparable to that in the train set using cross-validation. The results of Malt are lower both in accuracy and F1 over subjects, with a statistical significant difference (p-value < 0.005). Note that the error reduction is 19.64%.

The above results show that our approach is competitive over Malt for the subset of ambiguous subject-object relations. In order to show that our system can make a relevant difference over the overall performance of a parser, we corrected the output of Malt parser with the result of our classifier and evaluated over all dependencies. Table 4 shows the usual UAS (Unlabeled Attachment Score) and LAS (Labeled Accuracy Score) scores for both MALT and the post-processed MALT. Of course, the unlabeled score is the same for both, as we just changed some labels. The improvement in LAS is of 0.35 absolute points, statistically significant (p-value < 0.00009). In addition we also show the performance over objects and subjects. Note that the post-processed version improves both object and subject recognition.

6 Related Work and Discussion

As mentioned in the introduction, this work is framed following a parse correction strategy. In parsing, not all ambiguities show the same complexity, and not all the languages behave the same way with respect to the distribution of the ambiguities. In English, for example, prepositional phrase attachment (PP-attachment for short) ambiguity traditionally stirred interest since (Hindle and Rooth, 1993; Ratnaparkhi, 1998), among others, for being both common and difficult to solve. These seminal works presented PP-attachment resolution in isolation, with no evaluation over full sentences or integration with a parser or with the results of a parser. With the proliferation of statistical parsers the attention moved from solving specific ambiguities to treat ambiguities as a whole. Statistical parsers learned from treebanks, though, make it difficult to reach any conclusion on what is the relevant information for resolving specific ambiguities, and whether those need to be encoded explicitly in treebanks.

Focusing on a relevant ambiguity is helpful to achieve a better understanding of the intricacies of parsing structures. Along these lines, we find two main approaches, that of parsing correction, or that of transforming and enriching the treebank with additional information. Work in parse correction consists on the creation of corrective models to solve difficult ambiguities, such as PP-attachment ambiguity in English. Thus correction occurs as a post-process to parsing,

replacing the output of the parser (labels or attachments) with alternatives obtained in an independent classification process.

The literature differs on the languages and the criteria used to choose the target information to be corrected. Hall and Novak Hall and Novák (2005) worked on Czech. They highlight the problem of projectivity as particularly problematic when parsing free word-order languages, such as Czech, due to the frequency of sentences with non-projective constructions. They present a corrective model which recovers non-projective dependency structures by training a classifier to select correct dependency pairs from a set of candidates obtained from parses generated by an automatic parser. In this case, the baseline parsers were Collins (2000) and Charniak (Charniak and Johnson, 2005), and the authors showed an improvement on dependency accuracy from 81.6% to 82.8% and from 84.4% to 85.1% (respectively).

Kilian and Menzel Foth and Menzel (2006) developed a classifier to solve PP-attachment ambiguity in German, and integrating the classifier into a rule-based parser. They report an error reduction of 14% and an improvement of overall attachment accuracy from 89.3% to 90.6%. Kluber, Ivanova and Klett Kübler et al. (2007) dealt with English PP-attachment ambiguity. They used the MaltParser (Nivre et al., 2007), showing an overall labeled parser accuracy improvement from 86.2% to 86.5%, and more precisely an improvement in unlabeled attachment from 71.8% to 77.4% in the case of prepositions. Henestroza and Candito (Anguiano and Candito, 2011) focus on PP-attachment and coordination as being difficult attachment ambiguities for French, presenting an improvement from 89.78% to 90.47% over MaltParser, and an improvement from 91.04% to 91.36% over MSTParser.

Attardi and Ciaramita Attardi and Ciaramita (2007) worked on English and Swedish. They show a slightly different methodology in that they do not focus on a particular ambiguity. They collect parsing errors by comparing correct parse trees with incorrect trees produced by the base parser on a training corpus. From those, they define a learning task where the input is a set of features extracted at each node in the parse trees produced by the parser on the training corpus whose output is a set of revised decisions, allowing correction. Their analysis was based on DeSR (Attardi, 2006), a shift-reduce parser.

Hussain and Agrawal Husain and Agrawal (2012) make an exhaustive analysis of the parsing errors committed by MaltParser over the Hindi Dependency Treebank. Hindi, as Basque, is a MoR-FWO language, and ergative as well. They identified that 50% of the errors were related to verb argument structure. The relevant information to avoid the errors was either contained in the treebank in a way that was difficult to manage by the parser, or had to be added to the Treebank, enriching it. They pursued two different experiments on tree-transformation and enrichment of the treebank using linguistic information from a dictionary and VerbNet. They conclude that only tree transformations lead to improvements, while enriching the treebank seems to have no effect whatsoever. In related work, (Husain et al., 2010) analyze the importance of different linguistic features over two dependency parsers on Hindi. They conclude that case marking and several verbal features such as tense, aspect and modality are the most informative. This is in contrast with our findings, where case is important, but also transitivity and subcategorization models.

In a different tone, Atterer and Schütze Atterer and Schütze (2007) are critic on some parse correction experiments. They deem parse correction as being unrealistic because it relies on using the treebank as an oracle to select the ambiguous candidates to be evaluated. They argue that, in contrast to the use of gold morphological analysis from the treebank to select ambiguity cases, parsers do not have access to those gold annotations at parsing time. They also argue that,

due to wrong attachment decisions, the parser might miss some ambiguous head-dependent relations that were mined from gold standard treebanks. To avoid these inconveniences when selecting the ambiguous candidates in our work, we used the morphological tags assigned by a morphological analyzer. Furthermore, we used a positional heuristic for assigning dependents to verbs, leaving aside the elements that could be potentially problematic for a parser as explained in 5.1. This way, we find 2303 ambiguous candidates in the training set, compared to 2338 if we would use the output of Malt parser. The intersection between the two amounts to 2040. In order not to penalize MaltParser on those non-coincident candidates, we assigned the object relation to those cases.

An important argument in favor of parse correction is that it is parser-agnostic, that is, our classifier is an independent module which disambiguates certain difficult ambiguities, and its results can be replaced over the result of any parser. In this paper we report on the correction over MaltParser for Basque(Bengoetxea et al., 2011), but in the future we plan to use our system to correct the output of a rule-based parser for Basque (Aranzabe et al., 2004).

7 Conclusions and Future Work

This works confirms the relevance of complex lexical information when solving syntactic ambiguity. More specifically, we have focused on subject-object ambiguity in Basque where it is one the main ambiguities, as it surfaces in 33% of the sentences. This problem is relevant to processing similar ambiguities in other morphologically rich languages. For instance, Urdu and Hindi also display subject-object ambiguity due to erg-abs markings, as well as null-case markings.

We have explored the impact of complex lexical information, including verbal subcategorization in the form of verbal transitivity and verb-noun-case or verb-noun-case-auxiliary relations. In addition, we have studied several linguistically motivated features, and trained a supervised classifier. Our results show that all sources of lexical information and features employed contribute positively, proving their complementarity. In fact, our classifier is better than a state-of-the-art statistical parser trained for Basque, obtaining a statistically significant error reduction of 20% in the resolution of subject-object ambiguities. When used to correct the output of the parser, the improvement is small, albeit statistically significant, showing that the classifier impacts in the overall parser performance.

The analysis revealed that the most relevant pieces of information are the case carried by the dependent element and the transitivity of the verb. For the future we would like to study the similarities and differences with typologically related languages. In addition, we plan to incorporate some of the features into the treebank and statistical parsers.

Acknowledgements

This research is partially funded by the Ministry of Economy under grants TIN2009-14715-C04-01 (KNOW2 project) and TIN2009-14675-C03-01 (OPENMT-2 project).

References

Aduriz, I., Agirre, E., Aldezabal, I., Alegria, I., Arregi, X., Arriola, J. M., Artola, X., Gojenola, K., Maritxalar, A., Sarasola, K., and Urkia, M. (2000). A word-grammar based morphological analyzer for agglutinative languages. In *Proceedings of the 18th conference on Computational linguistics - Volume 1*, COLING '00, pages 1–7, Stroudsburg, PA, USA. Association for Computational Linguistics.

Aduriz, I., Aranzabe, M., Arriola, J., Atutxa, A., de Ilarraza, A. D., Garmendia, A., and Oronoz, M. (2003). Construction of a basque dependency treebank. In Nivre, J. and Hinrich, E., editors,

Proceedings of the Second Workshop on Treebanks and Linguistic Theories (TLT 2003), pages 201–204.

Aduriz, I., Aranzabe, M. J., Arriola, J. M., de Ilarraza Sánchez, A. D., Galletebeitia, K. G., Oronoz, M., and Uria, L. (2004). A cascaded syntactic analyser for basque. In Gelbukh, A. F., editor, *CICLing*, volume 2945 of *Lecture Notes in Computer Science*, pages 124–134. Springer.

Aldezabal, I., Aranzabe, M., Gojenola, K., Sarasola, K., and Atutxa, A. (2002). Learning argument/adjunct distinction for basque. In *Proceedings of the ACL-02 workshop on Unsupervised lexical acquisition - Volume 9*, ULA '02, pages 42–50, Stroudsburg, PA, USA. Association for Computational Linguistics.

Anguiano, E. H. and Candito, M. (2011). Parse correction with specialized models for difficult attachment types. In *EMNLP*, pages 1222–1233. ACL.

Aranzabe, M., Arriola, M., and de Ilarraza, D. (2004). Towards a dependency parser of basque. In *Proceedings of the Workshop on Recent Advances in Dependency Grammar, COLING'04 Workshop*, pages 49–56.

Attardi, G. (2006). Experiments with a multilanguage non-projective dependency parser. In *Proceedings of the Tenth Conference on Computational Natural Language Learning*, CoNLL-X '06, pages 166–170, Stroudsburg, PA, USA. Association for Computational Linguistics.

Attardi, G. and Ciaramita, M. (2007). Tree Revision Learning for Dependency Parsing. In Sidner, C. L., Schultz, T., Stone, M., and Zhai, C., editors, *HLT-NAACL*, pages 388–395. The Association for Computational Linguistics.

Atterer, M. and Schütze, H. (2007). Prepositional phrase attachment without oracles. *Computational Linguistics*, 33(4):469–476.

Ballesteros, M. and Nivre, J. (2012). Maltoptimizer: A system for maltparser optimization. In Chair), N. C. C., Choukri, K., Declerck, T., Doğan, M. U., Maegaard, B., Mariani, J., Odijk, J., and Piperidis, S., editors, *Proceedings of the Eight International Conference on Language Resources and Evaluation (LREC'12)*, Istanbul, Turkey. European Language Resources Association (ELRA).

Bansal, M. and Klein, D. (2011). Web-scale features for full-scale parsing. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1*, HLT '11, pages 693–702, Stroudsburg, PA, USA. Association for Computational Linguistics.

Bengoetxea, K., Casillas, A., and Gojenola, K. (2011). Testing the effect of morphological disambiguation in dependency parsing of basque. Dublin - Irlanda.

Briscoe, T., Carroll, J., and Watson, R. (2006). The second release of the rasp system. In *Proceedings of the COLING/ACL on Interactive presentation sessions*, COLING-ACL '06, pages 77–80, Stroudsburg, PA, USA. Association for Computational Linguistics.

Chang, C. and Lin, C. (2011). LIBSVM: a library for support vector machines. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 2(3):27.

Charniak, E. and Johnson, M. (2005). Coarse-to-fine n-best parsing and maxent discriminative reranking. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, ACL '05, pages 173–180, Stroudsburg, PA, USA. Association for Computational Linguistics.

Collins, M. (2000). Discriminative reranking for natural language parsing. In Langley, P., editor, *ICML*, pages 175–182. Morgan Kaufmann.

Dixon, R. (1994). *Ergativity*. Cambridge Studies in Linguistics. Cambridge University Press.

Foth, K. A. and Menzel, W. (2006). The benefit of stochastic pp attachment to a rule-based parser. In *Proceedings of the COLING/ACL on Main conference poster sessions*, COLING-ACL '06, pages 223–230, Stroudsburg, PA, USA. Association for Computational Linguistics.

Hall, K. and Novák, V. (2005). Corrective modeling for non-projective dependency parsing. In *Proceedings of the Ninth International Workshop on Parsing Technology*, Parsing '05, pages 42–52, Stroudsburg, PA, USA. Association for Computational Linguistics.

Hindle, D. and Rooth, M. (1993). Structural ambiguity and lexical relations. *Computational Linguistics*, 19:103–120.

Husain, S. and Agrawal, B. (2012). Analyzing parser errors to improve parsing accuracy and to inform tree banking decisions. *Linguistic Issues in Language Technology*, 7(1).

Husain, S., Mannem, P., Ambati, B., and Gadde, P. (2010). The icon-2010 tools contest on indian language dependency parsing. *Proceedings of ICON-2010 Tools Contest on Indian Language Dependency Parsing, ICON*, 10:1–8.

Kübler, S., Ivanova, S., and Klett, E. (2007). Combining dependency parsing with pp attachment. In *Fourth Midwest Computational Linguistics Colloquium*. Citeseer.

Laka, I. (1996). *A brief grammar of Euskara, the Basque language*. Euskal Herriko Unibertsitatea, Bilbao.

Lapata, M. and Keller, F. (2005). Web-based models for natural language processing. *ACM Trans. Speech Lang. Process.*, 2(1).

Leturia, I., Gurrutxaga, A., Areta, N., and Pociello, E. (2008). Analysis and performance of morphological query expansion and language-filtering words on basque web searching. In Chair), N. C. C., Choukri, K., Maegaard, B., Mariani, J., Odjik, J., Piperidis, S., and Tapias, D., editors, *Proceedings of the Sixth International Conference on Language Resources and Evaluation (LREC'08)*, Marrakech, Morocco. European Language Resources Association (ELRA). http://www.lrec-conf.org/proceedings/lrec2008/.

Nakov, P. I. (2007). *Using the Web as an Implicit Training Set: Application to Noun Compound Syntax and Semantics*. PhD thesis, EECS Department, University of California, Berkeley.

Nivre, J., Hall, J., Nilsson, J., Chanev, A., Eryigit, G., Kubler, S., Marinov, S., and Marsi, E. (2007). Maltparser: A language-independent system for data-driven dependency parsing. *Natural Language Engineering*, 13(2):95.

Ortiz de Urbina, J. (1989). *Parameters in the Grammar of Basque*. Dordrecht, Foris (Studies in Generative Grammar, 33).

Ratnaparkhi, A. (1998). Statistical models for unsupervised prepositional phrase attachement. In *COLING-ACL*, pages 1079–1085.

Sarasola, I. (1996). *Euskal Hiztegia*. Kutxa Fundazioa, Donostia.

Volk, M. (2006). How bad is the problem of pp-attachment?: a comparison of english, german and swedish. In *Proceedings of the Third ACL-SIGSEM Workshop on Prepositions*, pages 81–88, Stroudsburg, PA, USA. Association for Computational Linguistics.

Comparative quality estimation: Automatic sentence-level ranking of multiple Machine Translation outputs

Eleftherios Avramidis
German Research Center for Artificial Intelligence (DFKI)
Language Technology Lab
Alt Moabit 91c, 10559 Berlin, Germany
eleftherios.avramidis@dfki.de

ABSTRACT
A machine learning mechanism is learned from human annotations in order to perform preference ranking. The mechanism operates on a sentence level and ranks the alternative machine translations of each source sentence. Rankings are decomposed into pairwise comparisons so that binary classifiers can be trained using black-box features of automatic linguistic analysis. In order to re-compose the pairwise decisions of the classifier, this work introduces weighing the decisions with their classification probabilities, which eliminates ranking ties and increases the coefficient of the correlation with the human rankings up to 80%. The authors also demonstrate several configurations of successful automatic ranking models; the best configuration achieves acceptable correlation with human judgments (tau=0.30), which is higher than that of state-of-the-art reference-aware automatic MT evaluation metrics such as METEOR and Levenshtein distance.

KEYWORDS: Machine Translation, Quality Estimation, Ranking.

Proceedings of COLING 2012: Technical Papers, pages 115–132,
COLING 2012, Mumbai, December 2012.

1 Introduction

As machine translation (MT) comes closer to the everyday use, the need to predict the quality of machine-translated text is being of intense interest. For this reason, research has been focusing more and more on developing methods of assessing the translated content and deriving indications of translation performance in a real-time translation environment, without access to the correct (reference) translations.

Here we are focusing on a specific scenario: we need an automatic system able to rank several machine translation outputs for one given source sentence, according to their comparative quality. This kind of ranking, performed by human annotators, has been established as a practice for evaluating MT-output. Therefore, we attempt to perform "machine ranking", by employing Machine Learning approaches in order to imitate the human behaviour. This is done through a statistical classifier, which is trained given existing human ranks and several qualitative criteria on the text.

This idea can serve several existing applications in MT, as it touches the fields of Hybrid MT, System Combination and MT Evaluation. The applicability is even broader, as the approach presented is system-independent and relies on generic automatic analysis applied on any input containing sets of one source and several translation outputs.

2 Previous work

Quality Estimation is a rather recent aspect in research on Machine Translation. As a field, it tries to provide quality assessment on the translation output without the availability of reference translations. Previous work includes statistical methods on predicting word-level confidence (Ueffing and Ney, 2005; Raybaud et al., 2009b), correctness of a sentence (Blatz et al., 2004) and has been recently evolved into a regression problem (Specia et al., 2009; Raybaud et al., 2009a) for estimating correctness scores or correctness probabilities.

Whereas the aforementioned work has been focusing on estimating absolute measures of quality for a single output, our focus is on the comparative estimation of quality among several system outputs. In this direction, Rosti et al. (2007) perform sentence-level selection with generalized linear models, based on re-ranking N-best lists merged from many MT systems. Sánchez-Martínez (2011) uses only source-language information in order to build a classifier which chooses which machine translation system should be used in order to translate a sentence. As an application to statistical MT tuning, Hopkins and May (2011) improve the tuning MERT process by using the pairwise approach of ranking with a classifier. Others (Vilar et al., 2011; Soricut and Narsale, 2012) use machine learning for ranking the candidate translations and then selecting the highest-ranked translation as the final output.

A couple of contributions (Ye et al., 2007; Duh, 2008) introduce the idea of using ranking in MT evaluation, by developing a machine learning approach to train on rank data, though these are using reference translations and are only evaluated by producing an overall corpus-level ranking. METEOR, one of the state-of-the-art evaluation metrics (Lavie and Agarwal, 2007) also gets its components tuned over human rankings. Avramidis et al. (2011) do MT evaluation without references based on learned ranking, by using parsing features and Parton et al. (2011) also show a configuration of their metric which achieves good correlation with human judgments without any reference information, using target features produced by a language correction software.

Reported work has used various aspects of weighing or training over human ranking. Following

on a similar path, we are focusing on the ability of a mechanism to reproduce human preference rankings and compare its outcome with existing evaluation methods.

3 Methods

3.1 Problem description

As already introduced in Section 1, this work is aiming to a mechanism for ranking multiple translation outputs. In detail, the system is given one source sentence and several translations which have been produced for this sentence, with the use of many MT systems. The goal is to derive several qualitative criteria over the translations and use them to order the translations based on their quality, i.e. to *rank* them.

In this ranking process, each translation is assigned an integer (further called a *rank*), which indicates its quality relatively to the competing translations for the same source sentence. E.g. given one source sentence and n translations for it, each of the latter would get a rank in the range $[1, n]$. The same rank may be assigned to two or more translation candidates, if the translations are of similar quality (i.e. there is no distinguishable difference between them); such a case defines a *tie* between the two translation candidates.

It should be thereof clear that such a qualitative ordering does not imply any absolute or generic measure of quality. Ranking takes place on a sentence level, which means that the inherent mechanism focuses on only one sentence at a time, considers the available translation options and makes a decision. Any assigned rank has therefore a meaning only for the sentence-in-focus and given the particular alternative translation candidates.

Finally, one further assumption as part of the current problem specification is that our system is not bound to the MT systems providing the outputs. This means that the usually small number of alternative translations may derive from a bag of many more MT engines with different characteristics and internal behaviour. The systems are therefore seen as *black boxes* and their translation outputs are treated on a merely superficial level, i.e. without any further information of how they were produced. Thus, one assumption is that the source and translated text contain enough information for assessing translation quality, probably approaching the way the task would be perceived by a human annotator.

3.2 Machine learning on pairwise decomposition of ranking

The problem is hereby treated as a typical machine learning paradigm. Ranking is *learned* from a training material containing existing human rankings. The learning process results in a statistical model. This model can later reproduce the same task on unknown sentences or test data. Whereas the setup and the evaluation of the system takes place on a ranking level, for the core of the decision-making mechanism we follow the principle of going pairwise (Herbrich et al., 1999; Hüllermeier et al., 2008): ranking lists are decomposed to pairwise comparisons. Then, given one pair of translation candidates at a time, a classifier has to predict a binary decision on whether the first translation candidate is better than the latter.

In this context, we train one classifier for the entire data set. Each ranking of n candidate translations is decomposed to $n \times (n-1)$ pairs of all possible combinations of two system outputs with replacement. Each of the resulting pairs forms a training instance for the classifier. Each training instance provided to the classifier consists of a class value c and a set of features $(f_1, \ldots, f_n)$ regarding the translation quality/preferences. For the pairwise comparison of two translation candidates t_i, t_j with human ranks r_i and r_j respectively, the class value is therefore

set as:

$$c_{i,j} = \begin{cases} 1 & r_i < r_j \\ -1 & r_i > r_j \end{cases}$$

The approach of pairwise comparisons is chosen because it forms the machine learning question in a much simplified manner. Instead of treating a whole list of ranks, the classifier has to learn and provide a binary (positive or negative) answer to the simple question "*which of these two sentences is better?*". This also gives the flexibility of experimenting with many machine learning algorithms for the classification, including those which only operate on a binary class.

As explained in Section 3.1, ties may exist in the training material. Though, ties that appear on a pairwise level have been filtered out, since this does not add any useful information on the simple comparison explained above. This means that the pairwise comparisons of the tied outputs with the other outputs are not filtered out; only those between the two tied systems are. A further handling of ties by introducing a third class or a cascade of two classifiers would be a possibility to investigate, but we leave this aside for the moment in order to test the basic functionality given the most promising properties. As we will see later in this section, the ties are treated rather as an uncertainty of the system for either of the classes.

3.3 From pairs back to ranking

During the application of the statistical model on test data, data processing follows the same idea: The test instances are broken down to pairs of sentences and given to the classifier for a binary decision. Consequently there is a need to recreate a ranking list out of the binary pairwise classification decisions.

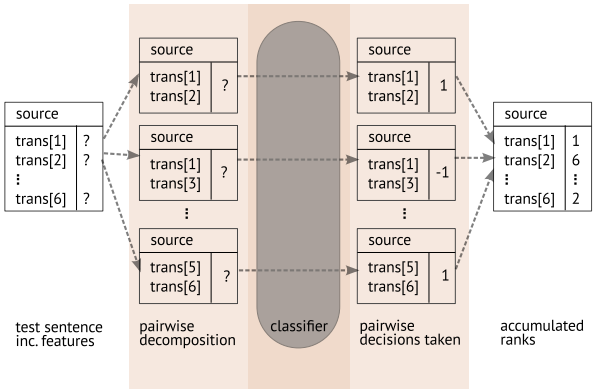


Figure 1: The application of the statistical model, through the pairwise decomposition (left) and recomposition (right)

3.3.1 Hard rank recomposition

The simplest way to go ahead with this is to sum up the decisions of the classifier. For a number of n systems, following the previous annotation, the rank r_i of translation t_i would be:

$$r_i = \sum_{j \neq i}^{n} c_{i,j}$$

The translation output which has "won" the most pairwise comparisons would get first on the list and then the outputs with less pairwise wins would follow accordingly (figure 1). We call this a *hard rank recomposition*, as only the binary decision of the classifier is taken into consideration upon summing up the predicted values.

3.3.2 Soft rank recomposition

One of the problems seen in previous work is that what we described here as a *hard rank recomposition* allows for the creation of ties. Indeed, it is intuitive that the classifier may predict an equal number of wins for two or more translation outputs and therefore generate a tie among them. This may also be intensified by the fact that the pairs have been generated in both directions, which would also result into a tie if the classifier is unable to distinguish the best out of two outputs but is forced to choose one of them.

However, the probabilistic set-up contains information which implies that not all classifier decisions are of "equal importance": statistical classifiers build their binary responses on a probabilistic basis. A translation output which has a number of wins with high certainty should be ranked higher than an output with an equal number of wins but more uncertain. One can therefore use the probability of each decision to weigh the sum described in Section 3.3.1. A translation output which has a number of wins with high certainty should be ranked higher than an output with an equal number of wins but more uncertain. This is thereof referred to as *soft recomposition*. This way, the rank r_i of translation t_i would be:

$$r_i = \sum_{j \neq i}^{n} p_{i,j} c_{i,j}$$

Since the probability $p_{i,j}$ is a long decimal in the range of [0,1] as opposed to a binary value, it is expected that it reduces the cases where two translation outputs end up with an equal sum.

3.4 Feature acquisition

Similar to the previous works on quality estimation, the source sentence and the corresponding translations are analyzed by several tools of linguistic analysis, in order to provide a set of features indicative of the translation quality. Since one of the goals is to not be bound to the systems participating (Section 3.1), these are *black-box* features, i.e. deriving solely from the text. The features used in this work fall into the following categories:

- **Count-based features**: count of tokens, average count of characters per token, count of unknown words
- **Parsing statistics**: One of the common issues that affect MT quality and acceptability is the grammaticality of the generated sentences. This is intense in many statistical systems

(particularly the ones following the phrase-based approach) since they treat the generation process in a rather shallow way. Most often language models are used within the MT systems in order to optimize the output for the highest probability of the consequent *n*-grams. As an additional measure of quality which can capture more complex phenomena (such as grammatical fluency, long distance structures, etc.) we include features derived from Probabilistic Context Free Grammars (PCFG) parsing (Petrov et al., 2006).

PCFG parsing operates by creating many possible tree parses for a given a sentence, forming an *n*-best list of parse hypotheses. These hypotheses are scored probabilistically, leading to the selection of the tree with the highest overall probability. We allowed an n-best list with a size of n=1000 and counted **the number of trees generated**. Although the *n*-best list reaches the limit for the majority of the sentences, some sentences have a smaller number of trees, which signifies less possible tree derivations, i.e. less parsing ambiguity, a feature which would be useful for our use.

Additionally, we extracted and included the basic parsing statistics of the **overall parse log-likelihood**, the **confidence for the best parse tree** and the **average confidence of all trees**.

- **Tree label counts**: In an effort to derive some adequacy features, we relied on the assumption of isomorphism; i.e. the fact that the same or similar grammatical structures should occur on both source and translation(s). Therefore, we counted the basic node labels of the parse tree, namely the NPs, VPs, PPs, verbs, nouns, sentences, subordinate clauses and punctuation occurrencies. The source and target equivalents of labels were manually matched so that their ratios could also be calculated. E.g. the failure to properly translate a Verb Phrase should be indicated by an improportional ratio.

- **Language checking**: Source and target sentences were subject to automatic rule-based *language quality checking* (Siegel, 2011), providing a wide range of quality suggestions concerning **style**, **grammar** and **terminology**, summed up in relevant quality scores.

- **Language-model probabilities**: Language models are also an indication of fluency, since they provide statistics on how likely the sequences of the words are for a particular language. Although Statistical MT systems are expected to already optimize over the language model, as mentioned above, other types of systems may still benefit from this features. This feature category includes the smoothed n-gram probability of the sentence.

- **Contrastive evaluation scores**: Each translation is scored with an automatic metric (e.g. Papineni et al., 2002), using the competitive translations as references. This has shown to perform well as a feature in similar tasks (Soricut et al., 2012).

Keeping the isomorphism assumption, an additional hint for the adequacy of the translation was applied for the features that are apparent in both source and target: The ratio of these features was calculated by diving the feature value of each one of the translation outputs with the respective feature value of the source.

3.5 Machine learning algorithms

The modular approach of the pairwise classification allowed the use of several machine learning algorithms as part of the system core.

- **Naïve Bayes** predicts the probability of a binary class c given a set of features

$$p(c, f_1, \ldots, f_n) = p(c) \prod_{n}^{i=1} p(f_i|c)$$

  $p(c)$ is estimated by relative frequencies of the training pairwise examples, while the probabilities $p(f_i|c)$ for the continuous features f are estimated with the locally weighted linear regression LOESS (Cleveland, 1979).

  Naïve Bayes has the drawback that it requires the features to be statistically independent. However, as an algorithm it can be trained pretty fast and scales well with large amount of data and big feature sets.

- The **k-nearest neighbour** (K-nn) algorithm performs classification to the closed training examples in the search space (Coomans and Massart, 1982). This algorithm does not have a priori assumptions about the distributions of the training data. Though, the method requires a choice for the number (k) of the nearest neighbors, which is problem-specific. For these experiments, we followed the common practice of setting the k equal to the square root of the number of training instances.

- **Logistic regression** tries to maximize a logistic function, whose values range between zero and one (Cameron, 1998). It was fitted using Newton-Raphson algorithm to iteratively minimize least squares error computed from training data (Miller, 2002), whereas Stepwise Feature Selection (Hosmer, 1989) was included. Logistic Regression generally performs better than the previous algorithms, though it has higher computational complexity and therefore its calculation is time-demanding, limiting the possibility to explore many experiment parameters.

3.6 Evaluation

3.6.1 Classification performance

As a classifier is the core part of the system, its robustness and ability to successfully take its binary decisions are indicative of the performance of the entire system. As a basic indications on the success of the learning process we compute **Classification Accuracy** (CA), as a result of cross-fold validation over the training set. This part is useful for evaluating the choice of the learning method and the feature set.

3.6.2 Correlation with human judgments

The system is tested on how well it can rank translation quality, compared to the ranking a human would do for this purpose. After building a model, the system is used in order to perform ranking on a test-set. This test-set has been excluded from the training data and provides human rankings which are hidden during the test classification, but are afterwards used for evaluating the success of the automatic process. The final goal is to measure the sentence-level correlation of the automatically produced rankings with the ones chosen by humans.

For the correlation measurement, we follow **Kendall's tau** coefficient (Kendall, 1938; Knight, 1966), which is suitable for measuring correlation between two ranking lists on a segment level: For every sentence, the machine-predicted and the human ranking are decomposed into

pairwise comparisons. Each automatic pairwise comparison is compared with the respective human comparison and it is counted as *concordant* or *discordant*, depending on whether these two comparisons agree or not. Tau is then given by the fraction:

$$\tau = \frac{\text{concordant} - \text{discordant}}{\text{concordant} + \text{discordant}}$$

with values that range between minus one and one, whereas the closer the value gets to one, the better the ranking is. The calculation follows the formula of the Workshop in Machine Translation (Callison-Burch et al., 2012), in order to be comparable with other methods: Pairwise comparisons with reference translations and pairwise ties in the human-annotated test-set are ignored. On the contrary, every tie on the machine-predicted rankings is penalized by being counted as a discordant pair.

We thereof present two versions of the tau:

- **Overall tau** (τ) where concordant and discordant counts from all segments (i.e. sentences) are gathered and the fraction is calculated with their sums
- **Average segment tau** ($\overline{\tau}_{\text{seg}}$) where tau is calculated on a segment level and then averaged over the number of sentences. This shows equal importance to each sentence, irrelevant of the number of alternative translations.

4 Experiment

4.1 Data sets

Both our training and test data were extracted from human-annotated data containing comparisons of the outputs of several German-to-English MT systems, as a result of the evaluation tasks run by the Workshops on Machine Translation (Callison-Burch et al., 2008, 2009, 2010, 2011), which have been freely available for further research. In the development phase, our training set consisted of the human rankings of years 2008, 2010[1], 2011 and the test-set from the year 2009. In order to re-assure that the system is not overfitting the development environment, the best systems were also tested upon a different set-up, where the human rankings of years 2008, 2009 and 2010 were used for training and the rankings from 2011 system combination task (2011c) were used for testing.

The provided data-sets contain human judgments organized in rankings of at most five sentence at a time. Therefore, the alternative translation outputs for one source sentence are often spanned along many 5-way rankings. For the test set, the multiple rankings of the same source sentence (produced by all available systems) were aggregated into one ranking, and the ties on both pairwise and ranking level were removed. One should also notice that repetitive human rankings of the same systems often disagree with each other, which signifies the existence of some noise in our data.

4.2 Implementation

PCFG parsing features were generated on the output of the Berkeley Parser, with the default grammars based on an English and a German treebank (Petrov and Klein, 2007). N-gram features were based on language models of order 5, built with the SRILM toolkit (Stolcke, 2002) on monolingual training material from the Europarl (Koehn, 2005) and the News (Callison-Burch et al., 2011) corpora. The *Acrolinx IQ*[2] was used to annotate source and target with language checking suggestions and provide style, grammar and spelling scores. The annotation process was organized with the Ruffus library (Goodstadt, 2010) and the learning algorithms were executed using the Orange toolkit (Demšar et al., 2004).

4.3 Strategy

The amount of features and learning options provide an exponential number of experiment parameters. However, in order to be able to draw a fair amount of conclusions in a decent amount of time, we followed an incremental approach: first, we devised some feature sets that have shown to perform well in previous work[3] and used them for learning and testing with the default parameters of all the available methods. Secondly, we repeated the experiments with variations of the most successful parameter set, e.g. by slightly modifying the features or adding promising new features. This approach may have stalled to local maxima, but it should suffice if it can provide a functioning system confirming the original idea.

[1] In all of the experiments we excluded the crowdsourced sentences contained in the set of 2010

[2] http://www.acrolinx.com (proprietary)

[3] We tried to come as close to the original feature set when not all features were technically available

4.4 Results

4.4.1 Searching for the best system

The search through different combinations of feature sets and classification methods is depicted in Table 1. Feature sets 2 - 5 derive from previous work (Soricut et al., 2012; Avramidis et al., 2011; Specia et al., 2012) and are explained in Table 3. Out of these, it appears that feature set 2 is the most successful one for this particular problem, providing a correlation which is acceptable to begin with. *K-nn* slightly outperforms Naïve Bayes.

Consequently, extensions to feature set 2 are considered for further experimentation. Feature set 2.1 provides an improved combination with logistic regression: It derives from the same annotation as feature set 2, with the difference that the features of the target had not been not divided with the features of the source, in order to provide a fixed ratio as a feature; instead, these features were given separately. Due to its power to do a logistic search, we could assume that this learner treats better the factors of the ratio if given separately.

Adding NP counts (feature set 2.2) did not show any improvement. Replacing parsing probability with spelling, grammar and style scores, achieves some improvement, particularly for Naïve Bayes, which has its highest coefficient here.

The most successful feature set is 2.4, which extends 2.1: Learned with logistic regression, it includes the number of unknown words, sentence length, the number of alternative parse trees, the count of VPs and the parse log-likelihood, but also additionally a contrastive METEOR score.

4.4.2 Improvement by soft recomposition

A basic contribution of this paper is the introduction of the soft recomposition of the ranks. This is obvious by reading Table 1 on the horizontal dimension: the soft recomposition achieves higher taus and significantly less ties for all the systems and particularly for the ones which show a positive correlation. In the best cases, using soft recomposition improves the correlation numbers by 40-80%.

4.4.3 Comparisons with state-of-the-art MT evaluation

Although our method uses no reference translations, it still maintains the notion of MT evaluation. Therefore, in lack of openly available competitors of its kind, it makes sense to compare its performance with automatic state-of-the-art MT metrics, to whom reference translations get available. Sentence-level smoothed-BLEU (Papineni et al., 2002), METEOR (Lavie and Agarwal, 2007) and Levenshtein distance (Levenshtein, 1966) were used. This comparison was also done on a set-up different to that of the development phase. The results (Table 2) show that even without references, the correlation of our best system with human judgment is at least 35% higher than that of the standard metrics.

model		CA	hard recomposition			soft recomposition		
feat	classifier		τ	$\overline{\tau}_{seg}$	ties	τ	$\overline{\tau}_{seg}$	ties
#1	kNN	57,0%	0,05	0,00	259	0,10	0,08	6
	Naive	57,8%	-0,03	-0,07	615	0,12	0,14	12
#2	kNN	**60,7%**	**0,10**	**0,12**	317	**0,18**	**0,22**	6
	Naive	**59,6%**	**0,12**	**0,11**	152	**0,17**	**0,18**	8
#3	kNN	55,8%	-0,06	-0,06	250	0,00	0,00	0
	LogReg	55,1%	0,06	0,04	29	0,06	0,04	23
	Naive	54,9%	0,02	0,05	30	0,02	0,05	26
#4	kNN	56,3%	-0,04	-0,02	261	0,04	0,05	0
	LogReg	55,0%	0,06	0,03	51	0,06	0,04	22
	Naive	55,2%	0,00	0,04	34	0,01	0,04	22
#2.1	kNN	58,4%	0,09	0,08	252	0,16	0,16	0
	LogReg	**61,5%**	**0,24**	**0,27**	72	**0,25**	**0,29**	24
	Naive	59,8%	0,16	0,15	194	0,21	0,21	20
#2.2	kNN	58,5%	0,06	0,08	233	0,12	0,14	0
	LogReg	**61,5%**	0,24	0,27	74	0,26	0,28	18
	Naive	59,6%	0,15	0,13	228	0,20	0,19	24
#2.3	Naive	**61,0%**	**0,22**	**0,26**	83	**0,23**	**0,28**	24
	LogReg	61,4%	0,24	0,26	60	0,25	0,27	31
#2.4	kNN	59,4%	0,06	0,08	249	0,12	0,16	0
	LogReg	61,3%	**0,26**	**0,28**	68	**0,27**	**0,30**	15
	Naive	60,8%	0,20	0,21	141	0,24	0,26	11

Table 1: Search of the most promising feature sets tested on the development test-set . Feature set #2 from previous work extended with additional features: Logistic Regression with feature sets #2.4 and #2.1 had the highest τ correlation with human rankings. Improvement by using *soft rank re-composition* (Section 3.3.2) is also illustrated

test-set	2009	2011c
SmoothBLEU	-0,23	-0,25
METEOR	0,20	0,12
Levenshtein	0,18	0,07
Machine Ranking (LogReg #2.4)	**0,27**	**0,24**

Table 2: Comparison of our best result with state-of-the-art reference-aware automatic metrics concerning correlation with human judgments (τ). The model is also succesfully tested in order to rank wmt2011-combo translation options (excluded from training)

#1	source:	avg. characters per word, tri-gram probability, count of tokens, NPs
	target:	parse log-likelihood, count of unknown words, ratio of VPs, ratio of PPs, NPs, verbs, ratio of tokens count (Specia et al., 2012)
#2	source:	count of unknown words
	target:	count of unknown words, tokens ratio, ratio of parse trees, ratio of VPs, ratio of parse log-likelihood (Avramidis et al., 2011)
#3	source:	count of unknown words, tokens, dots, commas, avg. characters per word, LM probability
	target:	contrastive-BLEU, LM probability (SVR model from Soricut et al., 2012)
#4	source:	count of unknown words, tokens, dots, commas, avg. characters per word, LM probability
	target:	contrastive-BLEU, LM probability (M5P model from Soricut et al., 2012)
#2.1	source:	count of unknown words, tokens, parse trees, VPs, parse log-likelihood
	target:	count of unknown words, tokens, parse trees, VPs, parse log-likelihood (same as #2 with no ratios)
#2.2	source:	count of unknown words, tokens, parse trees, VPs, NPs, parse log-likelihood
	target:	count of unknown words, tokens, parse trees, VPs, NPS, parse log-likelihood (same as #2.1 including NPs)
#2.3	source:	count of unknown words, tokens, parse trees, dots, commas, spelling score, grammar score, style score
	target:	contrastive-METEOR, count of unknown words, tokens, parse trees, dots, commas, spelling score, grammar score, style score
#2.4	source:	count of unknown words, tokens, parse trees, VPs, parse log-likelihood
	target:	contrastive-METEOR, count of unknown words, tokens, parse trees, VPs, parse log-likelihood (same as #2.1 with contrastive-METEOR)

Table 3: Description of the feature-sets used

5 Conclusion

Machine learning was successfully used as part of a mechanism which is able to perform preference ranking on alternative machine translation outputs. Correlation with human judgments indicates a success in building a mechanism which performs ranking, since even without access to reference information, its performance is higher than other state-of-the-art reference-aware metrics.

The fact that ranking was decomposed into pairwise decisions allowed the integration of several machine learning algorithms with positive results. The recomposition of a ranking from pairwise decisions was facing the problem of creating too many ties as a result of unclear and contradictory pairwise decisions. This was solved by weighing classification decisions with their prediction probabilities.

The best system uses logistic regression with a feature set that includes the number of unknown words, sentence length, a contrastive METEOR score, and parse statistics such as number of alternative parse trees, count of VPs and the parse log-likelihood.

Acknowledgments

This work has been developed within the TaraXŰ project, financed by TSB Technologiestiftung Berlin – Zukunftsfonds Berlin, co-financed by the European Union – European fund for regional development. Many thanks to Aljoscha Burchard, Hans Uszkoreit and David Vilar for their useful feedback, to Lukas Poustka for his technical help on feature acquisition and to Melanie Siegel for her support concerning the language checking tool.

References

Avramidis, E., Popovic, M., Vilar, D., Burchardt, A., and Popović, M. (2011). Evaluate with Confidence Estimation : Machine ranking of translation outputs using grammatical features. In *Proceedings of the Sixth Workshop on Statistical Machine Translation*, pages 65–70, Edinburgh, Scotland. Association for Computational Linguistics.

Blatz, J., Fitzgerald, E., Foster, G., Gandrabur, S., Goutte, C., Kulesza, A., Sanchis, A., and Ueffing, N. (2004). Confidence estimation for machine translation. In *Proceedings of the 20th international conference on Computational Linguistics*, COLING '04, Stroudsburg, PA, USA. Association for Computational Linguistics.

Callison-Burch, C., Fordyce, C., Koehn, P., Monz, C., and Schroeder, J. (2008). Further Meta-Evaluation of Machine Translation. In *Proceedings of the Third Workshop on Statistical Machine Translation*, pages 70–106, Columbus, Ohio. Association for Computational Linguistics.

Callison-Burch, C., Koehn, P., Monz, C., Peterson, K., Przybocki, M., and Zaidan, O. (2010). Findings of the 2010 Joint Workshop on Statistical Machine Translation and Metrics for Machine Translation. In *Proceedings of the Joint Fifth Workshop on Statistical Machine Translation and MetricsMATR*, pages 17–53, Uppsala, Sweden. Association for Computational Linguistics.

Callison-Burch, C., Koehn, P., Monz, C., Post, M., Soricut, R., and Specia, L. (2012). Findings of the 2012 Workshop on Statistical Machine Translation. In *Proceedings of the Seventh Workshop on Statistical Machine Translation*, pages 10–51, Montréal, Canada. Association for Computational Linguistics.

Callison-Burch, C., Koehn, P., Monz, C., and Schroeder, J. (2009). Findings of the 2009 Workshop on Statistical Machine Translation. In *Proceedings of the Fourth Workshop on Statistical Machine Translation*, pages 1–28, Athens, Greece. Association for Computational Linguistics.

Callison-Burch, C., Koehn, P., Monz, C., and Zaidan, O. (2011). Findings of the 2011 Workshop on Statistical Machine Translation. In *Proceedings of the Sixth Workshop on Statistical Machine Translation*, pages 22–64, Edinburgh, Scotland. Association for Computational Linguistics.

Cameron, A. (1998). *Regression analysis of count data*. Cambridge University Press, Cambridge UK; New York NY USA.

Cleveland, W. S. (1979). Robust locally weighted regression and smoothing scatterplots. *Journal of the American statistical association*, 74(368):829–836.

Coomans, D. and Massart, D. (1982). Alternative k-nearest neighbour rules in supervised pattern recognition. *Analytica Chimica Acta*, (138):15–27.

Demšar, J., Zupan, B., Leban, G., and Curk, T. (2004). Orange: From Experimental Machine Learning to Interactive Data Mining. In *Principles of Data Mining and Knowledge Discovery*, pages 537–539.

Duh, K. (2008). Ranking vs. Regression in Machine Translation Evaluation. In *Proceedings of the Third Workshop on Statistical Machine Translation*, pages 191–194, Columbus, Ohio. Association for Computational Linguistics.

Goodstadt, L. (2010). Ruffus: a lightweight Python library for computational pipelines. *Bioinformatics*, 26(21):2778–2779.

Herbrich, R., Graepel, T., and Obermayer, K. (1999). Support Vector Learning for Ordinal Regression. In *International Conference on Artificial Neural Networks*, pages 97 – 102.

Hopkins, M. and May, J. (2011). Tuning as ranking. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP '11)*, pages 1352–1362, Edinburgh, Scotland. Association for Computational Linguistics Morristown, NJ, USA.

Hosmer, D. (1989). *Applied logistic regression*. Wiley, New York [u.a.], 8th edition.

Hüllermeier, E., Fürnkranz, J., Cheng, W., and Brinker, K. (2008). Label ranking by learning pairwise preferences. *Artificial Intelligence*, 172(16-17):1897–1916.

Kendall, M. G. (1938). A New Measure of Rank Correlation. *Biometrika*, 30(1-2):81–93.

Knight, W. R. (1966). A computer method for calculating Kendalls tau with ungrouped data. *Journal of the American Statistical Association*, 61(314):436–439.

Koehn, P. (2005). Europarl: A parallel corpus for statistical machine translation. *MT Summit*, 5.

Lavie, A. and Agarwal, A. (2007). METEOR: An Automatic Metric for MT Evaluation with High Levels of Correlation with Human Judgments. In *Proceedings of the Second Workshop on Statistical Machine Translation*, pages 228–231, Prague, Czech Republic. Association for Computational Linguistics.

Levenshtein, V. (1966). Binary Codes Capable of Correcting Deletions and Insertions and Reversals. *Soviet Physics Doklady*, 10(8):707–710.

Miller, A. (2002). *Subset Selection in Regression*. Chapman & Hall, London, 2nd edition.

Papineni, K., Roukos, S., Ward, T., and Zhu, W.-J. (2002). BLEU: a Method for Automatic Evaluation of Machine Translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

Parton, K., Tetreault, J., Madnani, N., and Chodorow, M. (2011). E-rating Machine Translation. In *Proceedings of the Sixth Workshop on Statistical Machine Translation*, pages 108–115, Edinburgh, Scotland. Association for Computational Linguistics.

Petrov, S., Barrett, L., Thibaux, R., and Klein, D. (2006). Learning Accurate, Compact, and Interpretable Tree Annotation. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 433–440, Sydney, Australia. Association for Computational Linguistics.

Petrov, S. and Klein, D. (2007). Improved inference for unlexicalized parsing. In *Proceedings of Human Language Technologies: The 2007 Annual Conference of the North American Chapter of the Association for Computational Linguistics. Meeting of the North American Chapter of the Association for Computational Linguistics (NAACL-2007)*. Association for Computational Linguistics.

Raybaud, S., Lavecchia, C., David, L., and Kamel, S. (2009a). Word-and sentence-level confidence measures for machine translation. In *13th Annual Meeting of the European Association for Machine Translation (EAMT-2009)*, Barcelona, Spain. European Association of Machine Translation.

Raybaud, S., Lavecchia, C., Langlois, D., and Kamel, S. (2009b). New Confidence Measures for Statistical Machine Translation. *Proceedings of the International Conference on Agents*, pages 394–401.

Rosti, A.-V., Ayan, N. F., Xiang, B., Matsoukas, S., Schwartz, R., and Dorr, B. J. (2007). Combining Outputs from Multiple Machine Translation Systems. In *Proceedings of the North American Chapter of the Association for Computational Linguistics Human Language Technologies*, pages 228–235, Rochester, New York. Association for Computational Linguistics.

Sánchez-Martínez, F. (2011). Choosing the best machine translation system to translate a sentence by using only source-language information. In Forcada, M. L., Depraetere, H., and Vandeghinste, V., editors, *Proceedings of the 15th Annual Conference of the European Associtation for Machine Translation*, number May, pages 97–104, Leuve, Belgium. European Association for Machine Translation.

Siegel, M. (2011). Autorenunterstützung für die Maschinelle Übersetzung. In Hedeland, H., Schmidt, T., and Wörner, K., editors, *Multilingual Resources and Multilingual Applications: Proceedings of the Conference of the German Society for Computational Linguistics and Language Technology (GSCL)*, Hamburg.

Soricut, R. and Narsale, S. (2012). Combining Quality Prediction and System Selection for Improved Automatic Translation Output. In *Proceedings of the Seventh Workshop on Statistical Machine Translation*, pages 163–170, Montréal, Canada. Association for Computational Linguistics.

Soricut, R., Wang, Z., and Bach, N. (2012). The SDL Language Weaver Systems in the WMT12 Quality Estimation Shared Task. In *Proceedings of the Seventh Workshop on Statistical Machine Translation*, pages 145–151, Montréal, Canada. Association for Computational Linguistics.

Specia, L., Street, S., Court, R., and Felice, M. (2012). Linguistic Features for Quality Estimation. In *Proceedings of the Seventh Workshop on Statistical Machine Translation*, pages 96–103, Montréal, Canada. Association for Computational Linguistics.

Specia, L., Turchi, M., Cancedda, N., Dymetman, M., and Cristianini, N. (2009). Estimating the Sentence-Level Quality of Machine Translation Systems. In *13th Annual Meeting of the European Association for Machine Translation (EAMT-2009)*, pages pp. 28–35, Barcelona, Spain.

Stolcke, A. (2002). SRILM – An Extensible Language Modeling Toolkit. In *Proceedings of the Seventh International Conference on Spoken Language Processing*, pages 901–904. ISCA.

Ueffing, N. and Ney, H. (2005). Word-level confidence estimation for machine translation using phrase-based translation models. *Computational Linguistics*, pages 763–770.

Vilar, D., Avramidis, E., Popović, M., and Hunsicker, S. (2011). DFKI's SC and MT Submissions to IWSLT 2011. In *Proceedings of the International Workshop on Spoken Language Translation 2011*, San Francisco, CA, USA.

Ye, Y., Zhou, M., and Lin, C.-Y. (2007). Sentence Level Machine Translation Evaluation as a Ranking. In *Proceedings of the Second Workshop on Statistical Machine Translation*, pages 240–247, Prague, Czech Republic. Association for Computational Linguistics.

Constructing Reference Semantic Predictions from Biomedical Knowledge Sources

Demeke Ayele[1], *J. P. Chevallet*[2], *Million Meshesha*[1], *Getnet Kassie*[1]
(1) Addis Ababa University, Addis Ababa, Ethiopia
(2) University of Grenoble, Grenoble, France
{demekeayele, meshe8, getnetmk}@gmail.com, jean-pierre.chevallet@imag.fr

ABSTRACT

Semantic tuples are core component of text mining and knowledge extraction systems in biomedicine. The practical success of these systems significantly depends on the correctness and quality of the extracted semantic tuples. The quality and correctness of the semantic predictions can be measured against a benchmark semantic structure. In this article, we presented an approach for constructing a reference semantic tuple structure based on the existing biomedical knowledge sources in which the evaluation is based on the UMLS knowledge sources. In the evaluation, 7400 semantic triples are extracted from UMLS knowledge sources and the semantic predictions are constructed using the proposed approach. In the semantic triples, 87 concepts are found redundantly classified and 207 pair of semantic triples showed hierarchically inconsistent. 128 are found to be non-taxonomically inconsistent. The quality of the semantic triple is also judged using expert evaluators. The Cohen's kappa coefficient is used to measure the degree of agreement between two evaluators and the result is promising (0.9).

Construire des prévisions de référence sémantique à partir de sources de connaissances biomédicales

Les "tuples sémantiques" forment un élément essentiel à la fouille de texte et aux systèmes d'extraction de connaissances dans le domaine biomédical. Le succès en pratique des systèmes exploitant ces informations sémantique, dépend fortement de l'exactitude et de la qualité des tuples sémantiques. La qualité et l'exactitude des informations sémantiques produites automatiquement peuvent être mesurées par rapport à une structure de référence. Dans cet article, nous présentons une approche pour construire une structure sémantique à base de tuples, basée sur des sources existantes dans le domaine biomédicale. L'approche est évaluée en comparaison du méta-thésaurus UMLS. Dans une évaluation préliminaire, 7400 tuples sémantiques ont été aléatoirement extraits de UMLS et les prédictions de relations on été construites en utilisant l'approche proposée. Dans les triplets sémantiques étudiés, 87 concepts se révèlent être classés de manière redondante et 207 paires de triplets sémantiques ont une relation hiérarchique incompatible, et finalement 128 sont jugées taxonomiquement compatibles. La qualité de la relation sémantique est également jugée en utilisant des évaluateurs, experts du domaine. Le coefficient kappa de Cohen est utilisé pour mesurer le degré d'accord entre deux évaluateurs et le résultat est d'ors et déjà prometteur (0,9).

KEYWORDS: Acceptable semantics, domain semantics, knowledge extraction, semantic triples
MOTS-CLÉS: sémantique, la sémantique de domaine acceptables, extraction de connaissances, triples sémantiques

Proceedings of COLING 2012: Technical Papers, pages 133–148,
COLING 2012, Mumbai, December 2012.

1 Introduction

Semantic predictions, semantic triples, are the basic components of text mining and knowledge representation systems. Nowadays, large scale semantic prediction extraction and representation systems are increasingly emerging to sustain text mining and knowledge management systems in biomedicine. These in turn support intelligent and quality healthcare services and management (Abacha and Zweigenbaum, 2011; Cameron, 2011; Denecke, 2008; Harkema et al., 2004).

The practicality and usability of semantic relation extraction systems critically depends on the correctness, accuracy and quality of the extracted semantic predictions. The relations are formed under a general structure of *subject-predicate-object* triples (Harkema et al., 2004; Spasic, 2005), called semantic predictions hereafter. A benchmark is necessary to evaluate the accuracy and quality of the semantic relations generated by the automatic semantic relation extraction systems. This in turn improves the usefulness of the semantic predictions in the knowledge management systems (Abacha and Zweigenbaum, 2011; Denecke, 2008).

Most of the existing semantic triple extraction systems are based on either a shallow (e.g. Wordnet or Ontologies) or a narrower (e.g. terminologies) semantic resources for measuring the accuracy and quality of the extracted semantic relations (Abacha and Zweigenbaum, 2011; Cameron, 2011; Denecke, 2008). These semantic resources either lack the fine-grained semantics (e.g. Wordnet) or focus on narrower domains (e.g. terminological resources), and adopt different semantic representation contexts. This renders difficulties in the semantic resources to use them in benchmarking for independently developed semantic tuple extraction and representation systems (Denecke, 2008).

In biomedicine, various semantic resources have emerged recently (Bada and Hunter, 2007; Herre et al., 2011). They range from terminologies (e.g. UMLS (Keith et al., 1998; Lindberg et al., 1993)) to Ontologies (e.g. BioTop (Beisswanger, 2007)). Most of the ontological resources contain high level semantics of the domain (Beisswanger, 2007), resulting in lack of fine-grained semantic triples that may have significant impact on reasoning and intelligent systems application. Terminologies (e.g. UMLS) are the most common semantic resources utilized as reference in semantic triple extraction and representation because they contain the fine-grained semantic triples in a very specific domain.

For example, the Unified Medical Language System (UMLS) semantics is used to measure the correctness and usefulness of extracted semantic predictions in the work of Abacha and Zweigenbaum (2011), Cameron (2011) and Denecke (2008). According to Keith et al (1998) and Lindberg et al (1993), the UMLS is the integration of many vocabulary sources in biomedicine. It is a widely accepted semantic resource to represent the biomedicine. It has richer semantic content than other terminological resources in biomedical domain yet.

As pointed out previously, most terminological resources, however, are developed using experts for specialized application contexts in the domain (Keith et al., 1998; Lindberg et al., 1993). This makes the semantic tuples to have multiple semantic interpretation contexts and views, which leads to many inconsistencies and ambiguities in the domain representations (Erdogan, 2010; Fan and Friedman, 2008; Freitas et al, 2009).

This problem is intensified if the resources are combined (e.g. UMLS) to integrate the different views and interpretations of the semantic triples. This may significantly affects the accuracy, correctness and quality of the semantic triples (Erdogan, 2010; Morreya, 2009; Mougin and Bodenreider, 2005; Spasic, 2005; Vizenor et al, 2009).

Auditing systems have been developed to asses the semantic inconsistencies and ambiguities in biomedical semantic resources and to suggest corrective measures. For example, in (Erdogan, 2010; Morreya, 2009; Mougin and Bodenreider, 2005; Spasic, 2005; Vizenor et al, 2009), auditing systems are developed to asses the inconsistencies inherent to Unified Medical Language System (UMLS) knowledge sources.

But, while auditing systems have made large contributions in identifying the inconsistencies and ambiguities, the large volume and number of biomedical knowledge sources and many inconsistencies and ambiguities make them difficult to circumvent the inherent problems of the resources (Erdogan, 2010; Friedman et al, 2001). Consequently, using these resources as a benchmark for semantic triple extraction could lead to incorrect interpretation of the semantic triples, which results low accuracy and quality of the semantic predictions.

In this context, a reference semantic tuple structure is required to provide consistent, accurate, and high quality semantic triples for benchmarking semantic prediction extraction systems in biomedicine. The lack of a suitable gold standard reference semantic prediction structure has so far precluded the formal evaluation of semantic triple extraction systems. Most of the existing semantic extraction systems have been informally evaluated using statistical methods through error analysis. A formal evaluation requires measuring the semantic distances of extracted semantics against the benchmark semantics. That is, the spans of texts need to be mapped to concepts and their relationships in the reference semantic structure, which provides a consistent and formal representation of biomedicine.

Constructing such a reference semantic structure needs a comprehensive analysis of the biomedicine semantic knowledge resources (e.g. UMLS) to guarantee the correctness and quality of the semantic triples in them (Erdogan, 2010; Friedman et al, 2001). Furthermore, the analysis is made in perspectives where most inconsistencies and ambiguities are assumed to occur (Erdogan, 2010; Morreya, 2009; Mougin and Bodenreider, 2005; Spasic, 2005; Vizenor et al, 2009).

We have structured our semantic analysis in four perspectives before transforming the semantic knowledge sources into semantic predictions. The first semantic analysis is used to identify redundantly classified concepts to guarantee the correct assignments of concepts in the knowledge source (e.g. UMLS (Fan and Friedman, 2008)). The second semantic analysis is made for ensuring the consistency of hierarchical relationship semantics held by the biomedical knowledge sources (e.g. UMLS semantic network and Metathesaurus (Cimino et al., 2003)).

The third semantic analysis checks the consistency of non-taxonomically related semantic triples in the semantic knowledge sources (e.g. UMLS semantic network and Metathesaurus) (Bodenreider and Burgun, 2004; Vizenor et al., 2009). The fourth analysis verifies the alignment of concepts and semantic types between UMLS knowledge sources. Lastly, the UMLS semantics is transformed into a set of consistent and acceptable semantic predictions.

In this article, we presented a method to construct consistent and domain expert acceptable semantic tuple structure with assessment and analysis of the biomedical knowledge sources applied on Unified Medical Language System (UMLS). The techniques are developed to assess and identify the semantic inconsistencies and ambiguities in the biomedical knowledge sources and transform the knowledge source semantics into a set of semantic triples.

As the approach focuses at the semantic level (concept), it can be applied on languages included in the Unified Medical Language System (UMLS). That is, it can be applied for those languages in the Unified Medical Language System's knowledge sources (e.g. English and some European languages) or the language of its source vocabularies (e.g. SNOMED CT). This makes the proposed approach to have language independent nature.

The approach is based on the language model developed by the National Library of Medicine in designing the Unified Medical Language System to integrate multiple terminologies in the domain of biomedicine. It combines conceptual and lexical representations of the domain semantics. The third Unified Medical Language System (UMLS) resource component, for example, is the SPECIALIST Lexicon, which is designed to have morphological and syntactical language models.

The approach also measures the accuracy, quality and correctness of the transformed semantic tuples using experts. Each semantic tuples are transformed into human readable format and presented to experts. The experts rate the semantics of the tuples by providing judgmental value of 1 or 0, where 1 is acceptable and 0 is unacceptable. The degree of agreement between two evaluators is measured using Cohen's kappa coefficient (k). In this way, three expert evaluators judge the accuracy and quality of the semantic tuples. The result obtained is promising. Finally, the results are discussed and concluded in future works.

2 Background

According to literatures (Freitas and et al, 2009), several semantic resources have been emerging increasingly in biomedical domain. The resources may generally be categorized into lexical, terminologies and Ontologies based on the semantic content they have (Freitas and et al, 2009). For example, Wordnet could be a lexical resource, SNOMED CT or UMLS is a Terminological resource and BioTop is ontological resource.

The Unified Medical Language System (UMLS) is the largest terminological resource in the domain, which has been developed by the National Library of Medicine (NLM) since 1986 as a long term project. Currently, it is an integration of more than 150 biomedical vocabulary sources into its Metathesaurus. The Metathesaurus consists of more than 3 million concepts and their relationships (Keith et al., 1998; Lindberg et al., 1993).

The Unified Medical language System (UMLS) has three semantically correlated components that represent the biomedical domain at various level of semantic granularity. The Unified Medical Language System (UMLS) semantic network represents the high level conceptual domain representations with broader semantic classes, called semantic types. The Unified Medical Language System (UMLS) Metathesaurus also represents the fine-grained domain semantic concepts and the corresponding terms as well as relationships among concepts.

The Unified Medical Language System (UMLS) SPECIALIST Lexicon represents the linguistics knowledge sources and lexical resources. The linguistics knowledge sources include morphological and syntactic attributes of each term in the Metathesaurus. This creates a linkage to span of texts in biomedical documents.

The semantic tuples forming subject-predicate-object triples in the Metathesaurus are logically linked to the semantic network semantic tuples. In Metathesaurus, the concepts are the subjects and objects in the triple whereas the thesauri relationships are the predicates. In the semantic network, the subjects and objects are semantic classes (types) where as the predicate is the semantic network relationships.

The semantic concepts in the Metathesaurus are categorized in at least one semantic type in the semantic network. These concepts are in turn represented by several synonymous terms from multiple vocabulary sources. In this respect, the two knowledge sources of the UMLS, semantic network and Metathesaurus, are semantically linked to structure the semantics of biomedicine.

However, the integration of several vocabulary sources into UMLS has been made using experts with a goal to create a semantic link among the different biomedical resources by preserving the semantics and terms in the original resources. This leads the UMLS to have inherent inconsistency and ambiguity problems in its semantic content (Erdogan, 2010; Fan and Friedman, 2008; Freitas et al, 2009; Friedman et al., 2001; Harkema et al., 2004). According to empirical results in auditing the UMLS (Bodenreider, 2001, 2004; Cimino, 1998; Erdogan, 2010; Fan and Friedman, 2008; Friedman and et al., 2001; Morreya, 2009; Mougin and Bodenreider, 2005; Spasic, 2005; Vizenor et al, 2009), the major sources of these problems are: 1) Due to errors made by experts in the integration process; 2) inconsistencies and ambiguities that arise in the process of preserving the different views and semantic contexts of the original sources in the integration.

Erdogan et al. (2011) quantified the semantic inconsistencies in UMLS concepts from the perspective of their hierarchical relations and showed how inconsistent concepts can help reveal erroneous synonymy relations. The study evaluates consistency by comparing the semantic groups of hierarchically related pair of concepts. As a result, 81, 512 concepts were found to be inconsistent due to differences in semantic groups of a concept and its parents. Morrey et al. (2009), presented Neighborhood Auditing software Tool (NAT), which facilitated the UMLS auditing tasks. It supports neighborhood based auditing, where an auditor concentrates on a focused concept and one of a variety of neighborhoods of its closely related concepts. It also allows an auditor to display knowledge from the two UMLS knowledge sources.

Cimino (1998) developed semantic techniques to audit Metathesaurus for identifying possible inconsistencies. The result of the study showed that out of 57,592 concepts with multiple semantic types, 3.2% were judged ambiguous. Keyword analysis showed 7121 pairs of interchangeable terms. Using the keyword pairs, 5031 pairs of potentially redundant concepts were suggested, of which 65.1% were judged to actually be redundant. Review of the 100,586 parent–child relationships revealed 0.54% that was incorrect. Review of the 219,664 other relationships (RO) (e.g. see in TABLE 1 below) suggested 1299 places in the Semantic Network (SN) where relations between pairs of semantic types could be added.

CHD	Has child relationship	C_1 parent of C_2, inverse_ISA
PAR	Has parent relationship	C_1 child of C_2, ISA
RB	Has a broader relationship	C_1 parent of C_2, inverse_ISA
RN	Has a narrower relationship	C_1 child of C_2, ISA
RL	The relationship is similar or alike	C_1 alike C_2, mapping
RO	Relationships other than CHD, PAR, RB, RN and SY	Associative relationship of C1 & C2
RU	Related, unspecified	Inherited from SN, T1 & T2
SIB	Has sibling relationship	C_1 SIB C_2, sistership

TABLE 1 - META relationships and their mapping

Auditing methods can be classified as logic and non-logic based (Cornet, 2005; Mougin and Bodenreider, 2005). While the logic based methods have been better performing, the semantic structure of UMLS is not consistent with it (Cornet, 2005; Mougin and Bodenreider, 2005). The non-logic based methods (Bodenreider, 2001; Cimino, 1998, 2003; Erdogan, 2010; Fan and Friedman, 2008; Morreya, 2009; Mougin and Bodenreider, 2005; Vizenor et al., 2009) detect and avoid semantic inconsistencies and ambiguities based on semantical and structural properties of the UMLS semantics and fix the problems manually. The methods detect redundant assignments, hierarchical and associative semantics inconsistencies, and hierarchically circular relationships. The purpose of the methods is to enhance the correctness and semantic quality of the UMLS knowledge sources. More comprehensive literature survey about auditing methods can be referred in (Zhu, 2009).

Some semantic predictions systems, in biomedicine, have also used the UMLS semantics for accurate extraction of semantic predictions and measuring the quality of the resulting semantic propositions. For example, in 2008, Denecke the quality and correctness of the extracted semantic predictions are checked against the semantics of the UMLS semantic network in its evaluation. Accuracy is measured in terms of the number of concepts extracted compared to those actually exist in a sentence and the quality of the relation was compared to manually generated semantic structures.

In this context, a semantic structure is correct if it contains all medical concepts in a sentence and if the semantics of the concepts are according to manually constructed representations. Kilicoglu et al (2011) also developed a semantic prediction gold standard from biomedical literatures to evaluate semantic prediction systems (e.g. semRep). However, though the studies were concerned on accuracy, structural and semantical acceptability of the semantic predictions,

manual construction is very limited and consumes more time and effort in large scale semantic prediction systems, which results the need of developing alternative approaches.

3 MATERIALS

The Unified Medical Language System (UMLS) Semantic Network (SN) and Metathesaurus (MT) are used as a baseline semantic resource to evaluate the approach for generating consistent and acceptable semantic predictions under a general structure of object-attribute-value triple. According to the studies in (Keith et al., 1998; Lindberg et al., 1993), UMLS combines many medical vocabularies and provides a mapping structure among them. It is composed of the semantic knowledge components, the metathesaurus and semantic network, and lexical knowledge source, the SPECIALIST Lexicon. The semantic structure in the UMLS is inherently related to the semantic structures of its semantic network and Metathesaurus. Fig. 1 below depicts the semantic relationships of the two UMLS knowledge sources.

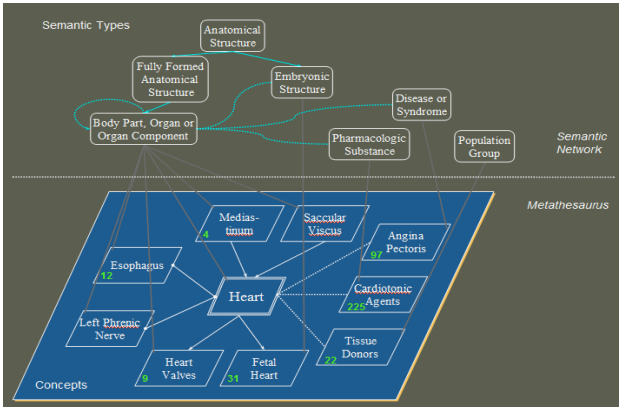


FIGURE 1- Semantic structure of the SN and MT

The upper one is the UMLS semantic network semantic types where as the lower one is the concepts in the UMLS Metathesaurus. The link between the two is the hierarchical relationship. For example, the semantic type **body part, organ and organ component** is a **fully formed anatomical structure**, which is in the semantic network. And **heart** is **body part, organ and organ component,** which is a semantic binding between Metathesaurus and semantic network.

The semantic network consists of 135 semantic types that have been aggregated into a set of 15 semantic groups to reduce complexity (McCray, 2001). For example, the semantic type **Finding** and **Pathologic Function** belong to the semantic group **Disorders**. The semantic types are linked using 54 semantic relationships. For example, the semantic type **Body Part, Organ, or Organ Component** is associated with the semantic type **body substance** by the semantic relationship *location_of*. The semantic type **dysfunction** is related to the semantic type **biologic function** hierarchically, *isa*.

In semantic network, semantic types are related taxonomically in a single inheritance relationship. The hierarchy is rooted at two nodes, the entity and event. Along the hierarchy, the associative relationships defined in the ancestor semantic types are easily inherited by the decedent semantic types unless otherwise the inheritance is blocked explicitly. If a relationship can not be inherited, it is blocked in two ways. The first is inheritance blocking (B), to mean the relationship cannot be inherited by the descendant semantic types. There are also cases where semantic relationships are Defined but Not Inherited (DNI). The relationships are used only in the defining semantic types but not inherited by its decedents.

The semantic types and concepts are related using categorization links. These links are assumed as hierarchical (*isa*) relationships. Intuitively, it is assumed that a semantic relationship defined between two semantic types is also inheritable between pair of concepts categorized in the two semantic types. For example, the relationship *affects* is defined between **Acquired Abnormality** and **organism function** as (*acquired abnormality, affects, organism function*). If it is inheritable, the relationship or its decedents is inherited between concepts categorized in **Acquired Abnormality** (e.g. C0001168) and **Organism Function** (e.g. C0000934) as (C0001168, affects/causes/induces, C0000934).

Fig. 2 below shows the general semantic inheritance structure between the UMLS semantic network and Metathesaurus. In the figure, the semantic types **fully formed anatomical structure** and **biologic function** is related by *location_of*. This semantic relationship can also be inherited by the descendent semantic types of **fully formed anatomical structure** and **biologic function,** which are **body part, organ and organ components**, and **diseases and symptom.** The same semantic relationship can also be inherited by the corresponding semantic concepts in the Metathesaurus between as shown in Fig. 2 below **adrenal cortex** and **adrenal cortical hypofunction**.

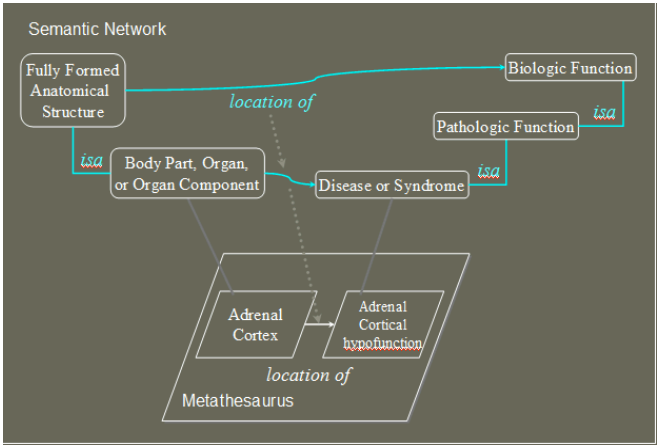


FIGURE 2 – Semantic inheritance between SN and MT

Though difficult and challenging [23], associative relationships (e.g. affects) can be inherited by pair of concepts in MT. They are not explicitly defined among concepts, which results the requirement of mapping the SN relationships. Furthermore, some MT relationships can't map to the existing SN relationships, which also results the need of defining additional SN relationships. In this study, the MT relationships considered are listed in table 1, and only the existing SN relationship mapping is made. Concepts in Metathesaurus are groups of similar terms from the various source vocabularies. These terms create linkage to the SPECIALIST Lexicon, which in turn enables to create linkage to domain texts. Similarly, relationships between concepts can be mapped among terms and in turn between span of texts in the discourse.

In this article, therefore, the UMLS semantic network, Metathesaurus and their semantic binding are used as a semantic resource except co-occurrence relationship. Within these, the SN (e.g. SRSTRE2.TXT) and MT (e.g. MRSTY.RRF, MRREL.RRF) and UMLS relationship files in addition to the semantic groups are used in constructing the semantic predictions.

4 METHOD

In this article, we proposed an approach for constructing consistent and acceptable semantic triples under a framework of *object-attribute-value*/*subject-predicate-object* triple from biomedical knowledge sources. In each semantic triple, the object/subject and value/object are either semantic types or semantic concepts or atoms. The attribute/predicate is the semantic relationship defined/inherited between semantic types or semantic concepts. For example, in the triple (***pharmacologic substance****, treats,* ***pathologic function***), ***pharmacologic substance*** is the *object/subject* and ***pathologic function*** is *value/object* while ***treats*** is *attribute/predicate*.

In the approach, two general steps are made to complete the construction. First, all possible semantic triples in the knowledge sources (e.g. UMLS semantic network and Metathesaurus) are extracted. Second, the consistency and acceptability of the triples are assessed. The notations C=concept, T=semantic type, G=semantic group, R=relationship, D=inheritable, B=Blocked, DNI=Defined but Not Inheritable are used henceforth. Fig. 3 shown below depicts the general semantic prediction process.

4.1 Semantic Triple Extraction

In Metathesaurus, semantic relationships are based at each semantic context of the terms, which we referred as semantic atoms, hereafter. Semantic triples can be constructed at the level of semantic atoms, concepts, types and groups. That is, semantic types in each semantic group, semantic concepts in each semantic type, and semantic atoms in each semantic concept are extracted to have explicit representation of the structure. This enables to identify concepts that a semantic atom belongs, semantic types that a concept belongs, and a semantic group that a semantic type belongs. The extraction is splitted into two steps. The first is the extraction of taxonomically related semantic triples and next, the extraction of non-taxonomically related semantic triples.

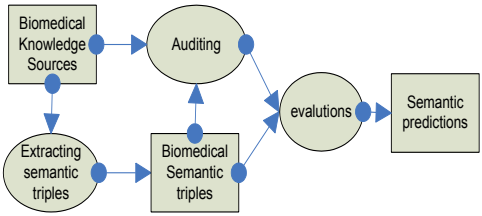


FIGURE 3– Semantic Prediction

Taxonomic (hierarchical) semantic triple extraction is straightforward. Because the taxonomy is transitive, relations that can be derived are easily inferred from the taxonomy. For instance, given a taxonomic hierarchy (C_3, C_2, C_1), the triples (C_2, C_1), (C_3, C_2), and (C_3, C_1) can be derived. The triple (C_3, C_1) is inferred from the transitive characteristics of taxonomic relationships. Fully inherited SN files (SRSTRE2.txt) and the hierarchical relation MT files (MRHIER.RRF and MRSTY.RRF) are used to construct the taxonomic structure. Fig. 4 below shows the result of the taxonomic semantic triple construction.

Algorithm: *building taxonomic semantic triples*

For each sem. group, G, obtain semantic types

For each sem. type, T, obtain semantic concepts

For each sem. concept, C, obtain sem. atoms, A

Build the taxonomic structure, ISA (A, C, T, G)

```
Semanti triples between T and G:
     (T195|Antibiotic, CHEM|Chemicals & Drugs)
     (T118|Carbohydrate, CHEM|Chemicals & Drugs)
     (T103|Chemical, CHEM|Chemicals & Drugs)
     (T200|Clinical Drug, CHEM|Chemicals & Drugs)
     (T111|Eicosanoid, CHEM|Chemicals & Drugs)
     (T126|Enzyme, CHEM|Chemicals & Drugs)
     (T125|Hormone, CHEM|Chemicals & Drugs)
     (T119|Lipid, CHEM|Chemicals & Drugs)
     (T192|Receptor, CHEM|Chemicals & Drugs)
     (T110|Steroid, CHEM|Chemicals & Drugs)
     (T127|Vitamin, CHEM|Chemicals & Drugs)

Semantic triples between C and T:
     (C0014184|Endonuclease, T116|Amino Acid, Peptide, or Protein)
     (C0014185|Endonuclease, T116|Amino Acid, Peptide, or Protein)
     (C0014186|Endonuclease, T116|Amino Acid, Peptide, or Protein)
     (C0014197|Deoxyribonuclease, T116|Amino Acid, Peptide, or Protein)
     (C0014207|Deoxyribonuclease, T116|Amino Acid, Peptide, or Protein)
     (C0014208|GdoI Endonuclease, T116|Amino Acid, Peptide, or Protein)
     (C0014209|GinI Endonuclease, T116|Amino Acid, Peptide, or Protein)
     (C0014210|GoxI Endonuclease, T116|Amino Acid, Peptide, or Protein)
     (C0014221|MleI Endonuclease, T116|Amino Acid, Peptide, or Protein)
```

FIGURE 4- Snapshot of taxonomic semantic triples

In non-taxonomic semantic triple extraction, all semantic classes (semantic types, concepts and atoms) are considered as concepts. Non-taxonomic semantic triples of a concept C_i in which C_i is the subject of the triple (C_i, R, C_k) are extracted. Then, triples that have the same relationships (in R) and object concepts (in C_k) are merged. Finally, only semantic triples differing with at least one of C_i, *R, C_k,* are considered as useful.

Relationship inheritance between semantic triple in SN (T_1, *R,* T_2) and the corresponding semantic triples in MT (C_1, *r,* C_2) in which C_1 and C_2 are related hierarchically to T_1 and T_2 respectively is the mapping of *R to r*, where *r* is either same as *R* or decedents of *R*. This mapping is valid if the inheritance of *R* is permitted (i.e. D) otherwise the mapping is blocked (B or DNI). The fully inherited SN files (SRSTRE2.txt) and a MT file (MRSTY.RRF and MRREL.RRF) are used to develop the nontaxonomic structure. The algorithm below constructs the nontaxonomic propositions.

Algorithm: *non-taxonomic semantic triples*

For each sem. type (T_i), obtain (T_i, R, T_k)

For each C_i in T_i, map R to r, obtain (C_i, r, C_k)

Collect tuples (T_i, R_{ij}, T_j) and (C_i, r_{ij}, C_j)

Repeat from i=1 to 135, all semantic types

```
Non-taxonomic semantic tuples (T, R, T):
    (Acquired Abnormality, affects, Organism)
    (Acquired Abnormality, affects, Virus)
    (Acquired Abnormality, location_of, Fungus)
    (Acquired Abnormality, location_of, Virus)
    (Acquired Abnormality, occurs_in, Age Group)
    (Acquired Abnormality, occurs_in, Group)
    (Acquired Abnormality, part_of, Amphibian)
    (Acquired Abnormality, part_of, Animal)
    (Age Group, exhibits, Behavior)
    (Age Group, exhibits, Individual Behavior)
    (Age Group, exhibits, Social Behavior)
    (Age Group, interacts_with, Age Group)
    (Age Group, performs, Activity)

Non-taxonomic semantic tuples (C, r, C):
    (C0991536, has_dose_form, C0716419)
    (C0991536, has_dose_form, C0716420)
    (C0991536, has_dose_form, C0716451)
    (C0991536, has_dose_form, C0716453)
    (C0944728, analyzed_by, C0027342)
    (C0944728, measured_by, C0043481)
    (C0944728, system_of, C0027342)
    (C0944728, component_of, C0043481)
    (C0944728, property_of, C1264657)
    (C0944729, system_of, C0005767)
    (C0944729, component_of, C0072980)
    (C0944729, property_of, C0560150)
    (C0944729, class_of, C1315017)
    (C0944730, analyzed_by, C0370231)
```

FIGURE 5 - Snapshot of Non-Taxonomic Semantic Triples

4.2 Consistent Semantic Triples

Consistency is defined as accurate representation of the semantic tuples or non-redundant classification of concepts in MT. Inconsistencies are resulted from inaccurate representation of semantic network and Metathesaurus relations, and inaccurate concept categorizations. Detecting and removing the redundant classifications and the inaccurate representation of semantic tuples could eliminate the semantic inconsistencies.

Redundant classification occurs in cases if T_1 is decedents of T_2 and a concept C_1 is classified under T_1 and T_2. In this situation, the assignment of C_1 to T_2 is redundant. This is because it can be inferred from the assignment of C_1 to T_1 transitively. The redundant assignment (or the semantic tuple C_1 *isa* T_2) is removed or made implicit to make consistent. The algorithm below is developed to detect and remove the redundancy.

Algorithm: *removing redundant classifications*

For each concept C_i *in MT, obtain its sem. types*

Obtain taxonomically related semantic types

Remove the ancestor STs, if any

Hierarchical relationship inconsistencies occur in cases where T_1 becomes an ancestor of T_2 in relationship conditions if C_1 and C_2 are related taxonomically in MT (C_1 *isa* C_2), and C_1 is in T_1 (C_1 *isa* T_1), C_2 is in T_2 (C_2 *isa* T_2). That is, T_1 must be decedent or the same as T_2 to make consistent. The next algorithm is developed to detect and remove such inconsistencies.

Algorithm: *hierarchical inconsistencies*

For each related concepts, C_i *and* C_j

Obtain the semantic types for each, C_i *and* C_j

Remove the intersection STs of C_i *and* C_j

Verify the STs of C_i *are decedents of that of* C_j

Remove the inconsistencies, if any

Unlike the semantic network relationships, Associative relationships in MT are not explicitly defined (Vizenor et al, 2009). This creates difficulties in mapping the SN semantics to the corresponding MT semantics, resulting associative inconsistencies. This occurs when the semantic relationships between two semantic types, T_1 and T_2, have no direct mapping to the semantic relationships made by two semantic concepts, C_1 and C_2, which are categorized in T_1 and T_2 respectively.

For example, the semantic type ***body part, organ and organ component*** is hierarchically related to ***fully-formed anatomical structure***. The semantic type ***disease and syndrome*** is also related to ***pathologic function*** hierarchically. A semantic relationship *location_of* exists between semantic type ***body part, organ and organ component,*** and ***disease and syndrome.*** *Adrenal cortex* and *adrenal cortical hypofunction* are two Metathesaurus concepts categorized in ***body part, organ and organ component,*** and ***disease and syndrome*** respectively**.** However, the relationship

between the two concepts are not explicitly defined or inherited. In order to make consistent semantic mapping, the relationship between the two concepts should be either *location_of* or its decedents, if any.

We assumed that the inheritable relationship (R) between semantic types T_1 and T_2 or its decedents in SN are also inheritable to all concepts categorized in T_1 and T_2. This leads to develop simple algorithm to map the semantic tuples in SN to semantic tuples in MT. In this article, the semantic mapping considers only semantic relationships in MT indicated in table 1. Specifically, for example, other relationships (RO) and unspecified relationships (RU) are considered for associative semantics mapping. After mapping the semantic relations between the two knowledge sources, manual assessment is made to assure the consistency of the mapping.

5 Results and Discussion

The approach is evaluated by extracting a total of 7400 semantic triples from the Unified Medical Language System (UMLS 2010AB). I.e. there is no special consideration for the semantics of either hierarchical or associative relationship triples. Out of 7400, 4040 are found to be hierarchically related semantic triples, which account 55% of the total. 3360 semantic triples, which accounts about 45% of the total, are found to be non-hierarchically (associatively) related.

This seems that hierarchically related semantic triples are provided more emphasis than associative relations. However, according to the empirical analysis, most of the semantic relationships in MT are hierarchical as they brought from thesauri relationships of the source vocabularies.

In an empirical analysis of the different causes of inconsistencies such as redundant classification, hierarchical and associative relationships, we have compared each of them from the total semantic triples and to the count of semantic triples in the two semantic classes, taxonomic and non-taxonomic. This enables to forecast the trend of the possible inconsistencies in about 15 million semantic triples in the UMLS.

Out of 4040 hierarchically related semantic triples, we have obtained 87 redundantly categorized concepts, which they are removed accurately. Similarly, in the taxonomically related semantic triples, only 207 semantic triples are found to have hierarchically inconsistent in the assignments of concepts to semantic types. This account 5% to the taxonomically related semantic triples and 0.03% to the total semantic triples extracted.

In the case of non-taxonomically related semantic triples, we obtained 128 semantic inconsistencies in mapping the semantic network triples to the corresponding Metathesaurus semantic triples. This accounts 0.04% of non-taxonomically related semantic triples. Some of these inconsistencies come from lexical variations of the relationship phrases and the blocking of inheritances.

Finally, one hundred randomly selected semantic triples are presented to expert evaluators. Each semantic triple is judged by the two evaluators and classified in either 1 (acceptable) or 0 (unacceptable). In evaluator A, 87 are accepted and 13 are unaccepted. In evaluator B, 93 are accepted and 7 are unaccepted. Five semantic triples are unaccepted by evaluator A but accepted by B. Three semantic triples are unaccepted by evaluator B but accepted by the A. Twelve semantic triples are unaccepted and eighty semantic triples are accepted in common.

Cohen's kappa coefficient (k) is computed to see the degree of agreements between two evaluators where k= (pr (a)-pr (e))/ (1-pr (e)). Pr (a) is the relative observed agreement and pr (e) is the probability of random agreement. The result is 0.9, which indicates better agreement between the two evaluators.

Conclusion and Future Work

In order to utilize the biomedical knowledge sources as a benchmark for quality semantic prediction extraction, the quality and correctness of the semantic triples should be assured by domain experts and the inherent inconsistency and ambiguity problems need to be alleviated.

In this article, we have developed an approach for assessing inconsistency problems and transforming the knowledge source semantics to consistent and domain expert acceptable semantic triples. In the approach, we have developed techniques to extract semantic triples in the Unified Medical Language System (UMLS) and transform the triple in the form of *subject-predicate-object* triplets. Furthermore, to assess the inconsistencies related to redundant classification, hierarchical and associative relationships, algorithms are developed.

A preliminary evaluation is conducted by extracting 7400 semantic triples from Unified Medical Language System (UMLS) knowledge sources. Though the number of semantic triples considered is small, the result of the evaluation is promising. However, for accurate result and for our purpose, we will increase the number of semantic triples to one hundred thousand. Furthermore, the quality (acceptability and naturalness) of the semantic triples are also judged using domain experts. The Cohen's kappa coefficient (k) is used to measure the degree of agreement between the evaluators and the result is promising (0.9).

The approach developed in this article is limited to the use of the study in knowledge extraction in biomedicine. But, to utilize the full semantic potential of the biomedical knowledge sources, a generic and rigorous approach, which transforms its semantics to standard semantic structure and eliminate the possible inconsistencies and ambiguities are required.

Reference

Abacha, A and Zweigenbaum, Z (2011). Automatic Extraction of Semantic Relations between Medical Entities: A Rule Based Approach. Journal of Biomedical Semantics: Fourth International Symposium on Semantic Mining in Biomedicine, 2(2011), 1-11.

Bada, M and Hunter, L (2007). Enrichment of OBO Ontologies. Journal of Biomedical Informatics, 40 (2007), 300–315.

Beisswanger, E (2007). BioTop: An Upper Domain Ontology for the Life Sciences. IOS Press, pp. 1-7.

Bodenreider, O (2001). Circular Hierarchical Relationships in the UMLS: Etiology, Diagnosis, Treatment, Complications and Prevention. AMIA, 57-61.

Bodenreider, O and Burgun, A (2004). Aligning Knowledge Sources in the UMLS: Methods, Quantitative Results, and Applications. IMIA: Medinfo, 327-331.

Cameron, D (2011). Semantic Predications for Complex Information Needs in Biomedical Literature. Proceedings of the 5th IEEE International Conference on Bioinformatics and Biomedicine, 512-519.

Cimino, J (1998). Auditing the Unified Medical Language System with Semantic Methods. Journal of the American Medical Informatics Association, 5 (1998), 41-51.

Cimino, J. and et al (2003). Consistency across the hierarchies of the UMLS semantic network and Metathesaurus. Journal of biomedical informatics, 36 (2003), 450–461.

Cornet, R (2005). Two DL-based Methods for Auditing Medical Terminological Systems. AMIA 2005 Symposium Proceedings, 166-170.

Denecke, K (2008). Semantic Structuring of and Information Extraction from Medical Documents Using the UMLS. Methods Inf. Med., 4 (2008), 425-434.

Erdogan, H (2010). Exploiting UMLS Semantics for Checking Semantic Consistency among UMLS concepts. MEDINFO, 749-753.

Fan, J. and Friedman, C (2008). Semantic reclassification of the UMLS concepts. Bioinformatics, 24 (2008), 1971-1973.

Freitas, F. and et al (2009). Survey of current terminologies and ontologies in biology and medicine. RECIIS – Elect. J. Commun. Inf. Innov Health, 7-18.

Friedman, C. and et al (2001). Evaluating the UMLS as a source of lexical knowledge for medical language processing. Proceedings of the AMIA Symposium, 189-193.

Harkema, H. and et al (2004). A Large Scale Terminology Resource for Biomedical Text Processing. HLT-NAACL 2004 Workshop: Biolink 2004, Linking Biological Literature, Ontologies and Databases, 53-60.

Herre, H. and et al (2004). OBML - Ontologies in Biomedicine and Life Sciences. Journal of Biomedical Semantics: Ontologies in Biomedicine and Life Sciences. 2(2011).

Keith, E and et al (1998). The Unified Medical Language System: Toward a Collaborative Approach For Solving Terminological Problems, JAMIA, 5(1998), 12-16.

Lindberg, D. and et al (1993). The Unified Medical Language System. Methods of Information in Medicine, 281-291.

McCray, A. T (2001). Aggregating UMLS Semantic Types for Reducing Conceptual Complexity. MEDINFO, 216-220.

Morreya, C. P (2009). The Neighborhood Auditing Tool: A Hybrid Interface for Auditing the UMLS. J Biomed. Inform, 42 (2009), 468-489.

Mougin, F. and Bodenreider, O (2005). Approaches to Eliminating Cycles in the UMLS Metathesaurus: Naïve Vs. Formal. AMIA Symposium Proceedings, 550-554.

Spasic, I (2005). Text Mining and Ontologies in Biomedicine: Making Sense of Raw Text, Briefings in Bioinformatics, Henry Stewart Publications., 6(2005), 239-251.

Vizenor, L and et al (2009). Auditing Associative Relations Across Two Knowledge Sources. Journal of Biomedical Informatics, 42 (2009), 426-439.

Zhu, X (2009). A Review of Auditing Methods Applied to The Content of Controlled Biomedical Terminologies. Journal of Biomedical Informatics, 42 (2009), 413-425.

Kilicoglu, H and et al (2011). Constructing a Semantic Predication Gold Standard from the Biomedical Literature. BMC Bioinformatics, 12(2011), 1-17.

Translation Quality-Based Supplementary Data Selection by Incremental Update of Translation Models

Pratyush Banerjee[1], Sudip Kumar Naskar[1], Johann Roturier[2], Andy Way[3], Josef van Genabith[1]
(1) CNGL, School of Computing, DCU
{pbanerjee,snaskar,josef}@computing.dcu.ie
(2) Symantec Ireland, Dublin
johann_roturier@symantec.com
(3) Capita Translation and Interpreting, Delph, UK
andy.way@capita-ti.com

ABSTRACT
Supplementary data selection from out-of-domain or related-domain data is a well established technique in domain adaptation of statistical machine translation. The selection criteria for such data are mostly based on measures of similarity with available in-domain data, but not directly in terms of translation quality. In this paper, we present a technique for selecting supplementary data to improve translation performance, directly in terms of translation quality, measured by automatic evaluation metric scores. Batches of data selected from out-of-domain corpora are incrementally added to an existing baseline system and evaluated in terms of translation quality on a development set. A batch is selected only if its inclusion improves translation quality. To assist the process, we present a novel translation model merging technique that allows rapid retraining of the translation models with incremental data. When incorporated into the 'in-domain' translation models, the final cumulatively selected datasets are found to provide statistically significant improvements for a number of different supplementary datasets. Furthermore, the translation model merging technique is found to perform on a par with state-of-the-art methods of phrase-table combination.

KEYWORDS: Statistical Machine Translation, Domain Adaptation, Supplementary Data Selection, Model Merging, Incremental Update.

Proceedings of COLING 2012: Technical Papers, pages 149–166,
COLING 2012, Mumbai, December 2012.

1 Introduction

Statistical Machine Translation (SMT) has grown to be the most dominant machine translation paradigm. A prerequisite for SMT is the availability of sizeable parallel training data. The popularity of SMT has led to the free availability of a variety of parallel corpora on the web. While some such corpora comprise data from wide-coverage domains such as politics or news, others are based on much more focused and narrower domains such as medical texts or software manuals. In order to create an SMT system for a specific domain which does not have sufficient parallel training data, domain adaptation methods are necessary to best utilise supplementary parallel training data from available out-of-domain or related-domain corpora. However, the conventional wisdom of more data being better does not always hold true for domain-specific systems (Axelrod et al., 2011). Adding a lot of out-of-domain data to an in-domain SMT system tends to overwhelm the in-domain characteristics, thereby negatively affecting translation quality. Thus relevant data selection from large supplementary out-of-domain data plays an important part in domain adaptation of SMT systems.

In this paper we focus our efforts on creating an SMT system for translating user-generated forum content in Symantec web forums. Being a multinational company, Symantec supports web forums for its products and services in multiple languages with the English forum being both the oldest and (by far) the largest with considerable amounts of relevant information. Translating the forum content enables this information to be available across all languages. Moreover, these forums have also become effective sources of self-service, thus providing an alternative to traditional customer service options (Roturier and Bensadoun, 2011). However, a major challenge in building a system for forum content translation is the lack of parallel forum data for training. To overcome this challenge, we utilise 'in-domain' (but 'out-of-style') parallel training data in the form of Symantec translation memories (TMs). Symantec TMs comprise internal documentation on its products and services along with user manuals, product descriptions and some customer service communications. The forum data on the other hand, consists of posts where customers and Symantec employees discuss and solve specific problems pertaining to specific products and services. Although the TM and the forum data are in the same domain, the TM data is professionally edited and generally conforms to Symantec's controlled language guidelines. By contrast, the forum data is often noisy, not controlled by any quality guidelines and in general having a wider vocabulary and colloquialisms. This difference between the training data and target domain necessitates the use of supplementary datasets to improve translation quality.

Given the TM-based domain-specific baseline model and an additional general-domain supplementary dataset, we iteratively select batches of sentences from the supplementary dataset and add this to the in-domain translation model of the baseline system and evaluate the translation quality in terms of automatic evaluation metrics on a development set (devset). A batch is approved for addition to the baseline model only upon improvement over the baseline evaluation metric scores. In order to incrementally and rapidly retrain and evaluate the evolving translation models with each additional batch of sentences, a translation model is estimated for each batch under consideration in isolation and subsequently merged with the larger translation model using a novel phrase-table merging mechanism.

Prior to the iterative batch selection process, the supplementary training data is ranked using perplexity (normalised with sentence length) with respect to a source-side forum data language model. This technique allows the selection of batches of sentence pairs from the supplemen-

tary data with perplexity scores within a close range. Our experiments are conducted for the English–French (En–Fr) and English–German (En–De) language pairs. We use three different freely available parallel corpora as supplementary sources of data. Our experiments show that the selected supplementary datasets when incorporated into the baseline translation model consistently improve translation quality over the baseline translations, for different supplementary data sources. Comparing our data selection method with existing data selection approaches confirms the superiority of our approach in terms of translation quality improvements. In addition to the data selection, we develop a phrase table merging technique as an efficient alternative to established methods of model combination. We compare our technique of model combination to the traditional approach of static retraining, use of multiple translation models (Koehn and Schroeder, 2007) as well as mixture modelling with linear interpolation (Foster and Kuhn, 2007) to find that our technique performs at par with most of these techniques in terms of translation quality.

While the translation quality based data selection technique performs well in the experiments presented in this paper, there is a risk that the approach may overfit on the small devsets used (small devsets are a typical situation in real-life domain adaptation scenarios). In particular, this can happen if the set is not 'fully' representative of the target domain in question. Hence the evaluation during the iterative data selection phase should ideally be carried out for multiple devsets and the intersection of selected datasets from each run should be used. However generating multiple devsets for a given target domain (here user forums) is prohibitively expensive involving considerable manual effort. To alleviate this issue, the source data of the devset selected for the set of experiments reported here, is randomly chosen from a large collection of the target domain data and is ensured to be truly representative of the the target domain in terms of meta-statistics.[1] Furthermore, due care is taken during the manual translation process to preserve the characteristics of the target domain.

The rest of the paper is organized as follows: Section 2 reviews related work relevant to the task. Section 3 introduces our approach of data selection and phrase-table merging. Section 4 presents the experimental setup for our and comparative approaches. Section 5 presents the results and analysis followed by conclusions and future work.

2 Related Work

The idea of supplementary data selection from related or unrelated domains to boost the performance of sparse 'in-domain' models has been widely practised in domain adaptation of SMT (Eck et al., 2004). A wide variety of criteria for data selection has been explored ranging from information retrieval techniques (Hildebrand et al., 2005) to perplexity or cross-entropy on 'in-domain' datasets (Foster and Kuhn, 2007; Banerjee et al., 2011). Out-of-vocabulary (OOV) words with respect to training data (Daume III and Jagarlamudi, 2011; Banerjee et al., 2012) are used to mine supplementary data sources for adaptation. (Axelrod et al., 2011) presents a technique of using the difference in cross-entropy of the supplementary sentence pairs on 'in-domain' and 'out-of-domain' datasets for ranking and selection by thresholding. All these techniques rely on selecting supplementary data based on its similarity with the target domain using different measures of similarity like perplexity or OOV word rate. However, perplexity reduction often does not correlate with translation quality improvement (Axelrod, 2006). In this paper we address this issue head-on by directly using translation quality as a

[1]The parameters used are average sentence length, average type-token ratio, average stop word and function word ratio and the standard deviations of the same measures.

guide for data selection. To the best of our knowledge this is a novel approach and one of the main contributions of the paper.

In order to facilitate incremental retraining, we develop a phrase-table merging mechanism that is used to incrementally merge small phrase-tables estimated on incremental batches of supplementary dataset. Incremental updates of translation models have been attempted using a stepwise online expectation-maximization algorithm (Cappé and Moulines, 2009) for stream-based translation models (Levenberg et al., 2010) or using suffix arrays (Callison-Burch et al., 2005) to store the source–target alignments in memory. Our approach differs from these methods primarily in how we update translation model probabilities. The domain-specific aspect of our experimental setup allows us to avoid costly incremental alignment estimations. Furthermore, our approach enables merging independent translation models estimated on different domain-specific word/phrase alignments providing an alternative to other model combination techniques. While simple concatenation of in-domain and out-domain data prior to (re-) training is a commonly used (but costly) technique, multiple phrase-tables (one on each domain) can directly be combined using the decoder (Koehn and Schroeder, 2007), or interpolated using linear or log-linear weighted combination using mixture modelling (Foster and Kuhn, 2007). Our phrase-table merging technique is motivated by the linear interpolation based approach, but differs in our use of phrase-counts to merge multiple phrase-pairs.

3 Incremental Data Selection and Model Merging

This section describes in detail our data selection method and the phrase-table merging technique developed for incremental model updates.

3.1 Batching Sentence Pairs in Supplementary Data

The primary objective of our experiments is to identify the sentence pairs in the 'out-of-domain' supplementary datasets which when incorporated into the 'in-domain' model would improve translation performance. Ideally, for every sentence pair in the supplementary datasets, a new translation model needs to be retrained and its performance evaluated in terms of evaluation metrics. A sentence is suitable for selection only when its inclusion improves the translation quality of the baseline system. However, to manage the scaling issue of this approach, instead of evaluating individual sentence pairs, we group a number of them together in every iteration. In addition, updating any sizeable SMT model with a single sentence pair is unlikely to produce any measurable changes in overall translation output. The supplementary datasets are initially ranked according to their normalised perplexity with respect to a language model estimated on the English user forum dataset. In every iteration, for each batch we pick up a set of sentence pairs whose perplexity lies within a small predefined range (to be supplied by the user as input). For our experiments we use an ad-hoc value of 1 for the range although a further detailed investigation on the effect of the range size on data selection is planned for future. Since perplexity is used as a measure of 'closeness' with respect to the target domain, all pairs in the selected batch have perplexity within a small range (with a value of 1) ensuring uniform closeness of all sentences within the batch with respect to the target domain.

3.2 Selection Algorithm

To decide whether a particular batch of supplementary sentence pairs is suitable for improving translation quality, we use the process outlined in Algorithm 1. The algorithm starts with a baseline translation model BL, a baseline translation score sc_0, a perplexity range r and

Algorithm 1 Supplementary data batch selection for translation performance improvement

Require: $BL \leftarrow$ Baseline Model, $sc_0 \leftarrow$ Baseline Score, $sup \leftarrow \{pp_i, src_i, trg_i\}$, $r \leftarrow$ Perplexity Range;

1: $itn \leftarrow 1$; $step \leftarrow r$;
2: $b_{itn} \leftarrow \{\}$; $i \leftarrow 1$;
3: **while** $not(EOF(sup))$ **do**
4: **if** $pp_i \leq step$ **then**
5: $b_{itn} \leftarrow b_{itn} \cup \{src_i, trg_i\}$; $i = i + 1$;
6: **else**
7: $model_{itn} \leftarrow train\_model\{b_{itn}\} \cup BL$;
8: $sc_{itn} = evaluate\_on\_dev\{model_{itn}\}$;
9: **if** $sc_{itn} \geq sc_0$ **then**
10: $BL \leftarrow model_{itn}$; $sc_0 \leftarrow sc_{itn}$;
11: **end if**
12: $itn = itn + 1$;
13: $step \leftarrow step + r$; $b_{itn} \leftarrow \emptyset$
14: **end if**
15: **end while**

a supplementary dataset comprising source and target sentence pairs along with perplexity scores. Source and target sentence pairs are batched into a group (lines 4-6) as long as their perplexity values fall below the specified range. Once the batch is selected, a new translation model is trained on the batch and the batch model is merged with the baseline model to generate an updated model $model_{itn}$ (line 7). The updated model is then used to evaluate the devset using automatic evaluation metrics (line 8) and generate a new translation score sc_{itn}. The algorithm tests if the new score is better than the previous baseline score (line 9) and if found better updates the baseline model and score with the current model and score value in the iteration. Eventually the perplexity range is extended to the next step, and the batch is cleared for accommodating the next batch of sentences (line 13). This process runs as long as there are no more batches to process. Selected batches are accumulated to produce the final supplementary dataset used for adaptation. Since the batches are ordered according to perplexity-based similarity with respect to the target domain the algorithm makes it increasingly harder for a batch to get into the final selection as (i) later batches are less similar to the targeted domain and (ii) they need to improve on a steadily improving baseline. Therefore the algorithm implements the intuition that only those parts of generic supplementary data are selected which are good enough to generate better translation quality on the devset.

A generic SMT system is usually comprised of three different statistical components: translation model (TrM), language model (LM) and a lexical reordering model (RoM). Algorithm 1 is general enough to handle updates in all these component models. However, in this paper we only report experiments with TrM and RoM model updates and use statically trained LMs (cf. Section 3.5)

3.3 Phrase-table Merging

Ideally for every iteration step, the selected batch of supplementary sentence pairs should be combined with the 'in-domain' training data of the baseline model and a new model should be estimated. Considering the computational cost involved in full retraining, clearly this is not feasible in an iterative framework. In order to facilitate an incremental approach we develop a

set of techniques to avoid full retraining by estimating a model only on the small incremental batch and then merging the models with the existing baseline models.

Word alignment estimation is the most computationally expensive process in TrM training. Thus in order to avoid re-estimation of word-alignments in every iteration, we once and for all pre-compute the word alignments on the entire supplementary dataset and use this in every iteration. This not only reduces the estimation overhead but also addresses the issue of having poor word alignments due to small amounts of parallel data in every iteration. Word-alignments are known to benefit from domain-specific over-fitting (Gao et al., 2011) which motivated us to keep our 'in-domain' (computed on Symantec TM data) and 'out-domain' (computed on supplementary dataset) word alignments separate from each other. Hence the phrase-pairs extracted for each domain (Symantec TMs or Supplementary Datasets) are only based on domain-specific word alignments estimated from the specific corpora.

To achieve lexical table merging, the standard tables are augmented with the source and target word counts (in addition to lexical probabilities). Once new lexical tables are created on the selected batch, the baseline lexical tables are scanned for shared entries and the corresponding probabilities are updated using the formulae in (1):

$$\begin{aligned} lex_{merged}(e|f) &= lex_{bl}(e|f) \times \frac{wc_{bl}(f)}{wc_{bl}(f)+wc_{inc}(f)} + lex_{inc}(e|f) \times \frac{wc_{inc}(f)}{wc_{bl}(f)+wc_{inc}(f)} \\ lex_{merged}(f|e) &= lex_{bl}(f|e) \times \frac{wc_{bl}(e)}{wc_{bl}(e)+wc_{inc}(e)} + lex_{inc}(f|e) \times \frac{wc_{inc}(e)}{wc_{bl}(e)+wc_{inc}(e)} \end{aligned} \tag{1}$$

where $lex_{bl}, wc_{bl}, lex_{inc}$ and wc_{inc} indicate the baseline lexical probability, baseline word count, incremental lexical probability and incremental word count, respectively. e and f indicate the source and target words in this context. Entries which are not shared between the base model and the batch lexical tables are simply added to the new merged lexical table. Equation 1 approximates the lexical probabilities which would result from full retraining.

Once the lexical tables have been updated, the phrase-table estimation is completed on the batch data using the merged lexical tables. Being estimated on the merged lexical table, the inverse and direct lexical weights are already up-to-date in the new phrase-table. Hence only the remaining probabilities and counts require updates. In a similar approach to the lexical table merging, every entry in the new (incremental) batch phrase-table, is compared against the older (baseline) phrase-table and the shared phrase pairs are updated by the formulae in (2):

$$\begin{aligned} \phi_{merged}(e|f) &= \phi_{bl}(e|f) \times \frac{c_{bl}(f)}{c_{bl}(f)+c_{inc}(f)} + \phi_{inc}(e|f) \times \frac{c_{inc}(f)}{c_{bl}(f)+c_{inc}(f)} \\ \phi_{merged}(f|e) &= \phi_{bl}(f|e) \times \frac{c_{bl}(e)}{c_{bl}(e)+c_{inc}(e)} + \phi_{inc}(f|e) \times \frac{c_{inc}(e)}{c_{bl}(e)+c_{inc}(e)} \end{aligned} \tag{2}$$

where $\phi_{bl}, c_{bl}, \phi_{inc}$ and c_{inc} indicate the baseline phrase translation probability, baseline phrase count, incremental phrase translation probability and incremental phrase count, respectively. e and f indicate the source and target phrases in the context. Entries which are not shared are simply copied to the merged phrase-table. Again the updates applied to the inverse and direct translation probabilities (in equation 2) are motivated by the aim to approximate the probabilities which would ideally have been generated by full retraining.

Using these merging techniques, we are able to efficiently merge the smaller incremental models to the larger baseline models to simulate the full retraining effect. Also since the actual training only happens on the smaller batches of selected data, it is computationally much faster than full retraining at every step. Note that (1) and (2) ensure that the updated lex_{merged} and ϕ_{merged} are true probabilities such that the conditions $0 \leq lex_{merged} \leq 1$ and $0 \leq \phi_{merged} \leq 1$ hold true and both probabilities sum up to 1.

3.4 Reordering Model Merging

While the basic idea behind phrase-table merging could also be applied to the re-ordering model, we choose a simpler option for re-ordering model updates. Once a new reordering model is computed on the selected batch of supplementary data, every entry is compared to the baseline reordering table, and only new entries are added to it to generate a merged RoM. For the shared entries the reordering probabilities are retained as in the baseline model. Not only does this allow faster merging of reordering models but also ensures that for common entries 'in-domain' reordering is preferred over the 'out-of-domain' ones.

3.5 Language Models

As already stated, we use statically trained LMs for all our experiments. We use 5-gram models with modified Kneser-Ney smoothing (Kneser and Ney, 1995) and interpolated back-off. With such models adding a single n-gram into an existing model affects the probability and back-off values of all n-grams in the model. Hence incremental merging of LMs can not be achieved as easily as in the case of TrMs. Accordingly, in the current experiments we use statically estimated interpolated LMs. Three different 5-gram LMs are estimated on monolingual German and French forum data, the target side of the entire TM data and supplementary datasets, respectively. We then combine them using linear interpolation. The interpolation weights are estimated by running expectation maximization (EM) (Dempster et al., 1977) on the target side of the devset.

4 Experimental Setup

In this section, we introduce the datasets, tools and software used in our experiments. We also present the experimental setups for comparing our data selection and model merging technique with established techniques in the literature.

4.1 Datasets

The training data for our baseline systems consists of En–De and En–Fr bilingual datasets in the form of Symantec TMs. Monolingual Symantec forum posts in German and French along with the target side of the TM training data serve as language modelling data. In addition, we also have about 1.1M monolingual sentences from the English forum data which is used to create the LM with respect to which the supplementary datasets are ranked. The dev and testsets are randomly selected from this English forum dataset, ensuring that they are representative of the forum data in terms of different statistics, and manually translated by professional translators. Table 1 reports the number of sentences in the different datasets along with the average sentence length (A.S.L.) used for all our experiments.

Apart from the 'in-domain' training data, we also used the following three freely available parallel corpora as supplementary datasets for our experiments.

1. Europarl (Koehn, 2005) version 6: a parallel corpus comprising of the proceedings of the European Parliament.
2. News Commentary Corpus: released as a part of the WMT 2011 Translation Task.[2]
3. OpenSubtitles2011 Corpus:[3] a collection of documents released as part of the OPUS corpus (Tiedemann, 2009).

[2] http://www.statmt.org/wmt11/translation-task.html
[3] http://www.opensubtitles.org/

	Dataset	En–De			En–Fr		
		Sent Count	En A.S.L	De A.S.L	Sent Count	En A.S.L	Fr A.S.L
Bi-text	Symantec TM	832,723	12.86	12.99	702,267	12.42	14.86
	Dev	1,000	12.91	12.20	1,000	12.91	14.99
	Test	1,031	12.75	11.99	1,031	12.75	14.69
Supp-lement.	Europarl	1,721,980	27.48	26.11	1,809,563	27.34	30.35
	News-Comm.	135,758	24.34	24.98	115,085	24.79	29.06
	Open-Subs.	4,649,247	7.61	7.16	12,483,718	8.61	8.17
Mono-lingual	English Forum	Sent Count		1,129,749	A.S.L		12.48
	German Forum			42,521			11.78
	French Forum			41,283			14.82

Table 1: Number of sentences and A.S.L. for training, dev and testsets, and target language forum datasets.

4.2 Software and Tools

The SMT system used in our experiments is based on the standard phrase-based SMT toolkit: Moses (Koehn et al., 2007). Word alignment is performed with Giza++ (Och and Ney, 2003) using the 'grow-diag-final' heuristic. The lexical, phrase and reordering tables are built on the word alignments using the Moses training scripts. The standard training scripts are modified to augment the count information in the lexical tables. The maximum phrase-length is set to 7. The automatic metric used to evaluate translation quality in the incremental setup is BLEU (Papineni et al., 2002), although the selection algorithm is general enough to accommodate any other evaluation metric. The feature weights for the log-linear combination of the features are tuned using Minimum Error Rate Training (MERT) (Och, 2003) on the devset in terms of BLEU. For the LMs used in each of our models, we used the IRSTLM (Federico et al., 2008) language modelling toolkit for estimation as well as for the linear interpolation weight computation. In order to merge interpolated weights into a single LM, we used the weighted mixing mechanism provided by SRILM (Stolcke, 2002). Once the LMs are estimated, they are binarized using KenLM (Heafield, 2011) to ensure faster multi-threaded access during the decoding phase. Finally, translations of the testsets in every phase of our experiments are evaluated using the BLEU and TER (Snover et al., 2006) metrics.

4.3 Experiments

The primary objective of the experiments is relevant data selection from supplementary parallel training data for domain adaptation. In order to evaluate the effect of our data selection technique, we compare our method with established methods in the literature. Additionally we also compare existing mechanisms to combine the selected data with the 'in-domain' data.

4.3.1 Baseline

Prior to running the incremental data selection experiments, the baseline TrMs were estimated on the 'in-domain' (Symantec TMs) datasets. The standard Moses training scripts were modified to augment the actual word counts to the existing lexical table format. The scoring mechanism of Moses was adjusted to handle the variation in the lexical table formats. This modified version of the training scripts was then used to estimate the baseline TrM only on the Symantec TM data. Three different interpolated LMs were estimated using the technique reported in Section 4.2 each with the target side of different supplementary datasets. For experiments with a particular

supplementary dataset, we used the respective interpolated LM as the baseline for fair comparison. Therefore, the baseline for each set of experiments (for every supplementary dataset) had the same TrM but different LMs. The Giza++ alignments for each of the supplementary datasets were pre-computed and used in the iterative setup.

4.3.2 Data Selection Experiments

To evaluate the quality of our data selection approach we compare the following four data selection techniques:

1. Full: The naive approach of using the full data for adaptation.
2. PP: Data selection by ranking the supplementary data using normalised perplexity with respect to the target domain and thresholding (Foster and Kuhn, 2007).
3. PPD: Using difference in cross-entropy between in-domain and out-domain datasets to rank supplementary data followed by thresholding (Axelrod et al., 2011).
4. TQS: Translation quality-based data selection (cf. Section 3).

In order to rank the supplementary dataset sentences by normalised perplexity (PP), we used a LM trained on the English forum data as the target-domain LM. For each sentence on the source side of the supplementary dataset, its perplexity is computed on the target-domain LM. Perplexity is found to have a strong correlation with the sentence length and hence we normalize the perplexity values by sentence length. Once the perplexity values are computed, they are sued to sort the sentences thereby ensuring that the sentences which are closest to the target domain appear at the top. The data selection is performed by selecting the top N sentences from this ranked corpus. The value of N is set by the number of sentences selected using our TQS method for fair comparison.

Following the technique presented in (Axelrod et al., 2011), the difference of cross-entropy based ranking (PPD) requires an out-of-domain LM in addition to the existing in-domain LM. An out-domain LM is built on a randomly selected sub-sample of the supplementary training data having the same number of sentences and the same vocabulary as the in-domain LM. A similar set of in-domain and out-domain language models are also built on the target language side using the German and the French forum datasets for in-domain LMs and random samples from supplementary datasets as the out-of-domain LMs. Eventually each supplementary data sentence is ranked according to the difference in cross-entropy with respect to the in-domain and out-of-domain LMs summed over both the source and the target languages. Like in the case of PP, the sentences are sorted by these scores and the lowest scoring sentences are selected. However in contrast to the previous case, this ranking biases towards the sentences which are both like the in-domain sentences and unlike the average of out-of-domain sentences.

The sentences selected using our translation quality-based technique (TQS) are selected in batches using the approach described in Section 3.2. In order to speed up the translation process in the iterative framework, we utilise the multi-threaded feature of the Moses decoder. Furthermore, the merged phrase-table and the reordering models were filtered using the source side of the devset to reduce memory requirements as well as ensure faster decoding. While the other two ranking techniques require the selection of a thresholding value to select an appropriate subset of the supplementary data for adaptation, our technique is designed to automatically select a subset of the same. Therefore we use the number of sentences selected by TQS methods as the thresholding value for PP and PPD selection schemes.

4.3.3 Data Combination Experiments

Once the supplementary data is selected, this data needs to be combined with the in-domain training data for adaptation. In addition to the naive approach of concatenating the selected data to the in-domain datasets and retraining the model, we investigate three configurations of model combination based on existing methods in the SMT literature.

1. Conc: The naive approach of concatenating the selected data with the in-domain data and retraining the SMT model (Foster et al., 2010).
2. Multiple phrase-table (MPT): Creating separate phrase-tables for the in-domain and the selected data and using the multiple decoding path feature of the Moses decoder (Koehn and Schroeder, 2007).
3. Linear Interpolation (LinMix): Using a weighted linear interpolation to combine the individual phrase-tables (Foster and Kuhn, 2007).
4. PTM: Using the phrase-table merging technique reported in this paper.

In the concatenation approach (Conc), the selected supplementary data is added to the in-domain training data and a new TrM is retrained from scratch. This model is then tuned using the devset and finally tested using the testset to reveal the effect of adaptation. The Multiple phrase-table (MPT) approach requires training separate phrase-tables on the in-domain and selected data and combining them using the multiple decoding feature of the Moses decoder. The decoder uses both phrase-tables to score each of the translation options during the decoding phase. The phrase pairs which occur in both the phrase-tables are separately scored using their respective phrase-tables. In the linear interpolation approach (linmix) the two phrase-tables are combined using weights in a linear interpolation scheme. In order to learn the interpolation weights, LMs are constructed on the target side of the in-domain training set and the selected supplementary data. These LMs are then interpolated using EM on the target side of the devset to learn the optimal mixture weights. These weights are subsequently used to combine the individual feature values for every phrase pair from two phrase-tables using the formula in (3).

$$p_{linmix}(s|t) = \lambda p_{in}(s|t) + (1-\lambda)p_{out}(s|t) \tag{3}$$

where $p_{in}(s|t)$ and $p_{out}(s|t)$ are the feature values of individual phrase pairs from the in-domain and out-of-domain phrase-tables, respectively. λ is the tunable weight between 0 and 1.

The phrase-table merging (PTM) technique outlined in Section 3 was developed to rapidly combine incremental and baseline TrMs to aid our iterative data selection method. However, here we use it as an alternative technique to combine the in-domain and out-of-domain phrase-tables. While the basic idea behind this technique is similar to that of linear interpolation, in our technique each feature is weighted according to its frequency in the respective phrase-tables in contrast to using a global weight for every feature in LinMix. Following model combination, all the models are tuned using MERT on the devset.

5 Results and Analysis

As stated in Section 4.2, the incremental data selection process is performed by evaluating translation quality in terms of BLEU scores on the devset data. Table 2 reports the baseline scores, the best scores and the number of sentences selected during the process of incremental data selection on the devset. Alongside the number of selected sentences, the percentage figures indicate the proportion of the selected sentences with respect to the entire size of the supplementary datasets as reported in Table 1. Note that the BLEU scores reported in this table

are all non-MERT scores and the supplementary data was combined with the baseline model using the PTM method.

Lang-Pair	Model	Europarl		Open-Subtitles		News-Commentary	
		BLEU	Sent #	BLEU	Sent #	BLEU	Sent #
En–De	Baseline	22.97	663,127	22.94	1,464,798	22.91	15,473
	Best	*__24.17__	38.51%	*__24.33__	31.51%	*__23.34__	11.39%
En–Fr	Baseline	31.33	571,736	31.72	1,705,273	31.16	52,797
	Best	*__31.85__	31.60%	*__32.77__	13.66%	__31.43__	45.88%

Table 2: BLEU scores on devset using incremental TrM updates and number of sentences selected.$*$ indicates statistically significant improvement at p$\leq$0.05, best scores are in bold.

The scores in Table 2 clearly show the improvements observed on the devset for both language pairs across all supplementary datasets. While the improvements obtained using the Europarl (EP) and Open-Subtitles (OPS) corpora are statistically significant at the p=0.05 level using bootstrap resampling (Koehn, 2004) for both language pairs, the News-Commentary (NC) corpus only provides significant improvement for En–De translations. Compared to the improvements obtained on the other two sets, NC improvements are much lower, which could be attributed to the smaller size of the corpus and hence consequentially the smaller size of the selected dataset. As already stated in Section 4.3.2, the number of selected sentences as reported in Table 2 for each supplementary dataset is used as the threshold values for data selection for the PP and PPD ranking methods.

5.1 Data Selection Results

The primary objective of our approach being data selection from supplementary sources, we first report the results of our data selection methods in comparison to the other data selection techniques described in Section 4.3.2. In this phase, the selected supplementary data is concatenated with the in-domain training data to train new TrMs which are then tuned using MERT on the devset. Table 3 reports the BLEU and TER scores for the different data selection techniques in addition to our own method.

	System	Europarl		Open-Subs.		News-Comm.	
		BLEU	**TER**	**BLEU**	**TER**	**BLEU**	**TER**
En–De	Baseline	21.98	0.6436	22.56	0.6312	22.10	0.6394
	PP	$*$22.69	0.6233	$*$23.03	**0.6100**	22.24	**0.6257**
	PPD	$*$22.80	0.6211	$*$23.14	0.6127	22.34	0.6405
	Full	$*$22.58	0.6246	22.67	0.6189	22.20	0.6279
	TQS	$*$§**23.10**	**0.6190**	$*$§ **23.50**	0.6122	**22.47**	0.6292
En–Fr	Baseline	31.87	0.5603	32.52	0.5474	31.82	0.5569
	PP	$*$32.73	0.5506	$*$33.18	0.5452	32.28	**0.5435**
	PPD	$*$§33.03	0.5485	$*$33.26	**0.5371**	$*$§32.38	0.5527
	Full	32.39	0.5570	32.96	0.5498	31.59	0.5545
	TQS	$*$§†‡**33.58**	**0.5410**	$*$§**33.56**	0.5424	$*$§**32.56**	0.5503

Table 3: Testset BLEU and TER scores using data selection methods. $*, \dagger, \ddagger, \S$ indicates statistically significant improvement in BLEU over baseline, PP, PPD and Full datasets, respectively.

The scores reported in Table 3 show that adding additional supplementary data to the in-domain TrMs improve translation quality scores over the baseline in nearly all cases (quality

only deteriorates over the baseline when the Full NC data is added to the En–Fr training data). The actual data selection methods (PP, PPD and TQS) provide improvements on the baseline scores as well as on the Full scores, indicating the success of the data selection process. Comparing the translation quality scores between PP, PPD and TQS, we observe that while the PPD scores are sightly better than the PP scores, the TQS method performs best, consistently improving over the other two data selection methods in terms of BLEU scores. Using EP as the supplementary corpus the TQS method provides improvements of 1.12 absolute (5.1% relative) and 1.71 absolute (5.37% relative) BLEU points over the baseline scores for En–De and En–Fr translations, respectively. With the OPS corpus, the improvement figures are 0.94 absolute (4.17% relative) and 1.04 absolute (3.2% relative) BLEU points for En–De and En–Fr translations, respectively. For the NC corpus, the method improves the baseline scores by 0.37 absolute (1.67% relative) and 0.74 absolute (2.33% relative) BLEU points for En–De and En–Fr translation, respectively. While the EP and OPS improvements are statistically significant at $p \leq 0.05$ level for both language pairs, for NC only the En–Fr improvement is statistically significant. Although the TQS method provides better scores than the PP and PPD methods on all counts, the differences are not statistically significant in most cases, except for En–Fr improvements using the EP dataset. However, when compared to the Full scores, the TQS method provides statistically significant improvements for nearly all the cases.

5.2 Data Combination Results

The results reported in Table 3 use the Conc approach (cf. Section 4.3.3) to combine the additional data to the in-domain dataset. However, combining in-domain and out-domain datasets using this approach may not always lead to the best results as is evident from the literature (Foster and Kuhn, 2007; Banerjee et al., 2011). Hence in the second phase we compare the translation quality achieved by using the different combination methods explained in Section 4.3.3. Since the data selected by the TQS method was the best-performing dataset using the BLEU scores as per Table 3, we report the results of the different data combination experiments on this particular set only. Table 4 reports the effect of different data combination methods on translation score using data selected by the TQS method.

	System	Europarl		Open-Subs.		News-Comm.	
		BLEU	TER	BLEU	TER	BLEU	TER
En–De	Conc	23.10	0.6190	23.50	0.6122	22.47	0.6292
	MPT	23.15	**0.6134**	23.25	0.6145	21.75	0.6349
	PTM	23.17	0.6161	23.78	0.6116	22.58	0.6270
	LinMix	**23.23**	0.6161	∗†**23.80**	**0.6092**	†**22.66**	**0.6249**
En–Fr	Conc	33.58	0.5410	33.56	0.5424	32.56	0.5503
	MPT	33.31	0.5418	33.34	0.5456	32.20	0.5453
	PTM	33.30	0.5473	33.71	**0.5360**	32.66	**0.5324**
	LinMix	**33.75**	**0.5391**	†**33.84**	0.5398	†**32.79**	0.5494

Table 4: Testset BLEU and TER scores using data combination methods. $*, \dagger, \ddagger$ indicates statistically significant improvement in BLEU over Conc,MPT, and PTM methods, respectively.

The translation quality scores in Table 4 confirm our assumption that concatenation is not always the best option to combine multiple datasets. The results show weighted linear interpolation to be the best-performing system for different datasets and language pairs. However, the difference in the evaluation scores between the different combination techniques are mostly

statistically insignificant. MPT is found to work better than Conc in some of the cases (for EP datasets in En–De and En–Fr) but in most cases is poorer than all the other methods. Weighted linear interpolation is known to work well in multi-domain phrase-table combination (Banerjee et al., 2011) and our experiments confirm the observation. Interestingly, using our phrase-table merging method (PTM) for model combination seems to work reasonably well for all the different datasets and language pairs. While it does not perform the best, it certainly performs at par with the other combination techniques experimented with, the differences being statistically insignificant in all cases.

Using the MPT configuration has a major advantage over the Conc approach in keeping the in-domain and out-domain phrase-tables separate. While this can really be an effective choice in some cases, this model has larger number of parameters which are difficult to optimize using MERT (Chiang et al., 2009). The linear interpolation mechanism avoids the large parameter setting by combining features from multiple tables into a single table. However, this requires the estimation of the interpolation weights and it is not very straightforward to optimize the linear weights directly in terms of translation quality. While the LinMix method uses global weights for all phrase pairs, the PTM method uses different weights based on the frequency of occurrence in each corpora. This avoids the problem of linear interpolation weight optimization as well as the large parameter setting. In our experimental setting, this method slightly underperforms with respect to LinMix, but the difference is statistically insignificant.

5.3 Combining Data Selection and Model Combination

The results in Table 4 clearly indicate that linear interpolation of phrase-tables provides the best scores among different data combination techniques at least for the datasets under consideration. Hence in the final phase we present the results on different data selection methods using linear interpolated mixture models as the combination technique in Table 5

	System	Europarl		Open-Subs.		News-Comm.	
		BLEU	**TER**	**BLEU**	**TER**	**BLEU**	**TER**
En–De	PP	22.96	0.6212	23.13	0.6117	22.33	**0.6237**
	PPD	23.05	0.6225	23.26	0.6188	22.41	0.6258
	Full	22.73	0.6219	22.83	0.6177	22.25	0.6319
	TQS	§**23.23**	**0.6161**	†‡§**23.80**	**0.6092**	**22.66**	0.6249
En–Fr	PP	33.00	0.5476	33.25	0.5412	32.41	0.5487
	PPD	33.29	0.5429	33.32	**0.5379**	32.62	**0.5481**
	Full	32.80	0.5467	33.01	0.5518	31.96	0.5558
	TQS	†§**33.75**	**0.5391**	†‡§**33.84**	0.5398	§**32.79**	0.5494

Table 5: Testset BLEU and TER scores with LinMix as combination method. †, ‡, § indicate statistically significant BLEU improvements over PP, PPD and Full scores.

Using linear interpolation to combine the models built on different datasets results in a more-or-less uniform improvement in all translation quality scores for all datasets and language directions when compared to the results in Table 3. The data selected using the TQS method provides statistically significant improvements over the baseline scores as well as those using the Full dataset. Furthermore, the TQS scores are now significantly better than the PP and PPD scores for the En–Fr translation on both EP and the OPS datasets and for the En–De translations on the OPS dataset. However, the improvements are still not statistically significant for the other datasets and language pair combinations.

The overall results in Tables 3 and 5 strongly suggest the success of data selection as an adaptation technique. While adding supplementary training data widens the coverage of the TrMs, thus reducing the number of untranslated words in the translations, it also provides richer lexical translation probabilities for some phrases and words which although present in the baseline models were sparsely represented. Furthermore, we have empirically shown that our translation quality based data selection method consistently outperforms perplexity ranking-based data selection approaches. While the TQS method directly uses translation quality to select supplementary sentences, the PP and PPD methods rely on the perplexity or cross-entropy for the same task. Since perplexity or cross-entropy have low correlation with actual translation quality, sentences selected using such techniques are not guaranteed to improve translation quality. In contrast the TQS method only selects groups of sentences which improve translation quality, which is our overall objective. Hence, while using the PP or the PPD method all the top sentences from the supplementary data are chosen, the TQS method discards a few of the top batches as they fail to improve translation quality on the devset in the iterative framework.

Conclusion and Future Work

In this paper we have introduced a novel method for supplementary data selection for domain adaptation of SMT systems. Sentence pairs are selected incrementally in batches from the supplementary out-of-domain bitext data and added to the baseline system and evaluated in terms of BLEU scores on a devset. A batch is selected only if it results in improved BLEU scores. Once all the batches in a supplementary dataset are processed, the batches that pass the selection are combined to produce the selected parallel data for domain adaptation. The data selected using this method is found to outperform other existing data selection methods in terms of translation quality on an unseen testset and for a number of supplementary datasets. Additionally we also present a phrase-table merging technique that is developed to facilitate iterative data selection. This technique is effectively used to combine multiple phrase-tables from different domains and performs on a par with other existing techniques in the field. Our experiments also show that data selection is an effective adaptation technique for translating user-generated content using TM based training data. Moreover, the relative comparison of different model or data combination strategies reveals that concatenating supplementary data to existing in-domain data may not always yield the best results and is outperformed by a linear interpolation approach.

Extending the concept of iterative incremental training to LMs is one of the prime future directions for this work. Further investigation into methods to avoid the overfitting issue is also necessary. Finally, some analysis on the effect of batch size on translation quality in an iterative setting would also be an interesting future direction. Furthermore, the phrase-table merging technique could effectively be utilised for incremental training of TrMs.

Acknowledgments

This work is supported by Science Foundation Ireland (Grant No. 07/CE/I1142) as part of the Centre for Next Generation Localisation (www.cngl.ie) at Dublin City University. We thank the reviewers for their insightful comments.

References

Axelrod, A., He, X., and Gao, J. (2011). Domain adaptation via pseudo in-domain data selection. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*,

EMNLP '11, pages 355–362, Edinburgh, UK.

Axelrod, A. E. (2006). Factored language models for statistical machine translation. Master's thesis, University of Edinburgh.

Banerjee, P., Naskar, S. K., Roturier, J., Way, A., and van Genabith, J. (2011). Domain Adaptation in Statistical Machine Translation of User-Forum Data using Component Level Mixture Modelling. In *Proceedings of the Thirteenth Machine Translation Summit*, pages 285–292, Xiamen, China.

Banerjee, P., Naskar, S. K., Roturier, J., Way, A., and van Genabith, J. (2012). Domain adaptation in smt of user-generated forum content guided by oov word reduction: Normalization andor supplementary data? In *Proceedings of the 16th Annual Conference of the European Association for Machine Translation (EAMT-2012)*, pages 169–176, Trento, Italy.

Callison-Burch, C., Bannard, C., and Schroeder, J. (2005). A compact data structure for searchable translation memories. In *Proceedings of 10th Annual Conference of European Association for Machine Translation (EAMT-2005)*, pages 59–65, Budapest, Hungary.

Cappé, O. and Moulines, E. (2009). On-line expectation-maximization algorithm for latent data models. *Journal of the Royal Statistical Society Series B*, 71(3):593–613.

Chiang, D., Knight, K., and Wang, W. (2009). 11,001 new features for statistical machine translation. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, NAACL '09, pages 218–226, Boulder, Colorado.

Daume III, H. and Jagarlamudi, J. (2011). Domain adaptation for machine translation by mining unseen words. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 407–412, Portland, Oregon, USA.

Dempster, A. P., Laird, N. M., and Rubin, D. B. (1977). Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society, B*, 39:1–38.

Eck, M., Vogel, S., and Waibel, A. (2004). Language model adaptation for statistical machine translation based on information retrieval. In *Proceedings of 4th International Conference on Language Resources and Evaluation, (LREC 2004)*, pages 327–330, Lisbon, Portugal.

Federico, M., Bertoldi, N., and Cettolo, M. (2008). IRSTLM: an open source toolkit for handling large scale language models. In *Interspeech 2008: 9th Annual Conference of the International Speech Communication Association*, pages 1618–1621, Brisbane, Australia.

Foster, G., Goutte, C., and Kuhn, R. (2010). Discriminative instance weighting for domain adaptation in statistical machine translation. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, EMNLP '10, pages 451–459, Cambridge, MA.

Foster, G. and Kuhn, R. (2007). Mixture-model adaptation for SMT. In *ACL 2007: Proceedings of the Second Workshop on Statistical Machine Translation*, pages 128–135, Prague, Czech Republic.

Gao, Q., Lewis, W., Quirk, C., and Hwang, M.-Y. (2011). Incremental Training and Intentional Over-fitting of Word Alignment. In *Proceedings of the Thirteenth Machine Translation Summit*, pages 106–113, Xiamen, China.

Heafield, K. (2011). Kenlm: Faster and smaller language model queries. In *Proceedings of the Sixth Workshop on Statistical Machine Translation*, pages 187–197, Edinburgh, UK.

Hildebrand, A. S., Eck, M., Vogel, S., and Waibel, A. (2005). Adaptation of the Translation Model for Statistical Machine Translation based on Information Retrieval. In *10th EAMT Conference: Practical Applications of Machine Translation, Conference Proceedings*, pages 119–125, Budapest, Hungary.

Kneser, R. and Ney, H. (1995). Improved backing-off for m-gram language modeling. In *In Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, volume I, pages 181–184, Detroit, Michigan.

Koehn, P. (2004). Statistical Significance Tests for Machine Translation Evaluation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing, (EMNLP 2004)*, pages 388–395, Barcelona, Spain.

Koehn, P. (2005). Europarl: A Parallel Corpus for Statistical Machine Translation. In *MT Summit X: The 10th Machine Translation Summit*, pages 79–86, Phuket, Thailand.

Koehn, P., Hoang, H., Birch, A., Callison-Burch, C., Federico, M., Bertoldi, N., Cowan, B., Shen, W., Moran, C., Zens, R., Dyer, C., Bojar, O., Constantin, A., and Herbst, E. (2007). Moses: open source toolkit for statistical machine translation. In *ACL 2007, Proceedings of the Interactive Poster and Demonstration Sessions*, pages 177–180, Prague, Czech Republic.

Koehn, P. and Schroeder, J. (2007). Experiments in domain adaptation for statistical machine translation. In *ACL 2007: Proceedings of the Second Workshop on Statistical Machine Translation*, pages 224–227, Prague, Czech Republic.

Levenberg, A., Callison-Burch, C., and Osborne, M. (2010). Stream-based translation models for statistical machine translation. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, HLT '10, pages 394–402, Los Angeles, CA.

Och, F. J. (2003). Minimum error rate training in statistical machine translation. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics - Volume 1*, pages 160–167, Sapporo, Japan.

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

Papineni, K., Roukos, S., Ward, T., and Zhu, W.-J. (2002). Bleu: a method for automatic evaluation of machine translation. In *40th Annual Meeting of the Association for Computational Linguistics, (ACL 2002)*, pages 311–318, Philadelphia, Pennsylvania.

Roturier, J. and Bensadoun, A. (2011). Evaluation of MT Systems to Translate User Generated Content. In *Proceedings of the Thirteenth Machine Translation Summit*, pages 244–251, Xiamen,China.

Snover, M., Dorr, B., Schwartz, R., Micciulla, L., and Makhoul, J. (2006). A study of translation edit rate with targeted human annotation. In *Proceedings of Association for Machine Translation in the Americas*, pages 223–231, Cambridge, MA.

Stolcke, A. (2002). SRILM–An extensible language modeling toolkit. In *ICSLP 2002, Interspeech 2002: 7th International Conference on Spoken Language Processing*, pages 901–904, Denver, CO.

Tiedemann, J. (2009). News from OPUS - A collection of multilingual parallel corpora with tools and interfaces. In Nicolov, N., Bontcheva, K., Angelova, G., and Mitkov, R., editors, *Recent Advances in Natural Language Processing*, volume V, pages 237–248.

Text Reuse Detection Using a Composition of Text Similarity Measures

Daniel Bär[1] *Torsten Zesch*[1,2] *Iryna Gurevych*[1,2]
(1) Ubiquitous Knowledge Processing Lab (UKP-TUDA)
Department of Computer Science, Technische Universität Darmstadt
(2) Ubiquitous Knowledge Processing Lab (UKP-DIPF)
German Institute for Educational Research and Educational Information
www.ukp.tu-darmstadt.de

ABSTRACT

Detecting text reuse is a fundamental requirement for a variety of tasks and applications, ranging from journalistic text reuse to plagiarism detection. Text reuse is traditionally detected by computing similarity between a source text and a possibly reused text. However, existing text similarity measures exhibit a major limitation: They compute similarity only on features which can be derived from the *content* of the given texts, thereby inherently implying that any other text characteristics are negligible. In this paper, we overcome this traditional limitation and compute similarity along three characteristic dimensions inherent to texts: *content*, *structure*, and *style*. We explore and discuss possible combinations of measures along these dimensions, and our results demonstrate that the composition consistently outperforms previous approaches on three standard evaluation datasets, and that text reuse detection greatly benefits from incorporating a diverse feature set that reflects a wide variety of text characteristics.

TITLE AND ABSTRACT IN GERMAN

Erkennung von Textwiederverwendung durch Komposition von Textähnlichkeitsmaßen

Die Frage, ob und in welcher Weise Texte in abgewandelter Form wiederverwendet werden, ist ein zentraler Aspekt bei einer Reihe von Problemstellungen, etwa im Rahmen journalistischer Tätigkeit oder als Mittel zur Plagiatserkennung. Textwiederverwendung wird traditionell ermittelt durch Berechnen von Textähnlichkeit zwischen einem Ursprungstext und einem potentiell wiederverwendeten Text. Bestehende Textähnlichkeitsmaße haben jedoch die starke Einschränkung, dass sie Ähnlichkeit nur anhand von Eigenschaften berechnen, die vom *Inhalt* der gegebenen Texte abgeleitet werden können, und somit implizieren, dass jegliche andere Textcharacteristika vernächlässigbar sind. In dieser Arbeit berechnen wir Textähnlichkeit anhand von drei Dimensionen: *Inhalt*, *Struktur* und *Stil*. Wir untersuchen mögliche Kombinationen von Maßen entlang dieser Dimensionen, und zeigen deutlich anhand der Ergebnisse auf drei etablierten Evaluationsdatensätzen, dass die Komposition generell bessere Ergebnisse liefert als bestehende Ansätze, und dass die Bestimmung von Textwiederverwendung stark von einem breiten Spektrum an Textcharacteristika profitiert.

KEYWORDS: text similarity, text reuse, plagiarism, paraphrase.
KEYWORDS IN GERMAN: Textähnlichkeit, Textwiederverwendung, Plagiat, Paraphrase.

Proceedings of COLING 2012: Technical Papers, pages 167–184,
COLING 2012, Mumbai, December 2012.

1 Introduction

Text reuse is a common phenomenon and arises, for example, on the Web from mirroring texts on different sites or reusing texts in public blogs. In other text collections such as content authoring systems of communities or enterprises, text reuse arises from keeping multiple versions, copies containing customizations or reformulations, or the use of template texts (Broder et al., 1997).

Problems with text reuse particularly arise in settings where systems are extensively used in a collaborative manner. For example, *wikis* are web-based, collaborative content authoring systems which offer fast and simple means for adding and editing content (Leuf and Cunningham, 2001). At any time, users can modify content already present in the wiki, augment existing texts with new facts, ideas, or thoughts, or create new texts from scratch. However, when users contribute to wikis, they need to avoid content duplication. This requires comprehensive knowledge of what content is already present in the wiki, and what is not. As wikis are traditionally growing fast, this is hardly feasible, though. To remedy this issue, we aim at supporting authors of collaborative text collections by means of automatic text reuse detection. We envision a semi-supervised system that informs a content author of potentially pre-existing instances of text reuse, and then lets the author decide how to proceed, e.g. to merge both texts.

Detecting text reuse has been studied in a variety of tasks and applications, e.g. the detection of journalistic text reuse (Clough et al., 2002), the identification of rewrite sources for ancient literary texts (Lee, 2007), or the analysis of text reuse in blogs and web pages (Abdel-Hamid et al., 2009). Another common instance of text reuse is plagiarism, with the additional constraint that the reuse needs to be unacknowledged. Near-duplicate detection is also a broad field of related work where the detection of text reuse is crucial, e.g. in the context of web search and crawling (Hoad and Zobel, 2003; Henzinger, 2006; Manku et al., 2007). Prior work, however, mainly utilizes fingerprinting and hashing techniques (Charikar, 2002) for text comparison rather than methods from natural language processing.

A common approach to text reuse detection is to compute similarity between a source text and a possibly reused text. A multitude of text similarity measures have been proposed for computing similarity based on surface-level and/or semantic features (Mihalcea et al., 2006; Landauer et al., 1998; Gabrilovich and Markovitch, 2007). However, existing similarity measures typically exhibit a major limitation: They compute similarity only on features which can be derived from the *content* of the given texts. By following this approach, they inherently imply that the similarity computation process does not need to take any other text characteristics into account.

In contrast, we propose that text reuse detection indeed benefits from also assessing similarity along other text characteristics (*dimensions*, henceforth). We follow empirical evidence by Bär et al. (2011) and focus on three characteristic similarity dimensions inherent to texts: *content*, *structure*, and *style*. Figure 1 shows an example of text reuse taken from the Wikipedia Rewrite Corpus (see Section 3.1) where parts of a given source text have been reused either verbatim or by using similar words or phrases. As the example illustrates, the process of creating reused text includes a revision step in which the editor has a certain degree of freedom on how to reuse the source text. This kind of similarity is detectable by content-centric text similarity measures. However, the editor has further split the source text into two individual sentences and changed the order of the reused parts. For detecting the degree of similarity of such a revision, text similarity measures for *structural similarity* are necessary. Additionally, the given texts exhibit a certain degree of similarity with respect to stylistic features, e.g. vocabulary richness.[1] In

[1]The type-token ratio (Templin, 1957) of the texts is .79 and .71, respectively.

Source Text. *PageRank is a link analysis algorithm used by the Google Internet search engine that assigns a numerical weighting to each element of a hyperlinked set of documents, such as the World Wide Web, with the purpose of "measuring" its relative importance within the set.*

Text Reuse. *The PageRank algorithm is used to designate every aspect of a set of hyperlinked documents with a numerical weighting. It is used by the Google search engine to estimate the relative importance of a web page according to this weighting.*

Figure 1: Example of text reuse taken from the Wikipedia Rewrite Corpus (Clough and Stevenson, 2011). Various parts of the source text have been reused, either verbatim (underlined) or using similar words or phrases (wavy underlined). However, the editor has split the source text into two individual sentences and changed the order of the reused parts.

order to use such features as indicators of text reuse, we propose to further include measures of *stylistic similarity*.

In this paper, we thus overcome the traditional limitation of text similarity measures to *content* features. In contrast, we adopt ideas of seminal studies by cognitive scientists (Tversky, 1977; Goodman, 1972; Gärdenfors, 2000) and discuss the role of three similarity dimensions for the task of text reuse detection: *content*, *structure*, and *style*, as proposed in our previous work (Bär et al., 2011). In Section 2, we report on a multitude of text similarity measures from these dimensions that we used for our experiments. In Section 3, we demonstrate empirically that text reuse can be best detected if measures are combined across dimensions, so that a wide variety of text characteristics are taken into consideration. Our approach consistently outperforms previous work on three standard evaluation datasets, and demonstrates the advantage of integrating text characteristics other than *content* into the similarity computation process.

2 Text Similarity Measures

In this section, we report on a variety of similarity measures which we used to compute similarity along characteristic dimensions inherent to texts.[2] We classify them into measures for *content similarity*, *structural similarity*, and *stylistic similarity*, as proposed by Bär et al. (2011).

2.1 Content Similarity

Probably the easiest way to reuse text is verbatim copying. It can be detected by using string measures which operate on substring sequences. The *longest common substring* measure (Gusfield, 1997) compares the length of the longest contiguous sequence of characters between two texts, normalized by the text lengths. However, the editorial process in journalistic text reuse or the attempt to obfuscate copying in plagiarism may shorten the longest common substring considerably, e.g. when words are inserted or deleted, or parts of reused text appear in a different order. The *longest common subsequence* measure (Allison and Dix, 1986) drops the contiguity requirement and allows to detect text reuse in case of word insertions/deletions. *Greedy String Tiling* (Wise, 1996) further allows to deal with reordered parts of reused text as it determines a set of shared contiguous substrings between two given documents, each substring thereby being a match of maximal length. A multitude of other string similarity measures have been proposed which view texts as sequences of characters and compute their degree of

[2] In addition, we release an open-source framework which contains implementations of all discussed measures in order to stimulate the development of novel measures: `http://code.google.com/p/dkpro-similarity-asl`

distance according to a given metric. We used the following measures in our experiments: *Jaro (1989)*, *Jaro-Winkler* (Winkler, 1990), *Monge and Elkan (1997)*, and *Levenshtein (1966)*.

Starting from the observation that not all words in a document are of equal importance, we further employed a similarity measure which weights all words by a *tfidf* scheme (Salton and McGill, 1983) and computes text similarity as the cosine between two document vectors.

Comparing *word n-grams* (Lyon et al., 2001) is a popular means for comparing lexical patterns between two texts. The more similar the patterns, the more likely is it that text reuse has occurred. After compiling two sets of n-grams, we compared them using the Jaccard coefficient, following Lyon et al. (2001), as well as using the containment measure (Broder, 1997). We tested n-gram sizes for $n = 1, 2, \ldots, 15$, and will use the original system name *Ferret* (Lyon et al., 2004) to refer to the variant with $n = 3$ using the Jaccard coefficient, henceforth.

Following the idea of comparing lexical patterns, we also used a measure which has not yet been considered for assessing content similarity: *character n-gram profiles* (Keselj et al., 2003).[3] We follow the implementation by Barrón-Cedeño et al. (2010) and discard all characters (case insensitive) which are not in the alphabet $\Sigma = \{a, \ldots, z, 0, \ldots, 9\}$, then generate all n-grams on character level, weight them by a *tfidf* scheme, and finally compare the feature vectors of both the rewritten and the source text using the cosine measure. While in the original implementation only $n = 3$ was used, we generalize the measure to $n = 2, 3, \ldots, 15$.

In cases where the editor replaced content words by synonyms, string measures typically fail due to the vocabulary gap. We thus used similarity measures which are capable of measuring semantic similarity between words. We used the following word similarity measures with WordNet (Fellbaum, 1998): *Jiang and Conrath (1997)*, *Lin (1998)*, and *Resnik (1995)*. In order to scale these pairwise word similarity scores to the document level, we follow the aggregation strategy by Mihalcea et al. (2006): First, a directional similarity score $sim_d(T_i, T_j)$ is computed from a text T_i to a second text T_j (Eq. 1). Therefore, for each word w_i in T_i, its best-matching counterpart in T_j is sought ($maxSim(w_i, T_j)$). The similarity scores of all these matches are summed up and weighted according to their inverse document frequency *idf* (Spärck Jones, 1972), then normalized. The final document-level similarity figure is the average of applying this strategy in both directions, from T_i to T_j and vice-versa (Eq. 2).

$$sim_d(T_i, T_j) = \frac{\sum_{w_i} maxSim(w_i, T_j) \cdot idf(w_i)}{\sum_{w_i} idf(w_i)} \quad (1) \qquad sim(T_i, T_j) = \frac{1}{2}\Big(sim_d(T_i, T_j) + sim_d(T_j, T_i)\Big) \quad (2)$$

We also tested text expansion mechanisms with the semantic word similarity measures described above: We used the Moses SMT system (Koehn et al., 2007), trained on Europarl (Koehn, 2005), to translate the original English texts via a bridge language (Dutch) back to English. Thereby, the idea was that in the translation process additional lexemes are introduced which alleviate potential lexical gaps. We computed pairwise word similarity with the measures described above and aggregated according to Mihalcea et al. (2006).

Furthermore, we used the statistical technique *Latent Semantic Analysis (LSA)* (Landauer et al.,

[3] Traditionally, character n-gram profiles have rather been shown successful for authorship attribution. However, the similarity scores of word n-grams and those of character n-gram profiles are highly correlated: Assuming 5 characters per word on average for English texts (Shannon, 1951), we set $n = 3$ for word n-grams and $n = 15$ for character n-grams, and computed Pearson's correlation r between the corresponding similarity scores. We obtained $r = .93$ and $r = .86$ on the datasets introduced in Sections 3.1 and 3.2, respectively, and thus conclude that this measure captures *content similarity* rather than *stylistic similarity*.

1998) for comparing texts. The construction of the semantic space was done using the evaluation corpora (see Section 3). We also used the vector space model *Explicit Semantic Analysis* (ESA) (Gabrilovich and Markovitch, 2007). Besides WordNet, we used two additional lexical-semantic resources for the construction of the ESA vector space: Wikipedia[4] and Wiktionary[5].

2.2 Structural Similarity

As discussed above, we presume that content similarity alone is not a reliable indicator of text reuse. Two independently written texts about the same topic are likely to make use of a common vocabulary to a certain extent. We thus propose to also use measures of structural similarity which compute similarity based on structural aspects inherent to the compared texts.

Stopword n-grams (Stamatatos, 2011) are based on the idea that text reuse often preserves syntactic similarities while exchanging content words. Thus, the measure removes all content words while preserving only stopwords. All n-grams of both texts are then compared using the containment measure (Broder, 1997). We tested n-gram sizes for $n = 2, 3, \ldots, 15$.

For the same reason, we also included *part-of-speech n-grams* in our feature set. Disregarding the actual words that appear in two given texts, computing n-grams along part-of-speech tags allows to detect syntactic similarities between these texts. Again, we tested n-gram sizes for $n = 2, 3, \ldots, 15$, and compared the two sets using the containment measure (Broder, 1997).

We also employed two similarity measures between pairs of words (Hatzivassiloglou et al., 1999). The *word pair order* measure assumes that a similar syntactical structure in reused texts may cause two words to occur in the same order in both texts (with any number of words in between). The complementary *word pair distance* measure counts the number of words which lie between those of a given pair. For each measure, we computed feature vectors for both texts along all shared word pairs and compared the vectors using Pearson's correlation.

2.3 Stylistic Similarity

Measures of stylistic similarity adopt ideas from authorship attribution (Mosteller and Wallace, 1964) or use statistical properties of texts to compute text similarity. The *type-token ratio (TTR)* (Templin, 1957), for example, compares the vocabulary richness of two texts. However, it suffers from sensitivity to variations in text length and the assumption of textual homogeneity (McCarthy and Jarvis, 2010): As a text gets longer, the increase of tokens is linear, while the increase of types steadily slows down. In consequence, lexical repetition causes the TTR value to vary, while it does not necessarily entail that a reader perceives changes in the vocabulary usage. Secondly, textual homogeneity is the assumption of the existence of a single lexical diversity level across a whole text, which may be violated by different rhetorical strategies. *Sequential TTR* (McCarthy and Jarvis, 2010) alleviates these shortcomings. It iteratively computes a TTR score for a dynamically growing text segment until a point of saturation – i.e. a fixed TTR score of .72 – is reached, then starts anew from that position in the text for a new segment. The final lexical diversity score is computed as the number of tokens divided by the number of segments.

Inspired by Yule (1939) who discussed sentence length as a characteristic of style, we also used two simple measures, *sentence length* and *token length*, in our system. These measures compute the average number of tokens per sentence and the average number of characters per token.

[4] www.wikipedia.org
[5] www.wiktionary.org

Text Similarity Feature	WP Rewrite		METER		Webis CPC	
	Acc.	$\bar{F}_1$	Acc.	$\bar{F}_1$	Acc.	$\bar{F}_1$
Majority Class Baseline	.400	.143	.715	.417	.517	.341
Ferret Baseline	.642	.517	.684	.535	.794	.789
Content Similarity						
Character 5-gram Profiles	.642	.537	.715	.417	.753	.742
ESA (Wikipedia)	.474	.323	.711	.484	.760	.753
Greedy String Tiling	.558	.457	.755	.645	.805	.800
Longest Common Substring	.621	.524	.719	.467	.743	.736
Resnik	.632	.500	.715	.417	.666	.656
Word 2-grams Containment	.747	.683	.727	.692	.801	.797
Structural Similarity						
Lemma Pair Distance	.611	.489	.715	.417	.775	.767
Lemma Pair Ordering	.642	.494	.715	.417	.785	.780
POS 3-grams Containment	.642	.554	.731	.701	.787	.783
Stopword 3-grams	.632	.515	.715	.417	.778	.776
Stopword 7-grams	.653	.527	.652	.482	.753	.750
Stylistic Similarity						
Function Word Frequencies	.453	.296	.715	.417	.727	.719
Sequential TTR	.400	.220	.715	.417	.667	.638
Sentence Ratio	.389	.268	.755	.625	.657	.653
Token Ratio	.432	.222	.755	.619	.778	.774
Type-Token Ratio	.379	.197	.715	.417	.723	.712

Table 1: Performance of selected similarity measures on the Wikipedia Rewrite Corpus, the METER Corpus, and the Webis Crowd Paraphrase Corpus, grouped by similarity dimension

Additionally, we compared the average sentence and token lengths between the reused text and the original source. We refer to these measures as *sentence ratio* and *token ratio*, respectively.

Finally, we compare texts by their *function word frequencies* (Dinu and Popescu, 2009) which have shown to be good style indicators in authorship attribution studies. Following the original work, this measure uses a set of 70 function words identified by Mosteller and Wallace (1964) and computes feature vectors of their frequencies for each possibly reused document and the source text. The comparison of the vectors is then performed using Pearson's correlation.

3 Experiments & Results

We utilized three datasets for the evaluation of our system which originate in the fields of plagiarism detection, journalistic text reuse detection, and paraphrase recognition: the **Wikipedia Rewrite Corpus** (Clough and Stevenson, 2011), the **METER Corpus** (Gaizauskas et al., 2001), and the **Webis Crowd Paraphrase Corpus** (Burrows et al., 2012), described below.

We carried out the same evaluation procedure for each of the three datasets: First, we computed text similarity scores between all pairs of possibly reused texts and their original sources using all the measures introduced in Section 2. We then used these scores as features for two machine learning classifiers in order to combine them across the three dimensions *content*, *structure*, and *style*. We experimented with two classifiers from the WEKA toolkit (Hall et al., 2009): a Naive Bayes classifier and a C4.5 decision tree classifier (J48 implementation).

In a 10-fold cross-validation setup, we ran three sets of experiments as follows: (i) First, we tested only the text similarity scores of one single measure at a time as single feature for the classifiers, in order to determine the individually best-performing measures per similarity

System	Acc.	$\bar{F}_1$
Majority Class Baseline	.400	.143
Ferret Baseline	.642	.517
Chong et al. (2010)[6]	*.705*	*.641*
Clough and Stevenson (2011)		
- our re-implementation[7]	.726	.658
- *as reported in their work*	*.800*	*.757*
Our Approach	**.842**	**.811**

exp. \ class.	cut&paste	light rev.	heavy rev.	no plag.
cut&paste	**15**	1	1	2
light rev.	3	**13**	3	0
heavy rev.	2	2	**15**	0
no plag.	0	0	1	**37**

Table 2: Results and confusion matrix (expected class vs. classification result) for the best classification on the Wikipedia Rewrite Corpus for the original 4-way classification

dimension. (ii) We then combined the measures per dimension by using multiple text similarity scores as feature set, in order to determine the performance of multiple measures within a single dimension. (iii) Finally, we combined the measures across dimensions to determine the best overall configuration. We compare our results with two baselines: the majority class baseline and the word trigram similarity measure *Ferret* (Lyon et al., 2004) (see Section 2.1). Additionally, we report the best results from the literature for comparison.

Evaluation was carried out in terms of accuracy and $\bar{F}_1$ score. By accuracy, we refer to the number of correctly predicted texts divided by the total number of texts. As the class distributions in both datasets are skewed, we report the overall $\bar{F}_1$ score as the arithmetic mean across the F_1 scores of all classes in order to account for the class imbalance.

3.1 Wikipedia Rewrite Corpus

Dataset The dataset contains 100 pairs of short texts (193 words on average). For each of 5 questions about topics of computer science (e.g. "What is dynamic programming?"), a reference answer (*source text*, henceforth) has been manually created by copying portions of text from a suitable Wikipedia article. Text reuse now occurs between a source text and an answer given by one of 19 participants. The participants were asked to provide short answers, each of which should comply to one of 4 rewrite levels and hence reuse the source text to a varying extent. According to the degree of rewrite, the dataset is 4-way classified as *cut & paste* (38 texts; simple copy of text portions from the Wikipedia article), *light revision* (19; synonym substitutions and changes of grammatical structure allowed), *heavy revision* (19; rephrasing of Wikipedia excerpts using different words and structure), and *no plagiarism* (19; answer written independently from the Wikipedia article). An example of a *heavy revision* was given in Figure 1.

Results We summarize the results on this dataset in Table 2.[8] In the best configuration, when combining similarity measures across dimensions, our system achieves a performance of

[6] Chong et al. (2010) report $\bar{F}_1 = .698$ in their original work. This figure, however, reflects the *weighted* arithmetic mean over all four classes of the dataset where one class is twice as prominent as each of the others. As dicussed in Section 3, we report all $\bar{F}_1$ scores as the *unweighted* arithmetic mean in order to account for the class imbalance.

[7] While we were able to reproduce the results of the *Ferret* baseline as reported by Chong et al. (2010), our re-implementation of the system by Clough and Stevenson (2011) (Naive Bayes classifier, same feature set) resulted in a much lower overall performance. We observed the largest difference for the *longest common subsequence* measure, even though we used a standard implementation (Allison and Dix, 1986) and normalized as described by Clough and Stevenson (2011).

[8] Figures in italics are taken from the literature, while we (re-)implemented the remaining systems. This applies to all result tables in this paper.

Text Similarity Dimension	Acc.	$\bar{F}_1$
Combinations within dimensions		
Content	.747	.693
Structure	.716	.660
Style	.442	.398
Combinations across dimensions		
Content + Style	.800	.757
Content + Structure	**.842**	**.811**
Structure + Style	.632	.569
Content + Structure + Style	.832	.798

Table 3: Results of the best combinations of text similarity measures within and across dimensions on the Wikipedia Rewrite Corpus

$\bar{F}_1 = .811$. It outperforms the best reference system by Clough and Stevenson (2011) by 5.4% points in terms of $\bar{F}_1$ score compared to their reported numbers, and by 15.3% points compared to our re-implementation of this system[7]. Their system uses a Naive Bayes classifier with only a very small feature set: *word n-gram containment* ($n = 1, 2, \ldots, 5$) and *longest common subsequence*. For comparison, we re-implemented their system and also applied it to the two datasets in the remainder of this paper. We report our findings in Sections 3.2 and 3.3.

In Table 1, we further report the detailed results for a selected set of individual text similarity measures, listed by similarity dimension.[9] Due to space limitations, we only report a selected set of best-performing measures per dimension and compare them with the baselines: While the majority class baseline performs very poor on this dataset ($\bar{F}_1 = .143$), the *Ferret* baseline achieves $\bar{F}_1 = .517$. Some content similarity measures such as *word 2-grams containment* show a reasonable performance ($\bar{F}_1 = .683$), while structural measures cannot exceed $\bar{F}_1 = .554$, and stylistic measures perform only slightly better than the majority class baseline ($\bar{F}_1 = .296$).

In Table 3, we report the best results for the combinations of text similarity measures within and across dimensions. When we combine the measures within their respective dimensions, *content* outperforms structural and stylistic similarity. However, all combinations of measures across dimensions in addition to content similarity improve the results. The best performance is achieved by combining the three similarity measures *longest common subsequence*, *stopword* 10-*grams*, and *character* 5-*gram profiles* from the two dimensions *content* and *structure*. This supports our hypothesis that the similarity computation process indeed profits from dimensions other than *content*. The effects of dimension combination held true regardless of the classifier used, even though the decision tree classifier performed consistently better than Naive Bayes.

Error Analysis We present the confusion matrix for our best configuration in Table 2. In total, 15 texts out of 95 have been classified with the wrong label. While all texts except a single one in the class *no plagiarism* have been classified correctly, 67% of errors (10 texts) are due to misclassifications in the *light* and *heavy revision* classes. We assume that these errors are due to questionable gold standard annotations as the annotation guidelines for these two classes are highly similar (Clough and Stevenson, 2011). For the *light revision* class, the annotators "could alter the text in some basic ways", thereby "altering the grammatical structure (i.e. paraphrasing)." Likewise, for the *heavy revision* class, the annotation manual expected the

[9]Table 1 also lists the detailed results for the METER Corpus and the Webis Crowd Paraphrase Corpus. We will discuss the numbers in the corresponding Sections 3.2 and 3.3.

System	Acc.	$\bar{F}_1$
Majority Class Baseline	.400	.190
Ferret Baseline	.768	.745
Clough and Stevenson (2011)[13]	.821	.788
Our Approach	**.884**	**.859**

exp. \ class.	cut&paste	potential	no plag.
cut&paste	**14**	3	2
potential	5	**33**	0
no plag.	0	1	**37**

Table 4: Results and confusion matrix on the Wikipedia Rewrite Corpus for the folded 3-way classification

System	Acc.	$\bar{F}_1$
Majority Class Baseline	.600	.375
Ferret Baseline	.937	.935
Clough and Stevenson (2011)		
- our re-implementation	.958	.957
- *as reported*	*.947*	*n/a*
Our Approach	**.968**	**.967**

exp. \ class.	plagiarism	no plag.
plagiarism	**55**	2
no plag.	1	**37**

Table 5: Results and confusion matrix on the Wikipedia Rewrite Corpus for the folded binary classification

annotators to "rephrase the text to generate an answer with the same meaning as the source text, but expressed using different words and structure."

As each text of this dataset was written by only a single person for a given rewrite category, we decided to conduct an annotation study, in which we were mostly interested in the inter-rater agreement of the subjects. We asked 3 participants to rate the degree of text reuse and provided them with the original annotation guidelines. We used a generalization of Scott's (1955) π-measure for calculating a chance-corrected inter-rater agreement for multiple raters, which is known as Fleiss' (1971) κ and Carletta's (1996) K.[10] In summary, the results[11] of our study support our hypothesis that the annotators mostly disagree for the *light* and *heavy revision* classes, with fair[12] agreements of $\kappa = .34$ and $\kappa = .28$, respectively. For the *cut & paste* and *no plagiarism* classes, we observe moderate[12] agreements, $\kappa = .53$ and $\kappa = .56$, respectively.

Based on these insights, we decided to fold the *light* and *heavy revision* classes into a single class *potential plagiarism*. This approach was also briefly discussed by Clough and Stevenson (2011), though not carried out in their work. We report the corresponding results and the confusion matrix in Table 4. As the classification task gets easier by the reduction to three classes, the results for the Ferret baseline improve, from $\bar{F}_1 = .517$ to $\bar{F}_1 = .745$. The re-implementation of the system by Clough and Stevenson (2011) achieves $\bar{F}_1 = .788$. Our system again outperforms all other systems with $\bar{F}_1 = .859$.

In our envisioned semi-supervised application scenario, potentially reused texts are presented to users in an informative manner. Here, fine-grained distinctions are not necessary, and we decided to go even one step further and fold all potential cases of text reuse. This variant of the dataset results in a binary classification of plagiarized/non-plagiarized texts. We present

[10] An exhaustive discussion of inter-rater agreement measures is given by Artstein and Poesio (2008).

[11] `http://www.ukp.tu-darmstadt.de/data/text-similarity/text-reuse-annotations`

[12] Strength of agreement for κ values according to Landis and Koch (1977)

[13] We report the results for our re-implementation of the system by Clough and Stevenson (2011). In their original work, they did not evaluate on this dataset.

System	Acc.	$\bar{F}_1$
Majority Class Baseline	.715	.417
Ferret Baseline	.684	.535
Clough and Stevenson (2011)[13]	.692	.680
Sánchez-Vega et al. (2010)	*.783*	*.705*
Our Approach	**.802**	**.768**

exp. \ class.	reuse	no reuse
reuse	**151**	30
no reuse	20	**52**

Table 6: Results and confusion matrix for the best classification on the METER Corpus

the results and the corresponding confusion matrix in Table 5. In this simplified setting, even the Ferret baseline achieves an excellent performance of $\bar{F}_1 = .935$. Our approach still slightly outperforms ($\bar{F}_1 = .967$) the re-implementation of the system by Clough and Stevenson (2011).

An interesting observation across all three variants of the dataset is that the same three texts always constitute severe error instances where e.g. a *cut & paste* text is falsely labeled as *no plagiarism*, which is more severe than mislabeling a *light revision* as a *heavy revision*. Two of the three cases account for the texts which describe the PageRank algorithm. One of these instances was falsely labeled as *cut & paste* while it is non-plagiarized, and the other one vice-versa. We attribute the misclassifications to the model built up in the classifier's training phase.

In the envisioned semi-supervised setting, the remaining less severe error instances, where e.g. a *light revision* was classified as a *heavy revision*, can be reviewed by a user of the system. We suppose it is even hard for users to draw a strict line between possibly reused and non-reused texts, as this heavily depends on external effects such as user intentions and the task at hand.

3.2 METER Corpus

Dataset The dataset contains news sources from the UK Press Association (PA) and newspaper articles from 9 British newspapers that reused the PA source texts to generate their own texts. The complete dataset contains 1,716 texts from two domains: law & court and show business. All newspaper articles have been annotated whether they are *wholly derived* from the PA sources (i.e. the PA text has been used exclusively as text reuse source), *partially derived* (the PA text has been used in addition to other sources), or *non-derived* (the PA text has not been used at all).

Several newspaper texts, though, have more than a single PA source in the original dataset where it is unclear which (if not all) of the source stories have been used to generate the rewritten story. However, for text reuse detection it is important to have aligned pairs of reused texts and source texts. Therefore, we followed Sánchez-Vega et al. (2010) and selected a subset of texts where only a single source story is present in the dataset. This leaves 253 pairs of short texts (205 words on average). We further followed Sánchez-Vega et al. (2010) and folded the annotations to a binary classification of 181 *reused* (wholly/partially derived) and 72 *non-reused* instances in order to carry out a comparable evaluation study.

Results We summarize the results on this dataset in Table 6. In the best configuration, our system achieves an overall performance of $\bar{F}_1 = .768$. It outperforms the best reference system by Sánchez-Vega et al. (2010) by 6.3% points in terms of $\bar{F}_1$ score. Their system uses a Naive Bayes classifier with two custom features which compare texts based on the length and frequency of common word sequences and the relevance of individual words. As in Section 3.1, we further report the detailed results for a selected set of individual text similarity measures

Text Similarity Dimension	Acc.	$\bar{F}_1$
Combinations within dimensions		
Content	.759	.712
Structure	.731	.701
Style	.755	.672
Combinations across dimensions		
Content + Style	.779	.733
Content + Structure	.739	.713
Structure + Style	.767	.739
Content + Structure + Style	**.802**	**.768**

Table 7: Results of the best combinations of text similarity measures within and across dimensions on the METER Corpus

in Table 1. From these figures, we learn that many text similarity measures cannot exceed the simple majority class baseline ($\bar{F}_1 = .417$) when applied individually.

In Table 7, we show that the performance of text reuse detection always improves over individual measures (cf. Table 1) when we combine the measures within their respective dimensions. An exception is the combination of structural similarity measures, which only performs on the same level as the best individual measure *part-of-speech* 3-*grams containment*. Combinations of content similarity measures show a better performance than combinations of structural or stylistic measures. Our system achieves its best performance on this dataset when text similarity measures are combined across all three dimensions *content*, *structure*, and *style*. The best configuration resulted from using a Naive Bayes classifier with the following measures: *Greedy String Tiling*, *stopword* 12-*grams*, and *Sequential TTR*. As for the previous dataset, the effects of dimension combination held true regardless of the classifier used.

The influence of the stylistic similarity measures is particularly interesting to note. In contrast to the Wikipedia Rewrite Corpus, including these measures in the composition improves the results on this dataset: Our classifier is able to detect similarity even for reused texts by expert journalists. This is due to the fact that a journalistic text which reuses the original press agency source most likely also shows stylistic similarity in terms of e.g. vocabulary richness.

Error Analysis We present the confusion matrix for our best configuration in Table 6. In total, 50 texts out of 253 have been classified incorrectly: 30 instances of text reuse have not been identified by the classifier, and 20 non-reused texts have been mistakenly labeled as such. However, the original annotations have been carried out by only a single annotator (Gaizauskas et al., 2001) which may have resulted in subjective judgments. Thus, as for the previous dataset in Section 3.1, we conducted an annotation study with three annotators to gain further insights into the data. The results[11] show that for 61% of all texts the annotators fully agree. The chance-corrected Fleiss' (1971) agreement $\kappa = .47$ is moderate[12].

For the 30 instances of text reuse which have not been identified by the classifier, it is particularly interesting to note that many errors are due to the fact that a lower overall text similarity between the possibly reused text and the original source does not necessarily entail the label *no reuse*. The newspaper article about the English singer-songwriter Liam Gallagher, for example, is originally labeled as text reuse. However, our classifier falsely assigned the label *no reuse*. It turns out, though, that the reused text is about four times as long as the original press agency source, with lots of new facts being introduced there. Consequently, only a low similarity score

System	Acc.	$\bar{F}_1$
Majority Class Baseline	.517	.341
Ferret Baseline	.794	.789
Clough and Stevenson (2011)[13]	.798	.795
Burrows et al. (2012)	*.839*	*.837*
Our Approach	**.853**	**.852**

exp. \ class.	paraphrase	no para.
paraphrase	**3,654**	413
no para.	759	**3,033**

Table 8: Results and confusion matrix for the best classification on the Webis Crowd Paraphrase Corpus

can be computed between the additional material in the newspaper article and the original source, and the overall similarity score decreases.

We conclude that applications will benefit from an improved classifier which better deals with theses instances. For example, similarity features could be computed per section, not per document, which would allow to also identify potential instances of text reuse for only partially matching texts. The currently achieved performance (see Table 6) of text reuse detection, though, is sufficient for our envisioned semi-supervised application scenario where content authors are provided only with suggestions of potential instances of text reuse and then are free to decide how to proceed, e.g. to merge both texts. The final decision probably also depends on external factors such as user intentions and the task at hand.

3.3 Webis Crowd Paraphrase Corpus

Dataset The dataset was originally introduced as part of the PAN 2010 international plagiarism detection competition (Potthast et al., 2010). It contains 7,859 pairs of original texts along with their paraphrases (28 to 954 words in length) with 4,067 (52%) positive and 3,792 (48%) negative samples. The original texts are book excerpts from Project Gutenberg[14], and the corresponding paraphrases were acquired in a crowdsourcing process using Amazon Mechanical Turk (Callison-Burch and Dredze, 2010). In the manual filtering process[15] of all acquired paraphrases, Burrows et al. (2012) hereby follow the paraphrase definition by Boonthum (2004), where a *good* paraphrase exhibits patterns such as synonym use, changes between active and passive voice, or changing word forms and parts of speech, and a *bad* paraphrase is rather e.g. a (near-)duplicate or an automated one-for-one word substitution. This definition implies that a more sophisticated interpretation of text similarity scores needs to be learned, where e.g. (near-)duplicates with very high similarity scores are in fact negative samples.

Results We summarize the results on this dataset in Table 8. Even though the Ferret baseline is a strong competitor ($\bar{F}_1 = .789$), our approach achieves the best results on this dataset with $\bar{F}_1 = .852$. The results reported by Burrows et al. (2012) are slightly worse ($\bar{F}_1 = .837$). Their best score was achieved by using a k-nearest neighbor classifier with a feature set of 10 similarity measures. They exclusively used similarity measures that operate on the texts' string sequences and thus capture the content dimension of text similarity only, e.g. *Levenshtein (1966)* distance and a *word n-gram* similarity measure. As in the previous sections, we report the detailed results for a selected set of individual text similarity measures in Table 1. These figures show that

[14] `www.gutenberg.org`

[15] Burrows et al. (2012) do not report any inter-annotator agreements for the filtering process, as the task was split across two annotators and each text pair was labeled by only a single annotator.

Text Similarity Dimension	Acc.	$\bar{F}_1$
Combinations within dimensions		
Content	.840	.839
Structure	.816	.814
Style	.819	.817
Combinations across dimensions		
Content + Style	.844	.843
Content + Structure	.838	.838
Structure + Style	.831	.830
Content + Structure + Style	**.853**	**.852**

Table 9: Results of the best combinations of text similarity measures within and across dimensions on the Webis Crowd Paraphrase Corpus

regardless of the similarity dimension many measures achieve a very reasonable performance when applied individually, with the measures *Greedy String Tiling* and *word 2-grams containment* performing best.

As for the previous datasets, our hypothesis holds true that the combination of similarity dimensions improves the results: When we combine the similarity features within each of the respective dimensions, the performance numbers increase (see Table 9 as compared to Table 1). The combination of content similarity measures is stronger than the combination of structural and stylistic similarity measures, and performs on the same level as the original results reported by Burrows et al. (2012). This is to be expected, as their system uses a feature set which also addresses the content dimension exclusively.

When we combine measures across dimensions, the results improve even further. An exception is the combination of content and structural measures, which performs slightly worse than content measures alone due to the lower performance of structural measures on this dataset. The best configuration of our system resulted from combining all three dimensions *content*, *structure*, and *style* in a single classification model using the decision tree classifier, resulting in $\bar{F}_1 = .852$. The final feature set contains 16 text similarity features which are listed in Table 10.

Error Analysis We present the confusion matrix for our best classification in Table 8. In total, 1,172 (15%) out of 7,859 text pairs have been classified incorrectly. Out of these, our classifier mistakenly labeled 759 instances of negative samples as true paraphrases, while 413 cases of true paraphrases were not recognized. However, in our opinion the 759 false positives are less severe errors in our envisioned semi-supervised application setting, as user intentions and the current task at hand may highly influence a user's decision to consider texts as reused or not.

In general, we attribute the errors to the particular properties of this dataset, which differ from those of the Wikipedia Rewrite Corpus and the METER Corpus (see Sections 3.1 and 3.2). For those two datasets, the more similar two texts are, the higher their degree of text reuse. For the Webis Crowd Paraphrase Corpus, however, a different interpretation needs to be learned by the classifier: Here, (near-)duplicates and texts with automated word-by-word substitutions, which will receive high similarity scores by any of our content similarity measures, are in fact annotated as *bad* paraphrases, i.e. negative samples. Unrelated texts, empty samples, or texts alike also belong to the class of negative samples. In consequence, positive samples are only those in the medium similarity range. We assume that the more elaborate definition of positive and negative cases makes it more difficult to learn a proper model for the given data.

Content	ESA (WordNet, with + w/o stopwords), Greedy String Tiling, Jaro, Longest Common Substring, Longest Common Subseq. (2 norm.), n-gram Jaccard ($n = \{6, 14, 15\}$), Resnik (SMT wrapper)
Structure	Lemma Pair Ordering, POS 2-grams Jaccard, Stopword 6-grams
Style	Function Word Frequencies, Sequential TTR, Token Ratio

Table 10: Feature set used to achieve the best results on the Webis Crowd Paraphrase Corpus

4 Conclusions and Future Work

The motivation for this work stemmed from the hypothesis that *content* features alone are not a reliable indicator for text reuse detection. As illustrated in Figure 1, a reused text may also contain modifications such as split sentences, changed order of reused parts, or stylistic variance. We thus devised an architecture which composes diverse text similarity measures in a supervised classification model. In this model, we overcome the traditional limitation of text similarity measures to *content* features and compute similarity along three characteristic dimensions inherent to texts: *content*, *structure*, and *style*.

We evaluated our classification model on three standard datasets where text reuse is prevalent and which originate in the fields of plagiarism detection, journalistic text reuse detection, and paraphrase recognition: the *Wikipedia Rewrite Corpus* (Clough and Stevenson, 2011), the *METER Corpus* (Gaizauskas et al., 2001), and the *Webis Crowd Paraphrase Corpus* (Burrows et al., 2012). Based on the evaluation results, we discussed the influence of each of the similarity dimensions, and demonstrated empirically that text reuse can be best detected if measures are combined across dimensions, so that a wide variety of text features are taken into consideration. The composition consistently outperforms previous approaches across all datasets.

As we showed, similarity computation works best if the similarity dimensions are chosen well with respect to the type of text reuse at hand. For the Wikipedia Rewrite Corpus, for example, the stylistic similarity features perform only poorly, which is why the composition of all three dimensions performs slightly worse than than the combination of only content and structural features. For the other two datasets, however, stylistic similarity is a strong dimension within the composition, and consequently the best performance is reached when combining all three dimensions. Based on these insights, we conclude that for novel datasets it is essential to address the dimensions explicitly in the annotation process, so that text reuse detection approaches can be evaluated precisely against particular characteristics of different kinds of data.

For future work, we expect that considering a dimensional representation of text similarity features will also benefit any other task where text similarity computation is fundamental and which is yet limited to *content* features, e.g. paraphrase recognition or automatic essay grading. For the latter, we see great potential for improvements by including, for example, measures for grammar analysis, lexical complexity, or measures assessing text organization with respect to the discourse elements. However, each task exhibits particular characteristics which influence the choice of a suitable set of similarity dimensions. As discussed above, a particular dimension may or may not contribute to an overall improvement based on the nature of the data.

Acknowledgements

This work has been supported by the Volkswagen Foundation as part of the Lichtenberg-Professorship Program under grant No. I/82806, and by the Klaus Tschira Foundation under project No. 00.133.2008. We thank Chris Biemann for his inspirations, as well as Carolin Deeg, Andriy Nadolskyy, and Artem Vovk for their participation in the annotation studies.

References

Abdel-Hamid, O., Behzadi, B., Christoph, S., and Henzinger, M. (2009). Detecting the Origin of Text Segments Efficiently. In *Proceedings of the 18th International Conference on World Wide Web*, pages 61–70, Madrid, Spain.

Allison, L. and Dix, T. I. (1986). A bit-string longest-common-subsequence algorithm. *Information Processing Letters*, 23:305–310.

Artstein, R. and Poesio, M. (2008). Inter-Coder Agreement for Computational Linguistics. *Computational Linguistics*, 34(4):555–596.

Bär, D., Zesch, T., and Gurevych, I. (2011). A Reflective View on Text Similarity. In *Proceedings of the International Conference on Recent Advances in Natural Language Processing*, pages 515–520, Hissar, Bulgaria.

Barrón-Cedeño, A., Rosso, P., Agirre, E., and Labaka, G. (2010). Plagiarism Detection across Distant Language Pairs. In *Proceedings of the 23rd International Conference on Computational Linguistics*, pages 37–45, Beijing, China.

Boonthum, C. (2004). iSTART: Paraphrase Recognition. In *Proceedings of the 42nd Meeting of the Association for Computational Linguistics: Student Research Workshop*, pages 31–36, Barcelona, Spain.

Broder, A. Z. (1997). On the resemblance and containment of documents. *Proceedings of Compression and Complexity of Sequences*, pages 21–29.

Broder, A. Z., Glassman, S. C., Manasse, M. S., and Zweig, G. (1997). Syntactic clustering of the Web. In *Proceedings of the 6th International World Wide Web Conference*, pages 1157–1166, Santa Clara, CA, USA.

Burrows, S., Potthast, M., and Stein, B. (2012). Paraphrase Acquisition via Crowdsourcing and Machine Learning. *Transactions on Intelligent Systems and Technology*, V(January):1–22.

Callison-Burch, C. and Dredze, M. (2010). Creating Speech and Language Data With Amazon's Mechanical Turk. In *Proceedings of the NAACL HLT Workshop on Creating Speech and Language Data with Amazon's Mechanical Turk*, pages 1–12, Los Angeles, CA, USA.

Carletta, J. (1996). Assessing Agreement on Classification Tasks: The Kappa Statistic. *Computational Linguistics*, 22(2):249–254.

Charikar, M. S. (2002). Similarity Estimation Techniques from Rounding Algorithms. In *Proceedings of the 34th Annual Symposium on Theory of Computing*, pages 380–388, Montreal, Canada.

Chong, M., Specia, L., and Mitkov, R. (2010). Using Natural Language Processing for Automatic Detection of Plagiarism. In *Proceedings of the 4th International Plagiarism Conference*, Newcastle upon Tyne, UK.

Clough, P., Gaizauskas, R., Piao, S. S., and Wilks, Y. (2002). METER: MEasuring TExt Reuse. In *Proceedings of 40th Annual Meeting of the Association for Computational Linguistics*, pages 152–159, Philadelphia, PA, USA.

Clough, P. and Stevenson, M. (2011). Developing a Corpus of Plagiarised Short Answers. *Language Resources and Evaluation: Special Issue on Plagiarism and Authorship Analysis*, 45(1):5–24.

Dinu, L. P. and Popescu, M. (2009). Ordinal measures in authorship identification. In *Proceedings of the 3rd PAN Workshop. Uncovering Plagiarism, Authorship and Social Software Misuse*, pages 62–66, San Sebastian, Spain.

Fellbaum, C. (1998). *WordNet: An Electronic Lexical Database*. MIT Press.

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

Gabrilovich, E. and Markovitch, S. (2007). Computing Semantic Relatedness using Wikipedia-based Explicit Semantic Analysis. In *Proceedings of the 20th International Joint Conference on Artificial Intelligence*, pages 1606–1611, Hyderabad, India.

Gaizauskas, R., Foster, J., Wilks, Y., Arundel, J., Clough, P., and Piao, S. (2001). The METER Corpus: A corpus for analysing journalistic text reuse. In *Proceedings of the Corpus Linguistics 2001 Conference*, pages 214–223.

Gärdenfors, P. (2000). *Conceptual Spaces: The Geometry of Thought*. MIT Press.

Goodman, N. (1972). Seven strictures on similarity. In Goodman, N., editor, *Problems and projects*, pages 437–446. Bobbs-Merrill.

Gusfield, D. (1997). *Algorithms on Strings, Trees and Sequences: Computer Science and Computational Biology*. Cambridge University Press.

Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., and Witten, I. H. (2009). The WEKA Data Mining Software: An Update. *SIGKDD Explorations*, 11(1):10–18.

Hatzivassiloglou, V., Klavans, J. L., and Eskin, E. (1999). Detecting text similarity over short passages: Exploring linguistic feature combinations via machine learning. In *Proceedings of the Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora*, pages 203–212, College Park, MD, USA.

Henzinger, M. (2006). Finding Near-Duplicate Web Pages: A Large-Scale Evaluation of Algorithms. In *Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 284–291, Seattle, WA, USA.

Hoad, T. C. and Zobel, J. (2003). Methods for identifying versioned and plagiarized documents. *Journal of the American Society of Information Science and Technology*, 54(3):203–215.

Jaro, M. A. (1989). Advances in record linkage methodology as applied to the 1985 census of Tampa Florida. *Journal of the American Statistical Association*, 84(406):414–420.

Jiang, J. J. and Conrath, D. W. (1997). Semantic similarity based on corpus statistics and lexical taxonomy. In *Proceedings of the 10th International Conference on Research in Computational Linguistics*.

Keselj, V., Peng, F., Cercone, N., and Thomas, C. (2003). N-gram-based author profiles for authorship attribution. In *Proceedings of the Conference of the Pacific Association for Computational Linguistics*, pages 255–264.

Koehn, P. (2005). Europarl: A Parallel Corpus for Statistical Machine Translation. In *Proceedings of the 10th Machine Translation Summit*, pages 79–86, Phuket Island, Thailand.

Koehn, P., Hoang, H., Birch, A., Callison-Burch, C., Federico, M., Bertoldi, N., Cowan, B., Shen, W., Moran, C., Zens, R., Dyer, C., Bojar, O., Constantin, A., and Herbst, E. (2007). Moses: Open Source Toolkit for Statistical Machine Translation. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics Companion Volume Proceedings of the Demo and Poster Sessions*, pages 177–180, Prague, Czech Republic.

Landauer, T. K., Foltz, P. W., and Laham, D. (1998). An introduction to latent semantic analysis. *Discourse Processes*, 25(2):259–284.

Landis, J. R. and Koch, G. G. (1977). The measurement of observer agreement for categorical data. *Biometrics*, 33(1):159–174.

Lee, J. (2007). A Computational Model of Text Reuse in Ancient Literary Texts. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 472–479.

Leuf, B. and Cunningham, W. (2001). *The Wiki Way: Collaboration and Sharing on the Internet*. Addison-Wesley Professional.

Levenshtein, V. I. (1966). Binary codes capable of correcting deletions, insertions, and reversals. *Soviet Physics Doklady*, 10(8):707–710.

Lin, D. (1998). An information-theoretic definition of similarity. In *Proceedings of International Conference on Machine Learning*, pages 296–304.

Lyon, C., Barrett, R., and Malcolm, J. (2004). A theoretical basis to the automated detection of copying between texts, and its practical implementation in the Ferret plagiarism and collusion detector. In *In Plagiarism: Prevention, Practice and Policies Conference*.

Lyon, C., Malcolm, J., and Dickerson, B. (2001). Detecting short passages of similar text in large document collections. In *Proceedings of Conference on Empirical Methods in Natural Language Processing*, pages 118–125.

Manku, G. S., Jain, A., and Sarma, A. D. (2007). Detecting Near-Duplicates for Web Crawling. In *Proceedings of the 16th International World Wide Web Conference*, pages 141–149, Banff, AB, Canada.

McCarthy, P. M. and Jarvis, S. (2010). MTLD, vocd-D, and HD-D: A validation study of sophisticated approaches to lexical diversity assessment. *Behavior research methods*, 42(2):381–392.

Mihalcea, R., Corley, C., and Strapparava, C. (2006). Corpus-based and Knowledge-based Measures of Text Semantic Similarity. In *Proceedings of the 21st National Conference on Artificial Intelligence*, pages 775–780, Boston, MA, USA.

Monge, A. and Elkan, C. (1997). An efficient domain-independent algorithm for detecting approximately duplicate database records. In *Proceedings of the SIGMOD Workshop on Data Mining and Knowledge Discovery*, pages 23–29, Tucson, AZ, USA.

Mosteller, F. and Wallace, D. L. (1964). *Inference and disputed authorship: The Federalist*. Addison-Wesley.

Potthast, M., Barrón-Cedeño, A., Eiselt, A., Stein, B., and Rosso, P. (2010). Overview of the 2nd International Competition on Plagiarism Detection. In *Notebook Papers of CLEF 10 Labs and Workshops*, Padua, Italy.

Resnik, P. (1995). Using Information Content to Evaluate Semantic Similarity in a Taxonomy. In *Proceedings of the 14th International Joint Conference on Artificial Intelligence*, pages 448–453, Montreal, Canada.

Salton, G. and McGill, M. J. (1983). *Introduction to Modern Information Retrieval*. McGraw-Hill.

Sánchez-Vega, F., Villaseñor-Pineda, L., Montes-y-Gómez, M., and Rosso, P. (2010). Towards Document Plagiarism Detection Based on the Relevance and Fragmentation of the Reused Text. In *Proceedings of the 9th Mexican International Conference on Artificial Intelligence*, pages 24–31, Pachuca, Mexico.

Scott, W. A. (1955). Reliability of content analysis: The case of nominal scale coding. *Public Opinion Quarterly*, 19(3):321–325.

Shannon, C. E. (1951). Prediction and Entropy of Printed English. *Bell System Technical Journal*, 30:50–64.

Spärck Jones, K. (1972). A statistical interpretation of term specificity and its application in retrieval. *Journal of Documentation*, 28(1):11–21.

Stamatatos, E. (2011). Plagiarism detection using stopword n-grams. *Journal of the American Society for Information Science and Technology*, 62(12):2512–2527.

Templin, M. C. (1957). *Certain language skills in children*. University of Minnesota Press.

Tversky, A. (1977). Features of Similarity. In *Psychological Review*, volume 84, pages 327–352.

Winkler, W. E. (1990). String Comparator Metrics and Enhanced Decision Rules in the Fellegi-Sunter Model of Record Linkage. In *Proceedings of the Section on Survey Research Methods*, pages 354–359.

Wise, M. J. (1996). YAP3: Improved detection of similarities in computer program and other texts. In *Proceedings of the 27th SIGCSE technical symposium on Computer science education*, pages 130–134, Philadelphia, PA, USA.

Yule, G. U. (1939). On sentence-length as a statistical characteristic of style in prose: With application to two cases of disputed authorship. *Biometrika*, 30(3/4):363–390.

Deriving Paraphrases for Highly Inflected Languages from Comparable Documents

Kfir BAR Nachum DERSHOWITZ
School of Computer Science, Tel Aviv University, Ramat Aviv, Israel
kfirbar@post.tau.ac.il, nachumd@tau.ac.il

ABSTRACT

We describe an automatic paraphrase-inference procedure for a highly inflected language like Arabic. Paraphrases are derived from comparable documents, that is, distinct documents dealing with the same topic. A co-training approach is taken, with two classifiers, one designed to model the contexts surrounding occurrences of paraphrases, and the other trained to identify significant features of the words within paraphrases. In particular, we use morpho-syntactic features calculated for both classifiers, as is to be expected when working with highly inflected languages. We provide some experimental results for Arabic, and for the simpler English, which we find to be encouraging. Our immediate interest is to incorporate such paraphrases within an Arabic-to-English translation system.

KEYWORDS : Paraphrases, highly inflected languages, morphologically rich languages, co-training, comparable documents, Arabic

Proceedings of COLING 2012: Technical Papers, pages 185–200,
COLING 2012, Mumbai, December 2012.

1 Introduction

Paraphrases are pairs of sequences of words, both in the same language, that have the same meaning in at least some contexts. Given a text, "paraphrasing" is the act of generating an alternate sequence of words that conveys the same meaning. Since the meaning of a text is determined only when its context is given, paraphrases are sometimes referred to as "dynamic translations" or "semantic equivalents". Identifying paraphrases is an important capability for many natural language processing applications, including machine translation, as a possible workaround for the problem of limited coverage inherent in a corpus-based translation approach (Callison-Burch et al., 2006; Marton et el., 2009). Other applications of paraphrasing include automatic evaluation of summaries (Zhao et al., 2008) and question answering (Duboue and Chu-Carroll, 2006; Riezler et al., 2007).

There are two main directions of work on paraphrases that one can find in this field: investigating an automatic approach for uncovering paraphrases in a given corpus and using paraphrases to improve the performance of a specific task. In this paper, we introduce a novel method for extracting Arabic paraphrases from a corpus of comparable documents as part of our work on improving Arabic-to-English machine translation. *Comparable* documents are ones that deal with the same topic, such as two newspaper reports of the same event. The extraction technique is based on a learning method, known as "co-training" (Blum and Mitchel, 1998) and inspired by the work of Barzilay and McKeown (2001) for finding paraphrases in a parallel monolingual English corpus. In order to validate our technique, we have applied it also to a similar English corpus.

Like many other Semitic languages, Arabic is highly inflected; therefore, data sparseness becomes even more noticeable than in English and extracting paraphrases from a corpus turns out to be even more complicated. Arabic words are derived from a root and a pattern (template), combined with prefixes, suffixes and circumfixes. Using the same root with different patterns may yield words with different meanings. Words are inflected for person, number and gender; prefixes and suffixes are then added to indicate definiteness, conjunction, various prepositions and possessive forms. We will list some of the morpho-syntactic features we use for identifying paraphrases in the corpus. Based on the definition, paraphrases are identified as part of the context they are mentioned in within the corpus. Paraphrase is in fact only one of the semantic relations that can be identified to hold between two word sequences in their contexts; it can be seen as a special case of textual entailment (Dagan and Glickman, 2004), where each sequence entails the other.

There are several existing approaches for inferring paraphrases from a corpus, which differ from one another in the type of corpus they employ. Some require bilingual parallel corpora (Callison-Burch et al., 2006; Zhao et el., 2008), some need monolingual parallel corpora (Barzilai and McKeown, 2001), some need general monolingual corpora (Marton et al., 2009) and others need corpora of comparable documents (Rui and Callison-Burch, 2011; Dolan et al., 2004). Bilingual parallel corpora, pairing Arabic with languages other than English, are very hard to obtain. In this paper we take the last approach, leaving the other directions for future investigation.

Section 2 cites some related work. Our proposal is described in Section 3 with some experimental results reported in Section 4. Conclusions are given in the last section.

2 Related work

To the best of our knowledge, ours is the first work on paraphrasing in Arabic. In our previous work (Bar and Dershowitz, 2010), we extracted Arabic synonyms, that is, single-word paraphrases, using the English glosses provided by SAMA (Maamouri, 2010), along with WordNet for English (Fellbaum, 1998). The inferred synonyms were used to improve a corpus-based translation system. Salloum and Habash (2011) developed a rule-based algorithm for generating Modern Standard Arabic (MSA) paraphrases for dialectical Arabic phrases given to a statistics-based automatic translation system. They focused only on input phrases that do not exist in the translation table used by the translation system, for the purpose of improving its coverage. The MSA paraphrases were generated mostly using different morphological variations of the input words. They reported a slight improvement in BLEU score (Papineni, 2002) over a baseline system that does not use their generated paraphrases. In another work, by Denkowski et al. (2010), 726 Arabic paraphrases were manually generated and confirmed using Amazon's Mechanical Turk, from the NIST OpenMT 2002 development set (Garofolo, 2002). That was mainly done with the purpose of improving the evaluation of an English-to-Arabic machine translation system.

There are also related works in other languages. We only mention a few. Marton et al. (2009) found paraphrases to improve Spanish-to-English and English-to-Chinese statistical machine translation (SMT). For each phrase (as defined in SMT) that was left without a translation, they looked for it in a monolingual corpus and recorded the contexts in which it appeared. They modeled the contexts using a vector that captured phrase occurrences with their context words, and searched for other phrases with the most similar vector of occurrences to improve the translation. Callison-Burch et al. (2006) measured the effect of using paraphrases on Spanish-to-English and French-to-English SMT. They reported a significant improvement in coverage and in the final translation. The paraphrases were automatically extracted following the technique developed by Bannard and Callison-Burch (2005), using several parallel corpora of French and Spanish paired with other languages. This method is usually referred to as "pivoting". Both of these works claimed improvements in the translations. Callison-Burch (2008) and Zhao et al. (2008) developed this approach further by adding syntactic constraints to the extraction algorithms. In a recent work by Wang and Callison-Burch (2011), English paraphrases were found in a corpus of comparable documents. Similar to what we have done, they started with a large English corpus to find comparable documents. Those documents were used to find comparable sentences from which they extracted sub-sentential comparable fragments, that is, paraphrases. They used a chunker for finding linguistically safe boundaries for the fragments they extracted, and matched fragments based on the *n*-gram alignment method.

The most inspiring work for us is the one by Barzilay and McKeown (2001), in which paraphrases are extracted from a corpus containing multiple English translations of the same source. Using this type of corpus allow them to mark initial aligned anchors, chosen based on the results of an alignment algorithm, and to train a classifier to identify the best context environments surrounding potential paraphrases. Based on the resulting contexts, another classifier was trained for finding new paraphrases. This "co-training" process was repeated until no new paraphrases were extracted. In our work, we follow the same idea, implemented on Arabic. Since there is no monolingual parallel corpus available for Arabic, we created a corpus of comparable documents and used it as a resource for paraphrasing. Considering that Arabic is a

morphologically rich language, we incorporated morphological features of the surrounding words as well as the paraphrase patterns themselves.

3 Inferring paraphrases

3.1 Preparing the corpus

As just mentioned, our approach to inferring paraphrases is based on the work of Barzilay and McKeown (2001) on finding paraphrases in different English translations of the same source text. Understanding how powerful such a resource can be for paraphrasing, but finding no such resource for Arabic, we built a corpus of comparable documents, that is, distinct documents dealing with the same topic or event. This corpus was extracted from the Arabic Gigaword 4.0 (Parker, 2009), which contains newswire documents published by several news agencies, grouped by their publication date. Pairing documents, based on their topic, was done automatically using cosine similarity over the lemma-frequency vector of every document, with the lemma of every word extracted using MADA (Habash and Rambow, 2005; Roth et al., 2008). We considered candidates for document pairs only when they were published by different news agencies on the same day. For every document published by one agency, we pair it with a document from the agency that maximizes the similarity score over all the other documents published by the same agency on the same day. Not only that, we require that the score be higher than a predefined threshold that was set, in our experiment settings, to make sure that every candidate pair is composed of two documents sharing at least one third of the largest one. We also tried using lower thresholds for which we retrieved additional pairs; however, precision decreased linearly. It is obvious, then, that this approach prefers precision to recall; in other words, we probably miss a large number of potential candidates, while the candidates that we do extract are likely correct.

All together, we created 690 document pairs, comprising about half a million words. Our corpus of comparable documents was manually evaluated by two Arabic speakers. We randomly selected 120 document pairs out of the 690 and, for each, asked the evaluators for a simple "yes" or "no" answer to the question, "Do both documents discuss the same event?" The results are encouraging: out of the 120 pairs, 100 were classified as correct by both evaluators. Of the other 20 instances, 5 were classified "yes" by one evaluator. The rest of the pairs actually dealt with the same general domain but were not specifically discussing the same event. This positive evaluation allowed us to use this corpus in the next step of our inference technique.

Every document was pre-processed with AMIRAN before being given to the inference classifier, described in the next section. AMIRAN, an updated version of the AMIRA tools (Diab et al., 2004, 2007), is a tool for finding the context-sensitive morpho-syntactic information. AMIRAN combines AMIRA output with morphological analyses provided by SAMA. AMIRAN is also enriched with Named-Entity-Recognition (NER) class tags provided by (Benajiba et al., 2008). For every word, AMIRAN is capable of identifying the clitics, diacritized lemma, stem, full part-of-speech tag (excluding case and mood), base-phrase chunks and NER tags. The corpus is obviously not annotated with paraphrasing-related information and there is no alignment indication included at any level.

3.2 Inference technique

To infer new paraphrases from the corpus, we follow the "co-training" technique, training two different classifiers: one for modeling the context of a potential paraphrase and another for modeling the features of the paraphrase pattern itself. The main idea of the co-training approach applied to unlabeled data is to use the two classifiers on different views of the data. In our case, the two views are the context (CX) and the pattern (PT), with one classifier labeling the most reliable unlabeled data items for training the second classifier. Then, the second classifier can label some of the data items for training the first one. This process is repeated several times, and the labeled data collected during the entire run is returned. The algorithm runs in iterations; each iteration increases the number of words a potential paraphrase may contain, that is, in the first iteration only single-word paraphrases are allowed to be found, in the second one, paraphrases composed of up to two words are allowed, and so on. The input of the algorithm is the pairs of documents that we found on the previous section, from which we extract pairs of word sequences. A *pair of word sequences* is composed of two sequences, one from each of a pair of comparable documents. Since alignment at any level does not exist for comparable documents, we consider all the possible pairs of word sequences, given a pair of documents. To avoid too much noise, we restrict a word sequence for consideration to be composed of at least one non-function word and it to not break a base-phrase in the middle, similar to (Wang and Callison-Burch, 2011). Function words, in our case, are identified based on their part-of-speech and base-phrase tags, as provided by AMIRAN. Otherwise, a huge number of pairs containing only function words, not too important for paraphrasing, would be considered. The number of iterations, and concomitantly, the maximum length of the output sequences, is a parameter we control. As implied before, we start with single-word sequences and increase this parameter with every iteration. During the entire run of the algorithm, we maintain two sets of pairs of word sequences:

1. Labeled – containing pairs of word sequences with their label, "true" to indicate paraphrases and "false" to indicate that the word sequences are not paraphrases of each other. This set starts off empty.
2. Unlabeled – containing pairs of word sequences that are still waiting for their label assignment by the algorithm.

In every iteration, the algorithm performs the following steps:

1. deterministic labeling of potential paraphrases;
2. training the CX classifier using the labeled set as training data;
3. running CX on unlabeled pairs and labeling the most reliable ones;
4. training the PT classifier using the labeled set as training data;
5. running the PT classifier on the labeled set;
6. labeling some unlabeled pairs, based on the labels provided by both classifiers.

We now describe these steps in greater detail.

We cannot estimate in advance the weight of the selected features and their effect on the predictions of the classifiers; therefore, we chose to use support vector machine (SVM) classifiers (Vapnik and Cortes, 1995) because of their good generalization property. Technically, the classifiers are trained on the WEKA platform (Hall et al., 2009) running with the LibSVM library (Chang and Lin, 2011). One drawback of using SVM in this kind of setting is the long running

time of the training algorithm. Because we are running the trainer twice during every iteration, this drawback becomes even more pronounced.

The labeled pairs are used as training data for both classifiers, with every pair formatted as a feature vector. The features for the CX classifier capture some morpho-syntactic information expressed by the window-based context words. In the current experiment, we use a window of size three, that is, three words before each word sequence from the pair, and three words afterward. That gives us twelve words from which we extract features for representing a single pair and that number does not change during the entire learning process. Table 1 shows an example for a context.[1] The two main columns represent two Arabic sentences with their corresponding English translations, for easy reading. The emphasized texts are the actual paraphrases while the surrounding words are composing the context, which is described in the last row. In this case, the paraphrase pair is composed of a single identical lemma, inflected differently for person. The context of a paraphrase pair is composed of four parts: left and right words of each of the paired texts.

	Sentence 1	**Sentence 2**
Sentence	مكتب السنيورة وديوان ألمرط ينفيان خبرا عن لقاء في شرم الشيخ	مكتب السنيورة ينفي خبرا عن لقائه مسؤولين إسرائيليين
Transliteration	*mktb Alsnywrp wdywAn >wlmrt* ***ynfyAn*** *xbrA En lqA' fy $rm Al$yx*	*mktb Alsnywrp* ***ynfy*** *xbrA En lqA}h ms&wlyn <srA}ylyyn*
Translation	Seniora's office and the Olmert administration **deny** a story about a meeting in Sharm al-Sheikh	Seniora's office **denies** a story about his meeting with Israeli officials
Context	*Alsnywrp wdywAn >wlmrt* […] *xbrA En lqA'*	*mktb Alsnywrp* […] *xbrA En lqA}h*

TABLE 1 – An example for a context. The word sequence (here of size one) is highlighted in boldface.

The PT classifier makes its predictions based on the word sequences themselves; their number varies as the iteration number increases. For both classifiers, we use a quadratic kernel for capturing the common effect of all the features on prediction. Tables 2 and 3 summarize the features we currently use for building the feature vectors for the CX and PT classifiers, respectively. NER tags are assigned to persons, organizations, geo-political organizations and locations. The gloss-match rate is calculated for both sides of the context. In the example of Table 1, there is no word that matches on the left side (note that proper nouns usually do not have glosses). However, on the right side خبرا عن لقاء *xbrA En lqA'* ("a story about a meeting") matches خبرا عن لقائه *xbrA En lqA}h* ("a story about his meeting") with all three words on the gloss level; therefore, the left gloss-match rate is 0 and the right one is 1. The same calculation works with lemma-match rates on the lemma level. The morphological features we currently use in the PT classifier capture some common Arabic morphological variations. They are all Boolean values indicating whether the word expresses the feature or not. For example, the word وبكتابه *wbktAbh* ("and in his book") expresses conjunction, preposition and possessive. When working with Arabic, a highly inflected language, morphological features may contribute to the classification

[1] We use the Buckwalter transliteration scheme (Buckwalter, 2002) for rendering Arabic script in Romanization throughout this paper.

performance. We intend to further explore this direction in the future. The *n*-gram score is a simple language model score for capturing the co-occurrence of the candidate sequence words.

Feature	Description
Lemma, POS, NER, BP	of each context word
Gloss-match rate	The rate of gloss match on each side of the context (left and right)
Lemma-match rate	The rate of lemma match on each side of the context

TABLE 2 – The features we use for training the CX classifier on Arabic.

Feature	Description
n-gram score	Normalized *n*-gram frequency score for word sequences up to 4 words (2-4 grams)
POS, NER, BP	of each sequence word
Boolean morphological features (exists / does not exist): Conjunction, Possessive, Determiner and Prepositions	of each sequence word
Sequence length	The number of words in each sequence

TABLE 3 – The features we use for training the PT classifier on Arabic.

The first time one of the classifiers is trained, it needs some labeled items. With "co-training", those items are usually provided by manual annotation of a relatively small fraction of the data or, in this case, by using an automatic deterministic annotation algorithm. Therefore, in the first step of every iteration, the algorithm enriches the labeled set with additional "true" labeled pairs following a deterministic approach. Since it is very difficult to obtain a word or sentence-level alignment of two given comparable documents, our algorithm simply adds all the pairs whose word sequences match on the lemma level, word by word. If the lemma does not exist, we use the word's surface form for matching. Lengths of word sequences are determined by the iteration number, so in the first iteration only sequences of size 1 are added, in the second iteration sequences of size 2 are added, and so forth. Such a pair, matched on the lemma level, is shown in Table 1. Note that paraphrases work on the sense level, rather than on the surface form; however, our assumption is that, because we are using sequences from comparable documents, their senses may be the same with a reasonable high probability. Note that, since we are using the context-sensitive lemmas for matching, one can think of that as matching words on the sense level. However, AMIRAN was trained mostly with morpho-syntactic features and therefore achieves good performance in identifying the common lemma of a context-sensitive part-of-speech tag for every word. When a word may have two or more different lemmas for the same part-of-speech tag that have different senses, AMIRAN does not perform as well. For example, the word أمانة

>mAnp has three different noun lemmas: *>amAnap_1* (“faithfulness”), *>amAnap_2* (“secretariat”) and *>amAnap_3* (“deposit”).

That approach leaves us with some deterministically selected “true” examples; however, it does not provide us with the necessary “false” examples. In the first iteration, we consider word sequences of size 1 only. Our assumption is that word pairs sharing some of their senses in common may be considered paraphrases, thus cannot be naturally selected as “false” examples. Currently we use the English gloss of every word, as provided by AMIRAN, to select word pairs with different gloss values as “false” examples. Therefore, under this condition, the Arabic word pair مجال *mjAl* and منطقة *mnTqp* is not considered as a “false” example because they share the same gloss value: “area”. An alternative approach, which we plan to employ in the future, would be using Arabic WordNet (Black et al., 2006). It implies that, in our first iteration, only word pairs that have the same English gloss and not the same Arabic lemma are put in the unlabeled set. That dramatically reduces the amount of paraphrases of size one, better known as “synonyms”, that we can find. Since we are more interested in longer paraphrases, we can live with this limitation.

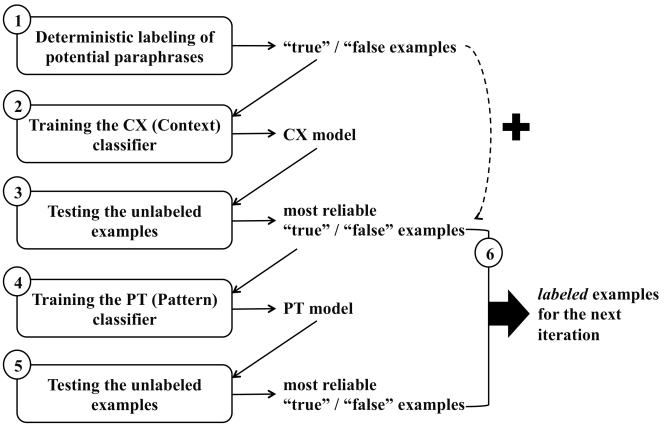


FIGURE 1 – An overview of the paraphrase inference co-training algorithm.

In subsequent iterations, “false” examples will be assigned automatically by the classifiers of the previous one, in the following way: after training the CX classifier in step 2, we use the classifier to tag the unlabeled pairs in step 3. Some pairs are assigned with the “true” label and some with “false”. Those for which the classifier has a “good sense” are added to the labeled set with their corresponding label. “Good sense” is measured with a confidence score that is provided by LibSVM along with every tested pair. Since this score is based on margin length calculations, one should use them carefully. Currently, we only set some threshold values for adding pairs to the labeled set, with a high score, empirically determined. The unlabeled set is also updated with additional examples of length not exceeding the iteration number. In that sense, the iteration number is actually an upper bound on the length of the examples, allowing the algorithm to select sequences of a lower length paired with longer sequences. For example, in the second iteration, the unlabeled set also contains examples that pair a sequence of one word with a sequence of two

words. The labeled set after step 3 contains "true" as well as "false" pairs, added by both the deterministic algorithm and the CT classifier, for training the PT classifier. Steps 4 and 5 train and test the classifier PT on the labeled and unlabeled sets respectively. Finally, in step 6, pairs that receive the same label from both classifiers, with a confidence score higher than the predefined threshold, are added to the labeled set with their corresponding label and stay there forever. This labeled set is used as part of the training data in the next iteration. The number of iterations is manually configured upon initialization of the algorithm and at the end, the "true" pairs are deemed paraphrases. The entire process is summarized in Figure 1.

To get a feeling for the robustness of the methodology, we applied the same technique to the task of generating paraphrases in English. English has shallow morphology as compared with Arabic, on one hand, but, on the other hand, uses more words than Arabic to convey the same meaning. Based on this observation, for English, we changed the settings of the data for using a window of size 4 instead of 3 and removed most of the morphology-related features. The English set of features for the CX and PT classifiers are summarized in Tables 4 and 5, respectively.

Feature	Description
Lemma, POS, NER, BP	of each context word
Lemma-match rate	The rate of lemma match on each side of the context

TABLE 4 – The features we use for training the CX classifier on English.

Feature	Description
n-gram score	Normalized *n*-gram frequency score for word sequences up to 4 words (2-4 grams)
POS, NER, BP	of each sequence word
Possessive form	of each sequence word
Sequence length	The number of words in each sequence

TABLE 5 – The features we use for training the PT classifier on English.

Comparable documents were extracted using the same technique from a relatively small part of the English Gigaword (5th ed.) (Parker et al., 2011). We preprocessed the documents using the OpenNLP library. For every word, we determined its part-of-speech, base-phrase and named-entity tags. The lemma of each word was retrieved from WordNet (Fellbaum et al., 1998) by providing it with the surface form and the part-of-speech tag as inferred by OpenNLP. Overall, we found 294 document pairs, containing about 220,000 words. Similar to the evaluation step of the Arabic corpus, we randomly selected 80 document pairs for vetting their correspondence to each other. Out of the selected 80 document pairs, 65 were classified as "yes" instances by both evaluators. Of the other 15 instances, 3 were classified as "yes" by only one evaluator. As for Arabic, the rest of the pairs were actually dealing with the same general domain but not specifically discussing the same event. The inference algorithm for English worked exactly as described above. Recall that in the first iteration on Arabic, we used a deterministic algorithm for labeling some of the data for training the classifiers for the first time. For Arabic, we used the

English gloss values of the Arabic words for finding "false" examples; for English, we use WordNet for the same task in such a way that synonyms are not considered as "false" examples.

4 Results and evaluation

4.1 Experiment settings

Our initial experiments perform only five iterations on both corpora (Arabic, as well as English), which means that we find paraphrases of no longer than five words. The two classifiers are configured with different thresholds. The confidence score given by LibSVM for every classification is a value between 0 and 1; therefore, we experimented with different threshold values and realized that the best settings in this case are obtained when using 0.85 for "true" pairs for the CX classifier and 0.75 for the PT classifier. For the "false" pairs, we use 0.75 for both classifiers. Since we noticed that the number of "false" pairs is much larger than the number of "true" ones in the training data of every iteration, we defined another parameter (currently 6) that limits the factor of "false" pairs allowed in the training data with respect to the "true" pairs.

In the next section, we show some results when running over 240 document pairs in Arabic, containing about 165,000 words, and 40 English document pairs containing about 11,000 words.

4.2 Results

First, we give some statistics on the results obtained by the inference algorithm on both the Arabic and English corpora, in Tables 6 and 7, respectively.

	"false" pairs	**"true" pairs**	**Unique paraphrase pairs**	**Unlabeled pairs**
Initialization	22,885,104	66,317		19,480
After iteration 1	23,799,787	(+1,726) 68,043		3,166,935
After iteration 2	24,759,791	(+3,757) 71,800	954	2,790,574
After iteration 3	25,349,489	(+2,623) 74,423	416	2,198,253
After iteration 4	26,221,889	(+451) 74,874	331	1,557,931
After iteration 5	26,900,833	(+101) 74,975	72	878,987
Total			**1,773**	

TABLE 6 – Statistics and final results of the inference algorithm running on the Arabic corpus.

In both tables, the initialization row shows the number of "true" and "false" examples as was labeled by the deterministic algorithm and the size of the unlabeled examples set. In the following rows, the numbers refer to the results of the specific iteration. The numbers of "true" and "false" pairs reported on every line are the aggregated numbers collected from all previous iterations. Recall that at the beginning of every iteration, a deterministic algorithm adds pairs of word sequences that match on the lemma level, word by word; hence, the number of "true" pairs in every line is the sum of the pairs from the previous iterations, the pairs added by the deterministic algorithm for the next iteration and the paraphrase pairs inferred by the current iteration. The third column, unique paraphrase pairs, is merely the number of unique paraphrase pairs inferred during the current iteration. The parenthesized numbers indicate the difference in

the quantity of "true" pairs from the previous iteration. So, the total number of extracted paraphrases is the number written on the total line in the unique paraphrases column. In Arabic, we found 1,773 paraphrase pairs and in English we found 525. This process can be scaled up for finding more paraphrases. We do not include paraphrases generated after the first iteration because, by definition, they are composed of synonymous words. Recall that, during initialization, the deterministic algorithm adds pairs to the unlabeled set if their paired words are synonyms in English or share the same English gloss, in Arabic. Table 8 shows some statistics for the entire inference process.

	"false" pairs	**"true" pairs**	**Unique paraphrase pairs**	**Unlabeled pairs**
Initialization	876,947	32,972		3,597
After iteration 1	960,840	(+868) 33,840		86,648
After iteration 2	1,058,970	(+1,633) 35,473	230	58,312
After iteration 3	1,109, 746	(+1,194) 36,667	177	21,332
After iteration 4	1,127,643	(+339) 37,006	94	6,677
After iteration 5	1,128,475	(+52) 37,058	24	1,490
Total			**525**	

TABLE 7 – Statistics and final results of the inference algorithm running on the English corpus.

The raw data corpus size is a rough estimation of the amount of words we had in the corpus at the beginning. Note that currently we did not use the entire Gigaword corpora: in Arabic we used about 30% of the entire set and in English we only used about 10% of the documents. The following column shows the number of comparable document pairs we found using the pairing algorithm described above. Since the pairing algorithm was designed to prefer recall over precision, the number of comparable documents is lower than might be expected considering the relatively large number of words we had in the raw corpus. We expect that this number will grow larger once we improve the pairing algorithm. The next column, number of words used in inference, sums up the number of words of the entire set of comparable document pairs from the previous column. The last column shows the number of paraphrase pairs extracted by the inference algorithm.

	Raw data corpus size	**Extracted comparable document pairs**	**Comparable documents used in inference**	**Number of words used in inference**	**Number of inferred unique paraphrases**
Arabic	~20,000,000	690	240	165,369	1,773
English	~1,000,000	294	40	11,600	525

TABLE 8 – General statistics on the entire inference process.

Comparing the results to the results retrieved by other works is difficult because there is neither a shared task for paraphrase extraction nor common resources for comparison. Therefore, we show some manual evaluations of our results. The evaluation was performed by two Arabic-English speakers by going over the reported paraphrases one by one. For each pair, we assigned one label: *P* – indicating correct paraphrase, *E* – indicating unidirectional entailment, *R* – related (for

other semantic relations except antonyms, e.g. San Diego/Los Angeles) and *F* – wrong (including antonyms). Table 9 and 10 summarizes our preliminary evaluation report on Arabic and English, respectively.

Length	Evaluated	*P*	*E*	*R*	*F*	Precision
2	120	49	12	25	34	71%
3	95	45	10	11	31	69%
4	70	26	4	5	35	50%
5	50	24	2	7	20	66%
Total	335	144	28	48	120	**66%**

TABLE 9 – Manual evaluation summary for Arabic. *P*: paraphrases, *E*: unidirectional entailment, *R*: related, *F*: wrong, i.e. unrelated or antonyms.

The evaluation results reported in both tables are based on the agreement of the two evaluators; in other words, we report here only on pairs that were annotated by both evaluators with the same tag. Note that the first column, length, indicates the number of words of the largest phrase included in the evaluated paraphrase pair. Paraphrase pairs containing a single word in both phrases were not evaluated at all. In the last column, we calculate the precision, considering pairs tagged with *P*, *E* and *R* as positive instances. The last row summarizes the results. In Arabic, 66% of the generated paraphrase pairs are at least considered as semantically related; among them, about 43% are considered real paraphrases. In English, only 63% of the paraphrase pairs are considered related, out of which 30% are real paraphrases. As can be seen from the tables, there is no preferred length for the inference algorithm. We see a slight improvement in the precision of paraphrases up to length three; however, this improvement does not seem significant, considering the relatively small amount of evaluated pairs.

When we increased the threshold on the confidence that is used by the PT classifier on English to 0.9, the number of paraphrases reported by the inference algorithm decreased to 330 and the average number of similar words in a pair, increased. As a results of that, the overall precision got improved to 72%, calculated over 250 evaluated pairs. These results helped us understand the effect of the PT classifier on performance. The pairs with a high confidence score, as reported by the PT classifier, are most likely to be real paraphrases; however, in most cases, the word sequences of such a pair share more words in common than do other pairs (e.g. “the U.S. Air Forces” ⇔ “the United States Air Force”).

Length	Evaluated	*P*	*E*	*R*	*F*	Precision
2	120	23	11	37	49	59%
3	60	28	6	9	17	71%
4	50	15	8	8	21	62%
5	25	8	5	2	10	60%
Total	255	74	30	56	97	**63%**

TABLE 10 – Manual evaluation results for English. *P*: paraphrases, *E*: unidirectional entailment, *R*: related, *F*: wrong, i.e. unrelated or antonyms.

Table 11 gives some examples of Arabic, as well as English, paraphrase pairs that were extracted by our inference algorithm. The third column is the evaluation score given by one of the evaluators.

Language	Paraphrase pair	Evaluation score
Arabic	الرئيس الفلسطيني *Alr}ys AlflsTyny* (“the Palestinian president”) ⇔ السلطة الوطنية الفلسطينية *AlslTp AlwTnyp AlflsTynyp* (“the Palestinian authority”)	Related
Arabic	جورج ووكر بوش *jwrj wwkr bw$* (“George Walker Bush”) ⇔ جورج بوش *jwrj bw$* (“George Bush”)	Paraphrases
Arabic	المؤتمر السادس *Alm&tmr AlsAds* (“the Sixth conference”) ⇔ الاجتماع الوزاري السادس *AlAjtmAE AlwzAry AlsAds* (“the Sixth ministerial meeting”)	Paraphrases
Arabic	دانييل جلازر *dAnyyl jlAzr* (“Daniel Glaser”) ⇔ دانيئل غلاسر *dAny}l glAsr* (“Daniel Glaser”)	Paraphrases
Arabic	كيلي غونزاليز وانجولو *kyly gwnzAlyz wAngwlw* (“Kaylie Gonzales and Angelo”) ⇨ الارجنتينيين الخطيرين *AlArjntynyyn AlxTyryn* (“the dangerous Argentinians”)	Unidirectional entailment
Arabic	البرلمان الجديد *AlbrlmAn Aljdyd* (“the new Parliament”) ⇔ المجلس الوطني السابع عشر *Almjls AlwTny AlsAbE E$r* (“the Seventeenth Parliament”)	Paraphrases
Arabic	الحدود السورية اللبنانية *AlHdwd Alswryp AllbnAnyp* (“the Syrian-Lebanese borders”) ⇔ الحدود السورية *AlHdwd Alswryp* (“the Syrian border”)	Unidirectional entailment
English	could veto ⇔ threatened to veto	Related
English	the U.S. Naval Task Force ⇔ a US Naval Task Group	Paraphrases

English	Beijing's policy ⇔ the China's policy	Paraphrases
English	a poor and little-developed province ⇔ its resource-rich northwestern province	Wrong
English	U.S. beef and related products ⇨ beef products	Unidirectional entailment
English	a magnitude 6.0 earthquake ⇨ the quiver	Unidirectional entailment
English	will only endanger ⇔ will not only endanger	Wrong

TABLE 11 – Example results in both Arabic and English.

Conclusions

The method suggested here has demonstrated its potential for inferring paraphrases from a corpus of comparable documents, using "co-training". As we have seen, incorporating morphological features for a highly inflected language, such as Arabic, is very effective. SVM with its generalization property was a natural option for dealing with combination of features that can play an important role for identifying paraphrases. Finding more features that help to properly match the true senses of word sequences is definitely a direction for future investigation. In a similar experiment performed on English, we still saw encouraging results, despite the smaller corpus. In the next stage of research, we plan to scale up the experiments and use more raw data along with an improved document-pairing algorithm for inferring additional paraphrases. We also plan to use those paraphrases within an Arabic-to-English translation system so as to hopefully improve the quality of the translations.

References

Bannard, C. and Callison-Burch, C. (2005). Paraphrasing with bilingual parallel corpora. In *Proceedings of the Annual Meeting of Association of Computational Linguistics (ACL 2005)*, pages 597-604, Ann Arbor, MI.

Bar, K. and Dershowitz, N. (2010). Using synonyms for Arabic-to-English example-based translation. In *Proceedings of the Ninth Conference of the Association for Machine Translation in the Americas (AMTA 9)*, Denver, CO.

Barzilay, R. and McKeown, K. (2001). Extracting paraphrases from a parallel corpus. In *Proceedings of the 39th Annual Meeting of Association of Computational Linguistics (ACL 2001)*, pages 50-57, Toulouse, France.

Benajiba, Y., Diab, M., and Rosso, P. (2008). Arabic named entity recognition: An SVM-based approach. In *Proceedings of the Arab International Conference on Information Technology (ACIT-2008)*, Hammamet, Tunisia.

Black, W., Elkateb, S., Rodriguez, H., Alkhalifa, M., Vossen, P., Pease, A., and Fellbaum, C. (2006). Introducing the Arabic WordNet project. In *Proceedings of the 3rd Global Wordnet Conference (GWC 2006)*, pages 295-299, Jeju Island, South Korea.

Blum, A. and Mitchell, T. (1998). Combining labeled and unlabeled data with co-training. In *Proceedings of 11th Annual Conference on Computational Learning Theory (COLT)*, pages 92–100, Madison, WI.

Buckwalter, T. (2002). Buckwalter Arabic morphological analyzer Version 1.0. LDC catalog number LDC2002L49.

Callison-Burch, C. (2008). Syntactic constraints on paraphrases extracted from parallel corpora. In *Proceedings of the Conference for Empirical Methods in Natural Language Processing (EMNLP 2008)*, pages 196-205, Honolulu, HI.

Callison-Burch, C., Koehn, P., and Osborne, M. (2006). Improved statistical machine translation using paraphrases. In *Proceedings of the North American Association for Computational Linguistics (NAACL 2006)*, New York City, NY.

Chang, C. C. and Lin, C. J. (2011). LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2:27:1-27:27. Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm.

Dagan, I. and Glickman, O. (2004). Probabilistic textual entailment: Generic applied modeling of language variability. In *PASCAL workshop on Learning Methods for Text Understanding and Mining*, pages 26-29, Grenoble, France.

Denkowski, M., Al-Haj, H., and Lavie, A. (2010). Turker-assisted paraphrasing for English-Arabic machine translation. In *Proceedings of the Workshop on Creating Speech and Language Data with Amazon's Mechanical Turk at the Conference of Human Language Technology-North American Association for Computational Linguistics (HLT-NAACL 2010),* pages 66-70, Los Angeles, CA.

Diab, M. (2009). Second generation tools (AMIRA 2.0): Fast and robust tokenization, POS tagging, and base phrase chunking. *MEDAR 2nd International Conference on Arabic Language Resources and Tools*, Cairo, Egypt.

Diab, M., Hacioglu, K., and Jurafsky, D. (2004). Automatic tagging of Arabic text: From raw text to base phrase chunks. In *Proceedings of Human Language Technology-North American Association for Computational Linguistics (HLT-NAACL 2004)*, pages 149-152, Boston, MA.

Dolan, B., Quirk, C., and Brockett, C. (2004). Unsupervised construction of large paraphrase corpora: Exploiting massively parallel news sources. In *Proceedings of the 20th International Conference on Computational Linguistics (COLING 2004),* Geneva, Switzerland.

Duboue, P. A. and Chu-Carroll, J. (2006). Answering the question you wish they had asked: The impact of paraphrasing for question answering. In *Proceedings of Human Language Technology-North American Association for Computational Linguistics (HLT-NAACL 2006)*, pages 33-36, New York City, NY.

Fellbaum, C. (1998). *WordNet: An Electronic Lexical Database*. MIT Press, Cambridge, MA.

Garofolo, J. (2002). NIST OpenMT Eval. http://www.itl.nist.gov/iad/mig/tests/mt/2002/.

Habash, N. and Rambow, O. (2005). Arabic tokenization, morphological analysis, and part-of-speech tagging in one fell swoop. In *Proceedings of the Conference of American Association for Computational Linguistics*, pages 578-580, Ann Arbor, MI.

Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., and Witten, I. H. (2009). The WEKA data mining software: An update. *SIGKDD Explorations*, volume 11, issue 1.

Maamouri, M., Graff, D., Bouziri, B., Krouna, S., Bies, A., and Kulick, S. (2010). Standard Arabic morphological analyzer (SAMA), Version 3.1. *Linguistic Data Consortium*, Philadelphia, PA.

Marton, Y., Callison-Burch, C., and Resnik, P. (2009). Improved statistical machine translation using monolingually-derived paraphrases. In *Proceedings of the Conference for Empirical Methods in Natural Language Processing (EMNLP 2009)*, Singapore.

Papineni, K., Roukos, S., Ward, T., and Zhu, W. J. (2002). Bleu: A method for automatic evaluation of machine translation. In *Proceedings of the ACL 40th Annual Meeting*, pages 311-318, Philadelphia, PA.

Parker, R., Graff, D., Chen, K., Kong, J., and Maeda, K. (2009). Arabic Gigaword, Fourth Edition. *Linguistic Data Consortium*, Philadelphia, PA.

Parker, R., Graff, D., Chen, K., Kong, J., and Maeda, K. (2011). English Gigaword, Fifth Edition, *Linguistic Data Consortium*, Philadelphia, PA.

Riezler, S., Vasserman, A., Tsochantaridis, I., Mittal, V., and Liu, Y. (2007). Statistical machine translation for query expansion in answer retrieval. In *Proceedings of Association for Computational Linguistics (ACL 2007)*, pages 464-471, Prague, Czech Repulic.

Roth, R., Rambow, O., Habash, N., Diab, M., and Rudin, C. (2008). Arabic morphological tagging, diacritization, and lemmatization using lexeme models and feature ranking. In *Proceedings of Association for Computational Linguistics (ACL 2008)*, pages 117-120, Columbus, Ohio.

Salloum, W. and Habash, N. (2011). Dialectal to standard Arabic paraphrasing to improve Arabic-English statistical machine translation. In *Proceedings of the Dialects workshop at the Conference for Empirical Methods in Natural Language Processing (EMNLP 2011)*. Edinburgh, UK.

Vapnik, V. and Cortes, C. (1995). Support vector networks. *Machine Learning*, vol. 20, pages 273-297.

Wang, R. and Callison-Burch, C. (2011). Paraphrase fragment extraction from monolingual comparable corpora. In *Proceedings of Fourth Workshop on Building and Using Comparable Corpora (BUCC)*, Istanbul, Turkey.

Zhao, S., Wang, H., Liu, T., and Li, S. (2008). Pivot approach for extracting paraphrase patterns from bilingual corpora. In *Proceedings of the Annual Meeting of Association of Computational Linguistics (ACL 2008*), pages 780-788, Columbus, OH.

Harvesting parallel text in multiple languages with limited supervision

Luciano Barbosa[1] *Vivek Kumar Rangarajan Sridhar*[1]
Mahsa Yarmohammadi[2] *Srinivas Bangalore*[1]
(1) AT&T Labs - Research, 180 Park Avenue, Florham Park, New Jersey 07932
(2) Oregon Health & Science University, Beaverton, Oregon 97006
lbarbosa,vkumar,srini@research.att.com, yarmoham@ohsu.edu

Abstract
The Web is an ever increasing, dynamically changing, multilingual repository of text. There have been several approaches to harvest this repository for bootstrapping, supplementing and adapting data needed for training models in speech and language applications. In this paper, we present semi-supervised and unsupervised approaches to harvesting multilingual text that rely on a key observation of *link collocation*. We demonstrate the effectiveness of our approach in the context of statistical machine translation by harvesting parallel texts and training translation models in 20 different languages. Furthermore, by exploiting the *DOM trees* of parallel webpages, we extend our harvesting technique to create parallel data for resource limited languages in an unsupervised manner. We also present some interesting observations concerning the socio-economic factors that the multilingual Web reflects.

Keywords: web crawling, parallel text, document model object tree, machine translation.

1 Introduction

The amount of information and knowledge in the World Wide Web (WWW) is increasing at a rapid rate. As of September 2011, approximately 500 million websites were estimated to be present in the WWW; a jump of 1000% from 1995 [1]. An increasingly large proportion of these websites are in non-English languages. For example, the total proportion of English webpages on the WWW has been estimated to have dropped from 80% in 1996 to about 45% in 2008 (Pimienta et al., 2009). Furthermore, only 27% of the internet users claim English as their native language (Miniwatts Marketing Group, 2011). The opportunity to capitalize on this new market has encouraged internet service providers to provide web content in multiple languages. As a result, the Web has become an attractive resource for acquiring structured and unstructured data across several low-resource languages and domains.

Multilingual web data has been especially useful in a variety of natural language processing tasks such as information retrieval, language modeling and machine translation. Statistical machine translation requires the creation of a large corpus of parallel text, i.e., translations of text across languages. Harvesting such data automatically by exploiting multilingual webpages provides a low-cost, scalable alternative to expensive expert human translations. Moreover, bilingual subject matter experts may be extremely difficult to find for certain

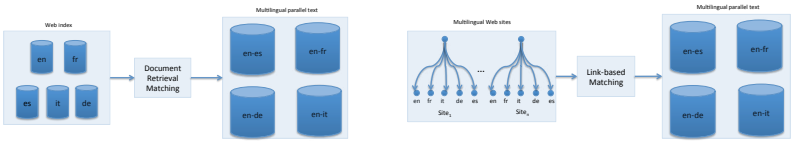
[1] http://news.netcraft.com/archives/2011/09/06/september-2011-web-server-survey.html

Proceedings of COLING 2012: Technical Papers, pages 201–214,
COLING 2012, Mumbai, December 2012.

language pairs. As a result, web harvesting of parallel text has been addressed extensively in the recent past (Resnik and Smith, 2003; Shi et al., 2006; Pomikálek, 2008; Utiyama et al., 2009; Hong et al., 2010; Almeida and Simões, 2010; Uszkoreit et al., 2010).

Parallel text acquisition can be performed by examining the Web in either an unstructured or structured way. In the unstructured view of the Web, a large Web index is used as a starting point and the webpages are matched using cross-language document retrieval (see Figure 1(a)). The basic idea behind such an approach is to use a seed translation model to translate the entire index in one particular language and then match the pages using document retrieval techniques (Uszkoreit et al., 2010). The feasibility of such an approach is dependent on the availability of a large Web index as well as computational power to perform large scale machine translation and document alignment. In the structured approach, websites containing parallel text are typically identified using search engine APIs and the crawler proceeds to identify new websites based on the preceding ones (Resnik and Smith, 2003; Shi et al., 2006; Utiyama et al., 2009; Hong et al., 2010). However, such a scheme is amenable primarily for pair of languages.

Our work takes a structured view of the Web and exploits the link structure of websites to collect multilingual parallel text. We leverage the property that multilingual websites typically provide content simultaneously in several languages. Furthermore, a link that represents an entry point to a particular language in these websites usually co-occurs in the DOM tree with entry points to other languages (see Figure 1(b)). The co-occurring language versions of webpages on a particular website is influenced by geographic and economic factors of the underlying service or business. For example, a hospitality website identified as a possible source for harvesting English-French parallel text may also contain German, Italian and Spanish versions of the site, whereas a website with English-Chinese parallel text may have corresponding Japanese and Korean counterparts.



(a) Unstructured view of Web: Web index based parallel text acqusition

(b) Structured view of Web: Multilingual crawling using language co-occurrence

Figure 1: Strategies for harvesting parallel text across multiple languages from the Web

In this work, we present a framework for harvesting parallel text across multiple language pairs using the approach illustrated in Figure 1(b). We present semi-supervised and unsupervised approaches to harvesting multilingual text that rely on a key observation of *link collocation*, i.e., entry points to different languages on multilingual websites are often collocated on the HTML DOM tree. Subsequently, we use an intra-site crawler and a suite of alignment procedures to generate parallel text across multiple pairs of languages and evaluate their utility in machine translation. We demonstrate significant improvements in translation quality for almost all of the 20 language pairs (with English as the source language) in the Europarl corpus (Koehn, 2005) through the addition of parallel text harvested through our approach. We also report experiments in English-Hindi translation

by using a completely unsupervised approach.

The rest of the paper is organized as follows. In Section 2, we describe related work in web mining for parallel text and contrast our work with prior efforts. We describe the semi-supervised approach for obtaining multilingual entry points in a website in Sections 3 and 4 followed by a description of the overall framework for harvesting parallel text from the entry points. In Section 6 we present statistical machine translation experiments using the data harvested through our framework and present a detailed case study for English-Hindi translation in Section 7. We provide a brief discussion in Section 8 and conclude in Section 9 along with directions for future work.

2 Related Work

Prior research on acquiring parallel text from the Web has also focused primarily on a specific pair of languages. Even though in principle the algorithm and framework can be extended to other pairs of languages, it requires either parallel webpages or comparable documents to trigger the process. For example, (Resnik and Smith, 2003) used a crawling procedure to harvest parallel text in English-Arabic starting from the Internet Archive and using several language specific patterns in the URLs. The work in (Munteanu and Marcu, 2005) matches comparable documents in English-Chinese and English-Arabic and subsequently extracts parallel text while (Fung and Cheung, 2004) extract parallel text from quasi-comparable documents in English-Chinese. The work in (Hong et al., 2010; Shi et al., 2006) uses several web crawling strategies for harvesting parallel text in English-Chinese and the work in (Utiyama et al., 2009) addresses the extraction of Japanese-English parallel text from mixed language pages.

Conventionally, the crawling procedure to detect websites containing parallel text has been through a query-based approach (Resnik and Smith, 2003; Chen and Nie, 2000; Hong et al., 2010). They submit carefully constructed queries to a search engine that might yield potential parallel webpages. However, such a procedure depends on the quality of the query strategy as well as the Web index provided by the search engine. Furthermore, the process is typically constrained for particular pair of languages. Subsequently, the websites are mined independently and matched through a variety of techniques ranging from document retrieval (Munteanu and Marcu, 2005) to matching based on HTML document object model (DOM) tree similarity (Resnik and Smith, 2003; Shi et al., 2006). Once the documents are aligned, the sentences are aligned next using dynamic programming based on sentence length and a bilingual dictionary (Resnik and Smith, 2003) or through a classifier (Munteanu and Marcu, 2005).

Multilingual parallel text extraction has been specifically addressed in (Uszkoreit et al., 2010). However, the starting point for harvesting parallel text is a large Web index that is aligned using a seed translation model and subsequent document matching. It may not be feasible to acquire such a large index or run computationally expensive document matching for many efforts. Furthermore, their work reports translation quality experiments for about 7 language pairs and requires a seed translation model from the language of interest into English. Our interest, on the other hand, lies in crawling the Web to detect multilingual pairs of entry points belonging to websites that contain parallel text.. We push the task of identifying parallel web pages upfront to the crawler instead of downstream document matching over a large snapshot of the Web. Our framework can be used with or without a seed bilingual dictionary or translation model. In the absence of a seed dictionary, we

use an unsupervised DOM tree similarity procedure to harvest parallel text as we describe later.

3 Background: Bilingual crawler

The simplest manifestation of a multilingual crawler is for a pair of languages, i.e., bilingual crawling. To perform this task, the crawler needs to detect bilingual sites by traversing interesting regions of the Web. The bilingual site detector (BiSite detector) is the component responsible for determining whether a website contains bilingual content. The detector performs its task in two phases: link-based prediction and language identification. The role of the *link predictor* is to predict links that are entry points to a particular language in a website. The *link predictor* relies on the property that these entry points contain some common link pattern. For instance, entry points to the French content might have words as "fr" or "francais" in their URLS. In order to be able to handle different types of patterns in the links, it uses features in 5 different contexts: tokens in the URL, anchor, around the link, image alt and image src tags. Thus, for each language, a link predictor is built using supervised learning. Subsequently, the BiSite detector verifies if the pages whose links were considered relevant by the link predictor are in the languages of interest. Once a pair of links in a website are hypothesized as entry points in two different languages, the crawler uses an intra-site crawling policy similar to that described in (Rangarajan Sridhar et al., 2011) to traverse the Web sites and collect the parallel content. Figure 2 depicts a simple illustration of the bilingual crawler. Further details about the BiSite detector is presented in (Barbosa et al., 2011).

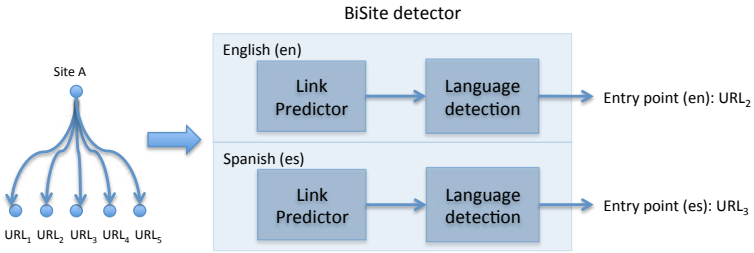


Figure 2: BiSite detector detects entry points to parallel text for a pair of languages on a given website.

4 From bilingual to multilingual crawling

The task of a multilingual crawler is to detect websites that contain content in multiple languages. A simple way to perform this task is to extend the approach used in the BiSite detector by building entry point detectors for each language of interest. The BiSite detector was built by labeling positive and negative examples of entry points in a pair of languages. Although effective, a fully supervised approach is not feasible for building entry point detectors for many pairs of languages as it requires significant labeling effort.

We propose a semi-supervised approach for building multilingual detectors. Our approach relies on the observation that, in a given website, entry points to different languages are

collocated links on the same page. The bootstrapping algorithm works as follows: BiSite detectors in a small set of language pairs are constructed using manual labeling and are in turn used to identify entry points in these languages. The algorithm then extracts links collocated with the detected entry points, generating training data to build detectors in new language pairs. The new detectors can now be used in the first phase of the bootstrapping, and iterated to generate more entry points. Figure 3 describes the components of the bootstrapping algorithm. Therefore, the only supervision provided is positive (entry points) and negative examples for the initial detectors whose accuracy the bootstrapping algorithm heavily relies on.

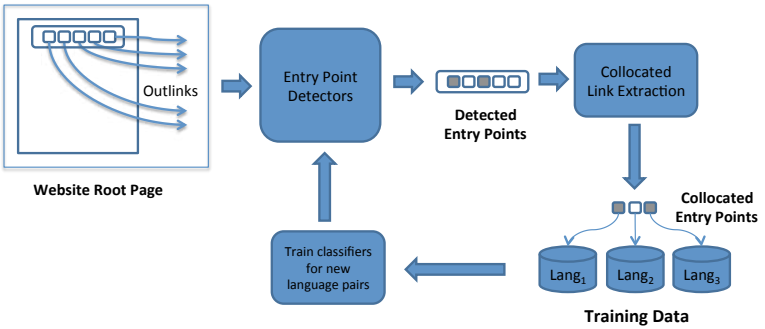


Figure 3: Bootstrapping algorithm for creating classifiers for new pairs of languages

4.1 Extraction of co-occurring links

The extraction of co-occurring links is an important step in the bootstrapping algorithm. Its goal is to extract candidate entry points collocated with entry points identified by the initial BiSite detectors. Formally, let OL be the set of *outlinks*[2] on a given Web page WP, and EP be the set of language entry points in WP ($EP \subseteq OL$). The algorithm's objective is to identify EP, if it exists. Our assumption is that this subset is the one with highest diversity with respect to languages among OL. In other words, while most of the outlinks in OL point to pages in similar languages, the links in EP point to pages in distinct languages. The algorithm then searches for the partition of OL, $\widehat{EP}$, with the maximum normalized entropy with respect to its languages among the subsets ($\mathbf{SL} = \{SL_1, \cdots, SL_K\}$) of OL:

$$\widehat{EP} = \underset{\mathbf{SL_i} \in \mathbf{SL}}{\arg\max}\ NormEntropy(\mathbf{SL_i}) \tag{1}$$

$$NormEntropy(SL_i) = \frac{-\sum_{k=1}^{|L_i|} p(l_k) log(p(l_k))}{log(|SL_i|)} \tag{2}$$

where $L_i = l_{i1}, ...l_{i|L|}$ is the set of the languages in SL_i and $p(l_k) = count(l_k)/|SL_i|$. NormEntropy is equal to 1 when SL_i is composed of distinct languages.

Since finding an optimal subset of links is a combinatorial search problem, our algorithm adopts a greedy approach. The algorithm imposes the following constraints to make the

[2] *Outlinks* are the links pointing out from a given webpage.

solution tractable: (1) $\widehat{EP}$ must contain the entry points, $iniEPs$, detected by initial detectors; (2) the elements of $\widehat{EP}$ are not spread over the entire Web page but located in a similar region in the page's DOM tree. Starting from a detected entry point $iniEP_i$, the algorithm first locates the DOM node ($node_i$), represented by the "a href" HTML tag, associated to $iniEP_i$ and calculates the normalized entropy of the subset of OL contained in the DOM subtree of the parent of $node_i$. In the next step, it goes up one level in the tree and calculates the normalized entropy of the subtree. If the normalized entropy of the current subtree is higher than the previous one, it continues the search up one more level, otherwise stops. The links of the best subtree identified by this process SL_i are considered as the candidates associated with $iniEP_i$. This procedure is repeated for each $iniEP_i$ and the set SL_i with the highest entropy among the candidate sets is considered for the final step. Subsequently, the algorithm discards elements in SL_i that are from the same language as we assume that there is only one single entry point for each language in a given page. Finally, the output of this process is a set of candidate entry points (outlinks) in which each outlink is associated to a particular language. For each outlink, its link neighborhood is extracted (described previously) and used as a positive example for building a *link predictor* for that particular language.

4.2 Evaluation of semi-supervised multilingual crawler

In this section, we assess the quality of the entry points identified through our bootstrapping algorithm. As inputs to the algorithm we provided two entry point detectors (English and Spanish), created using labelled data, and 10,000 Web sites, collected using an unrestricted crawler. We then ran the algorithm over these Web sites and candidate entry points were extracted in 45 different languages. Figure 4 shows the top-10 languages collocated with English and Spanish in this dataset. The most popular languages were European languages while Japanese was the most popular Asian language, beating European languages such as Polish and Slovenian.

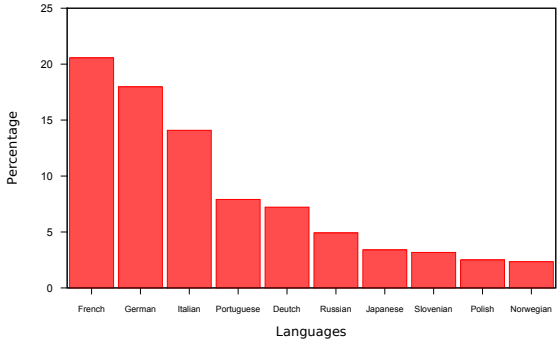


Figure 4: Distribution of languages collocated with English and Spanish

For 5 of the top-10 languages: French, German, Portuguese, Russian and Japanese, we manually labelled test data (about 500 positive and 1,000 negative examples for each language) to evaluate the performance of their respective detectors created using the semi-

supervised approach. We also assessed the accuracy of the co-occurring link extraction (CLE) by manually inspecting how many entry points extracted from the CLE procedure were correct. Table 1 presents the precision of CLE and F-measure of the detectors. For each of the top 5 languages the collocated link extraction algorithm achieves greater than 90% precision. The detectors created automatically by the bootstrapping algorithm have a high F-measure (from 0.8 to 0.87). This is a direct consequence of the high precision obtained by the CLE algorithm that provides the positive examples for training the classifiers.

Language	**F-measure (Detector)**	**Accuracy (CLE)**
German	0.87	0.96
French	0.85	0.93
Japanese	0.85	0.95
Russian	0.8	0.98
Portuguese	0.8	0.95

Table 1: Precision and F-measures for the detectors constructed using the semi-supervised entry point detector approach

5 Parallel text acquisition

The multilingual crawler generates pairs of entry points for multiple language pairs that subsequently need to be mined for parallel text. We adopt a recursive intra-site crawling approach that aligns text and URLs across the initial entry points to harvest parallel text. Our framework uses a document matching framework to align the URLs (Munteanu and Marcu, 2005) and a bilingual dictionary based dynamic programming match to align the sentences across the hypothesized parallel documents (Ma, 2006). The document matching process is constrained by a window size that is dependent on the total number of parallel URLs that need to be aligned. The framework also enables us to highly parallelize mining across multiple language pairs. The bilingual dictionary in this work was obtained by performing automatic word alignment on seed parallel data (Och and Ney, 2003). The harvested bitext was then filtered using a word-overlap filter as well as a source and target vocabulary restriction filter. By varying the thresholds for the various filters, one can control the amount and quality of the bitext. We also check for the fidelity of the translation by matching a subset of the harvested text for each website against translations from Google Translate and Microsoft Bing. We omit the data from the entire website if the translations have a high correlation with the online translation engines (cosine distance ≥ 0.8). This step is performed to avoid the use of machine translated parallel text. A detailed description of the intra-site mining procedure can be found in (Rangarajan Sridhar et al., 2011).

6 Machine translation experiments

In this section, we validate the quality of the parallel text obtained using our multilingual crawling approach through machine translation experiments. Our objective is to evaluate the translation quality with and without the parallel text harvested from the Web for a large number of language pairs. We used the Moses[3] toolkit (Koehn et al., 2007) for performing phrase-based translation experiments. The standard pipeline (sentence alignment using GIZA++, phrase extraction with maximum phrase length of 7 using *grow-diag-final* option, lexicalized reordering model with *msd-bidirectional-fe* option) was used to build the

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

models. The language models were interpolated Kneser-Ney discounted trigram models, all constructed using the SRILM toolkit (Stolcke, 2002). The language models were constructed only from the target side of the bitext for each language pair. We performed minimum error rate training (MERT) on a development set to optimize the weights of the log-linear model.

We ran machine translation experiments for all of the 20 languages (English as source language) present in the Europarl corpus (Koehn, 2005). The baseline models were trained on Europarl data. For each of the 20 languages, we harvested parallel text (see Section 5) from the entry points hypothesized by the multilingual crawler. Statistics of the parallel text obtained using our procedure is shown in Table 2. Subsequently, we trained a translation model by combining the Europarl text with the parallel text harvested using our approach.

Language pairs	# websites	# webpages	# parallel sentences	Language pairs	# websites	# webpages	# parallel sentences
en-bg	253	290	825	en-it	14538	79582	1654730
en-cs	5144	13058	250598	en-lt	200	1154	30725
en-da	3384	9678	138482	en-lv	128	298	10771
en-de	33582	117043	1215186	en-nl	8295	35109	568534
en-el	705	1908	9645	en-pl	1894	7184	154508
en-es	13075	56687	1347795	en-pt	3542	11900	268065
en-et	288	1766	37141	en-ro	767	2627	51498
en-fi	682	1680	20355	en-sk	443	2556	87172
en-fr	15429	54315	1140140	en-sl	327	2001	59935
en-hu	1481	6492	129291	en-sv	3390	14382	247326

Table 2: Parallel text obtained using our framework for the 20 languages in Europarl corpus

Figure 5 shows the BLEU scores across the 20 languages with and without the addition of parallel text from the Web. We performed the decoding experiments on three different test sets: Europarl, Web and Europarl+Web. We chose the three sets to represent a variety of test domains. A total of 1000 sentences was used for testing. The test sentences from the Web were chosen randomly such that they satisfied a sentence length constraint (≥ 5 words). We spot checked a small subset of the test sets to ascertain the fidelity. Ideally, a manually constructed general vocabulary test set for each of the language pairs would have been the best choice. However, it is expensive and difficult to create test sets through human experts for each of the 20 language pairs. All the models are optimized based on a randomly chosen development set from Europarl comprising of 1000 sentences.

Overall, the results in Figure 5 indicate significant improvements in BLEU scores when the harvested parallel text is added to the Europarl data. As expected, the improvements are not quite marked when tested on Europarl data, however, they are significant on data obtained from a general domain (a mix of websites). Our objective is to translate sentences closer to a general domain (hospitality, business, medicine, etc.) and we show that parallel data harvested from the Web provides significant improvement.

7 Unsupervised parallel text acquisition: English-Hindi

In the previous section, we demonstrated the utility of our multilingual crawler for languages with reasonable resources. However, such data may not be present for several other language pairs. One of the main objectives of our Web crawling scheme is to harvest parallel text

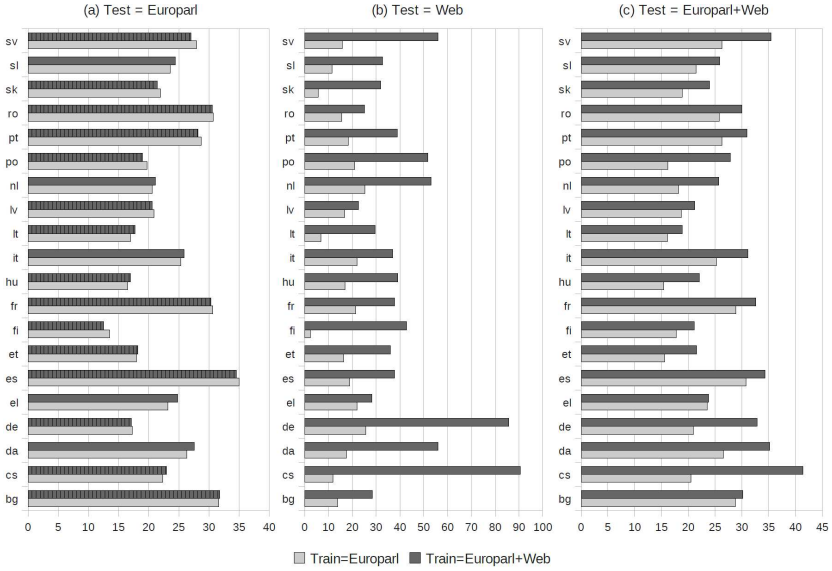


Figure 5: Translation quality as measured through BLEU score for various test sets with and without web crawled parallel text. Hatched bars indicate insignificant difference (Koehn, 2004) with respect to the baseline model built from Europarl data.

for language pairs with limited resources, i.e., lack of publicly available large database or language resources. As an instantiation of this goal, we conducted a detailed study on English-Hindi. We used the collocated link extraction procedure described in Section 4.1 to compile 1638 potential entry points in English-Hindi. Unlike the language pairs used in the previous section, we did not have access to parallel text or a bilingual dictionary for English-Hindi. Hence, we used a completely unsupervised scheme to harvest parallel text from the initial entry points. The intra-site crawler (see Section 5) was modified to perform document matching using the HTML structure of the webpages and the dynamic programming text alignment procedure relied only on sentence length and identity anchor words (words that are present in the source and target sentence). We computed the distance between the DOM trees of two HTML pages to decide if the pages contained parallel text (Pawlik and Augsten, 2011). We ran the crawler for three iterations, each iteration using the parallel entry points identified in the previous step. Figure 6 shows the number of Web sites, pages and bitext harvested using the setup. Since, Hindi characters have a pre-defined range, we could filter out bitext that did not contain Hindi.

From Figure 6 it can be seen that the growth of the entry points hypothesized at each iteration of the intra-site crawling procedure is not linearly related to the bitext harvested. We conjecture that websites containing English-Hindi parallel text do not point to significant number of newer English-Hindi websites. Unlike European languages the domain of the WWW containing English-Hindi parallel text is rather limited.

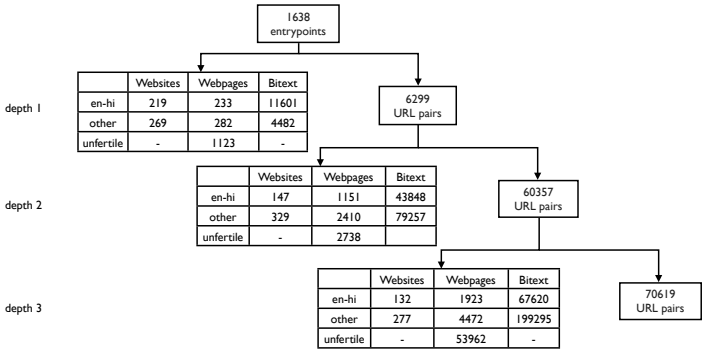


Figure 6: Illustration of the number of websites, webpages and bitext harvested in an unsupervised manner for English-Hindi. *other* refers to bitext in a language pair other than English-Hindi and *unfertile* refers to entry points that did not harvest any bitext.

We also performed machine translation experiments using the English-Hindi parallel text harvested through the unsupervised alignment approach. The baseline model was trained on the Indic multi-parallel corpus (Birch et al., 2011) and a new model was trained by adding the harvested parallel text. Since the parallel text harvested using the unsupervised approach is prone to be noisy, we filtered the sentences using a word-overlap filter constructed from the IBM Model1 dictionary obtained from the baseline translation model. We used the Indic corpus development and test sets for tuning and testing, respectively. The results are reported in Table 3. The results demonstrate a significant improvement ($p = 0.05$) in BLEU score when the Web crawled parallel text is added to the baseline data. The filtering procedure using the dictionary obtained through automatic alignment yields lesser amount of parallel text and hence results in smaller improvement in BLEU score. The experiments clearly indicate the benefit of the parallel text harvested using our scheme. It is important to note that we started this process for English-Hindi with no resources whatsoever.

Training data	**BLEU**
Indic training data	18.6
Indic+unsupervised parallel text	19.4
Indic+ parallel text filtered using IBM Model1 dictionary	19.1

Table 3: BLEU scores on English-Hindi corpus using parallel text harvested form multilingual crawler.

8 Discussion

Languages addressed by multilingual crawler. The multilingual crawler presented in this work generates entry points for over 500 language pairs (with atleast 500 entry points). We have presented machine translation experiments only for a subset of these languages, namely, ones contained in the Europarl corpus and English-Hindi. Some of the prominent language pairs with a large number of entry points are English-Chinese (26893), English-Japanese (25804), English-Russian (24011), English-Turkish (10879), English-Norwegian (6564), English-Arabic (5001), English-Korean (4425), English-Persian (3398), English-

Indonesian (1224), English-Hebrew (1514) and English-Thai (1000). One can potentially harvest parallel text for all of these language pairs and subsequently exploit it for machine translation. Other notable language pairs not containing English as one of the languages are German-French (18874), German-Italian (16275), German-Spanish (14410), German-Dutch (10530), and French-Spanish (16295). It is interesting to note that we can obtain entry points for a multitude of language pairs not containing English. Directly obtaining parallel text for some of these language pairs can obviate the need to use English as a pivot language during translation.

Distribution of parallel text on the Web in European languages. Based on the parallel text extracted for the language pairs in the Europarl data (see Table 2), English-Bulgarian (en-bg) is the poorest language pair in terms of net harvested parallel text. The total bitext of 825 sentences is obtained from 253 websites, i.e., the yield per website is only about 3 sentences. Most of the websites that contained any Bulgarian translations did so only at the top level (menu items, homepage information, etc.) and did not have any significant content translated across internal webpages. English-Greek (en-el) is another language pair with very low parallel content on the Web. English-Estonian (en-et), English-Lithuanian (en-lt), English-Slovakian (en-sk) and English-Slovenian (en-sv) are the language pairs with the highest density of parallel text per website. While there are not many websites with parallel text in these languages, the ones that do contain them are very fertile. As expected English-French (en-fr), English-Spanish (en-es), English-German (en-de), and English-Italian (en-it) language pairs produce large amounts of parallel text in comparison with other languages due to the dominance of these European languages on the Web. Our experiments also indicate that on an average there are approximately 3-6 unique webpages for each website that yields parallel text.

Language co-occurrence. Figure 7 presents a graph showing how the languages are collocated in our data. The edges between languages represent that entry points of these languages are collocated on the same websites. Given a language, we calculated the proportion of languages collocated with it. The edge weight in the graph represents this proportion. Since almost all languages are collocated with each other, we pruned some edges to make the figure clearer. For that, we removed edges with weights lower than 0.05, deleted the English node, since it is the most popular collocated language, and some languages from middle Europe. First thing one can note is that German is the most popular language after English. One can also observe there are some language clusters mainly due to geographic or social/economic aspects: the Western European (German, French, Spanish, Italian, Dutch and Portuguese), the Eastern European (Russian, Estonian, Ukrainian, Latvian, Lithuanian and Bulgarian), the Middle Eastern (Arabic, Urdu and Farsi) and the Far East Asian (Japanese, Korean, Chinese Mandarin and Taiwanese Mandarin). Japanese has a high collocation with European languages because many European websites provide content in Japanese. There is a high collocation of German with Turkish. We believe the reason for that is the high population of Turkish in Germany.

Socio-linguistic observations. Another study that we performed was to obtain the distribution of the languages in the multilingual websites per each country (here we relied on country top-level domain codes). There are countries where very few multilingual sites are available, e.g., India and Cuba, whereas others such as Germany, Netherlands, China and Japan provide a great number of multilingual sites. As one expected, English is the

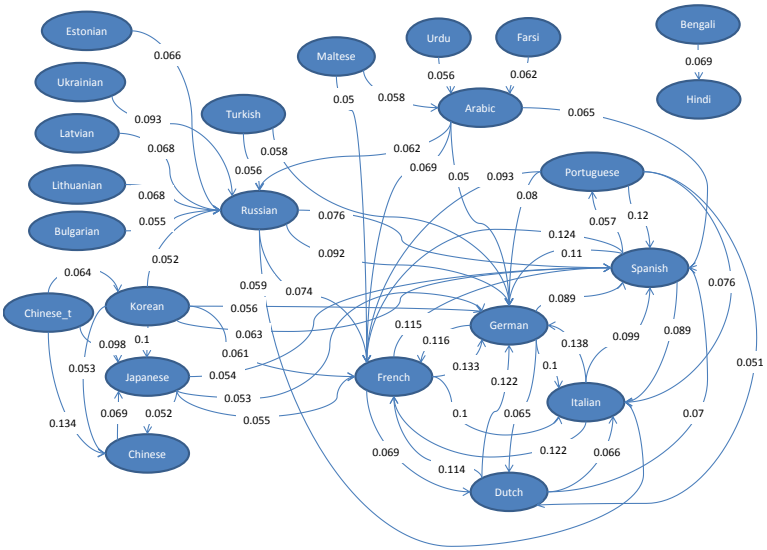


Figure 7: Distribution of language collocations.

foreign language most popular in all countries. It is interesting to note that the popularity of collocated languages in a country depends on geographic and social/economic aspects. Here are some examples:

- Geographic: the most popular foreign languages in Czech multilingual websites (.cz) are in this order: English, German, Russian, Slovak and Polish. These languages are either from neighbouring countries or from economic powers. Interestingly, however Czech is not a very popular language in Polish websites (.pl).
- Social/economic: In Israel, the number of websites in Russian is comparable to Hebrew and English, due to the Russian immigrants. In Iran, French is the most popular language after English, Farsi and Arabic, for historical reasons. Portuguese is the most popular European language in Japanese websites (probably to reach the huge number of Brazilian immigrants that live there).

9 Conclusion

We presented a novel semi-supervised approach for detecting parallel text across multilingual websites. Our approach takes a structured view of the Web and crawls regions of the Web that are likely to produce significant amount of parallel text. First, we constructed supervised classifiers for a few language pairs to detect websites with potential parallel text. By exploiting the property that in many multilingual websites, entry points to different languages are often collocated on the HTML DOM tree, we use a collected link extraction

algorithm to extract entry points for new language pairs. Subsequently, the data is used to train supervised classifiers to detect entry points for new language pairs. Starting from a classifier trained to detect English-Spanish entry points, we were able to obtain entry points in 45 language pairs. We used an intra-site crawling approach to mine the entry points and align the text using document retrieval and dynamic programming techniques. We demonstrated significant improvements in translation quality for all of the 20 languages in the Europarl corpus. We contrasted the experiments conducted on Europarl with those performed on English-Hindi that did not have any resources to learn a seed translation model or dictionary for sentence alignment. Finally, we also presented some socio-linguistic observations inferred through our crawling procedure. We plan to conduct experiments on the other 24 language pairs not reported in this work as part of future work. We are interested in acquiring parallel text for languages with no or low resources and identifying websites that can be mined frequently due to their dynamic nature (e.g., news, broadcasting stations, etc.).

References

Almeida, J. and Simões, A. (2010). Automatic parallel corpora and bilingual terminology extraction from parallel web sites. In *Proceedings of the 3rd Workshop on Building and Using Comparable Corpora*.

Barbosa, L., Bangalore, S., and Rangarajan Sridhar, V. K. (2011). Crawling back and forth: Using back and out links to locate bilingual sites. In *Proceedings of 5th International Joint Conference on Natural Language Processing*, pages 429–437, Chiang Mai, Thailand. Asian Federation of Natural Language Processing.

Birch, L., Callison-Burch, C., Osborne, M., and Post, M. (2011). The indic multi-parallel corpus. `http://homepages.inf.ed.ac.uk/miles/babel.html`.

Chen, J. and Nie, J. (2000). Parallel web text mining for cross-language IR. In *RIAO*, volume 1, pages 62–78.

Fung, P. and Cheung, P. (2004). Multi-level bootstrapping for extracting parallel sentences from a quasi-comparable corpus. In *Proceedings of the 20th international conference on Computational Linguistics*.

Hong, G., Li, C.-H., Zhou, M., and Rim, H.-C. (2010). An empirical study on web mining of parallel data. In *Proceedings of the 23rd International Conference on Computational Linguistics*.

Koehn, P. (2004). Statistical Significance Tests for Machine Translation Evaluation. In *Proceedings of EMNLP*.

Koehn, P. (2005). Europarl: A parallel corpus for statistical machine translation. In *MT Summit*.

Koehn, P., Hoang, H., Birch, A., Callison-Burch, C., Federico, M., Bertoldi, N., Cowan, B., W., S., Moran, C., Zens, R., Dyer, C. J., Bojar, O., Constantin, A., and Herbst, E. (2007). Moses: Open source toolkit for statistical machine translation. In *Proceedings of ACL*.

Ma, X. (2006). Champollion: A robust parallel text sentence aligner. In *LREC 2006: Fifth International Conference on Language Resources and Evaluation*, Genova, Italy.

Miniwatts Marketing Group (2011). Number of internet users by language. *Internet World Stats*.

Munteanu, D. S. and Marcu, D. (2005). Improving machine translation performance by exploiting non-parallel corpora. *Computational Linguistics*, 31(4):477–504.

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

Pawlik, M. and Augsten, N. (2011). RTED: A robust algorithm for the tree edit distance. *Proceedings of VLDB Endowment*, 5(4):334–345.

Pimienta, D., Prado, D., and Blanco, A. (2009). Twelve years of measuring linguistic diversity in the internet: balance and perspectives. *Paris: Unesco*.

Pomikálek, J. (2008). *Building parallel corpora from the Web*. PhD thesis, Masarykova univerzita.

Rangarajan Sridhar, V. K., Barbosa, L., and Bangalore, S. (2011). A scalable approach to building a parallel corpus from the Web. In *Proceedings of Interspeech*.

Resnik, P. and Smith, N. A. (2003). The web as a parallel corpus. *Computational Linguistics*, 29(3):349–380.

Shi, L., Niu, C., Zhou, M., and Gao, J. (2006). A DOM tree alignment model for mining parallel data from the web. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*.

Stolcke, A. (2002). SRILM – an extensible language modeling toolkit. In *Proceedings of ICSLP*.

Uszkoreit, J., Ponte, J. M., Popat, A. C., and Dubiner, M. (2010). Large scale parallel document mining for machine translation. In *Proceedings of the 23rd International Conference on Computational Linguistics*.

Utiyama, M., Kawahara, D., Yasuda, K., and Sumita, E. (2009). Mining parallel texts from mixed-language web pages. In *Proceedings of MT Summit*.

An Evaluation of Statistical Post-editing Systems Applied to RBMT and SMT Systems

Hanna Béchara[1,2] *Raphaël Rubino*[1]
Yifan He[3] *Yanjun Ma*[4] *Josef van Genabith*[1,2]
(1) NCLT, School of Computing, DCU, Dublin, Ireland
(2) CNGL, School of Computing, DCU, Dublin, Ireland
(3) CIMS, NYU, New York, USA
(4) Baidu, Beijing, China
hbechara@computing.dcu.ie, raphael.rubino@computing.dcu.ie,
yhe@cs.nyu.edu, yma@baidu.com, josef@computing.dcu.ie

ABSTRACT

Statistical post-editing (SPE) of the output produced by rule-based MT (RBMT) systems has been reported to produce extraordinary BLEU (and other automatic evaluation) score improvements. SPE has also been applied to the output of statistical MT (SMT) systems, albeit with more mixed results. We present a statistical post-editing pipeline and evaluate the outputs using automatic and human evaluation techniques, comparing the two SPE pipeline systems (RBMT + SPE and SMT + SPE) with the pure RBMT and SMT system, in an SPE scenario that uses independently existing bitext data, rather than manually corrected first stage MT output, as its training data. Our results show that although automatic evaluation metrics favour the pure SMT system, human evaluators prefer the output provided by the statistically post-edited RBMT system.

KEYWORDS: Rule-Based Machine Translation, Statistical Machine Translation, Statistical Post-Editing.

Proceedings of COLING 2012: Technical Papers, pages 215–230,
COLING 2012, Mumbai, December 2012.

1 Introduction

Human evaluation is a core component of shared tasks such as WMT, and is often considered the gold standard in evaluation of translation systems. Automatic evaluation metrics, however, are much less costly, much more time efficient and enable automatic tuning of SMT system parameters (e.g. using MERT), which may require a number of iterations to converge.

Statistical post-editing of the output of RBMT and SMT systems is an active field of research and RBMT + SPE pipelines are by now a commercial reality. [1] Automatic post-editing of rule-based machine translation systems (Simard et al., 2007; Terumasa, 2007; Kuhn et al., 2010) has shown (in some cases) spectacular improvements in translation quality measured in terms of automatic evaluation scores. Furthermore, SPE has also been applied to the output of statistical MT (SMT) systems (Oflazer and El-Khalout, 2007; Potet et al., 2011; Béchara et al., 2011; Rubino et al., 2012), albeit with more mixed results. However, to date, despite considerable interest in the area, the comparison between SPE pipelines and pure SMT and RBMT systems is not fully researched.

Previous research can be categorised into roughly two classes: in one approach (Simard et al., 2007; Dugast et al., 2007; Potet et al., 2011), manually corrected (i.e. post-edited) MT output is used as the target side for training the SPE system (i.e. a "mono-lingual" SMT system trained on the output of the first stage MT system as source and the manually corrected first stage MT output as target, and then applied to the output of the first stage MT system on unseen source side input data), while the other approach (Oflazer and El-Khalout, 2007; Béchara et al., 2011; Rubino et al., 2012) simply uses available bi-text training data (such as translation memories (TMs) in industrial applications or more generic SMT training data) and trains the SPE system on the output of the first stage MT system and the target side of the bi-text training data.

In the first approach, the SPE system is effectively trained in such a way as to only fix mistakes committed by the first stage MT system: the difference between the output of the MT system (the source side of the SPE training data) and the target side of the SPE training data is in translation mistakes identified and fixed by human post-editors. By contrast, in the second approach, the SPE system is simply trained on the difference between the MT output and the target side of the training data (which may not necessarily constitute a mistake, but just be an instance of a paraphrase or another valid translation alternative).

One of the central research questions addressed in this paper focuses on the second approach: do the (often spectacular) differences in automatic evaluation scores between RBMT and RBMT + SPE pipelines correspond to to "real" improvements in translation quality, as determined by human evaluators, or are they largely due to SPE moving RBMT output closer to a reference string used in automatic evaluation and rewarded by the automatic metrics?

Note that it is reasonable to assume that in the first approach improvements in automatic evaluation scores do indeed correspond to "real" improvements in translation quality, as determined by human evaluators, as in this case the SPE system is solely trained on genuine translation mistakes or (more generally) translation shortcomings.

At the same time, the second approach arguably addresses an important commercial reality in that in many industrial applications it is often the case that specialised TMs exist that have been developed (over many years) to support translation automation and that can now be used

[1] http://www.systran.co.uk/translation- products/server/systran-enterprise-server

as training data for SMT systems for translating input that yields only low fuzzy matches in the TMs. By contrast, the first approach involves a long term commitment to correct the first stage MT output to collect sizeable training data sufficient to train a successful SPE component, which is not always possible given deadlines and the variety of data that needs to be translated in commercial applications.

A second, related, research question addressed by the paper is to determine how RBMT + SPE pipelines compare with a pure SMT system (trained on the same data) or corresponding SMT + SPE systems (again trained on the same data), and how are these rated by human evaluators? Again we focus on this research question in the context of the second approach to SPE based on independently available bi-text training data such as TMs.

As automatic evaluation metrics are inherently biased by the reference translation, it is difficult to tell if these SPE scores correspond to an actual improvement in translation quality. In this study we use human evaluation to answer that question, and to further discover whether the statistical machine translation system outperforms the SPE combination systems, or whether the bias of the automatic evaluation metric is getting in the way of choosing the better system.

The remainder of this paper is organised as follows. In Section 2, we review related research and how it ties into our experiments. In Section 3 we detail the machine translation systems used in both the first and post-editing stages of our experiments, along with the automatic evaluation results of these systems and their combinations. Section 4 presents the human evaluation results, and Section 5 provides an error analysis that helps to answer some of the questions posed in this paper.

2 Related Work

Statistical post-editing has been applied to different types of MT systems to varying degrees of success. The main idea behind SPE for MT is to capture the mistakes made by the MT system and to automatically correct them. (Allen and Hogan, 2000) conducted early studies on the subject (without actually building a SPE system) by using a parallel corpus composed of three tiers: the source text, its automatic translation and the manually post-edited (i.e. corrected) automatic translation. This study inspired the original work on SPE by (Simard et al., 2007), who used the Portage System (a PBSMT system) to automatically post-edit the output of an RBMT system, using the raw RBMT output and the manually post-edited (i.e. corrected) output as "source" and "target" side, respectively, of the SPE training data.

SPE is generally mono-lingual, operating on the target side of the translation direction. In a sense, it can be viewed as a re-writing step on MT output a posteriori, usually unpacked as a supervised learning problem. In one approach, such as that of (Simard et al., 2007), SPE directly negotiates between specific errors in the RBMT (or generally first stage MT) output and the corresponding manual corrections (where mistakes in first stage MT output are corrected by human translators), in the other approach independent bitext data (such as TMs) are used to train SPE systems on the output of the first stage RBMT (or more generaly any MT) system on the sorce side of the bitext data and the corresponding target side of the bitext data. We present both approaches in the two subsections below:

2.1 SPE with Manually Post-Edited MT Output

Amongst the published work on SPE with manually post-edited MT output, several studies have been conducted by combining RBMT and PBSMT as the MT and the SPE systems, respectively.

(Simard et al., 2007) show that a commercial RBMT system combined with the PBSMT system Portage (Sadat et al., 2005) in an SPE pipeline achieve improved translation quality. On a translation task from French to English, using SPE shows an improvement of 13.7% BLEU (Papineni et al., 2002) (absolute) over the RBMT system alone. The authors also conduct experiments combining two PBSMT systems, both in the translation and the post-editing phase, and show that this approach leads to higher BLEU scores if the training corpora for the translation and the post-edition systems are different.

(Dugast et al., 2007) carry out a qualitative analysis at the linguistic level, conducted on the MT and the SPE outputs. They combine Systran with the PBSMT systems Moses (Koehn et al., 2007) and Portage. The output of the combined systems (Systran+SPE) shows significant improvements in terms of lexical choice. They also report gains in terms of BLEU scores up to 10 points absolute on a German to English translation task, compared to the RBMT system individually.

More recently, (Potet et al., 2011) combine a full PBSMT pipeline (SMT+SMT) for translation and post-editing from French to English. The first system translates the French text into English. The MT output is then manually post-edited and introduced into the pipeline following three approaches:

- as supplementary material to enrich the training corpus used to build the translation model,
- as the target side of the parallel corpus used to build the post-editing model,
- as a the target side of the development corpus used to optimize the translation model components weights.

This preliminary study shows a slight improvement over a standalone MT system, but further experiments on larger corpora are needed in order to obtain significant results.

2.2 SPE with Independent Bitext Data

Due to the time and expense involved in manually post-editing MT output, using independently available parallel training data (such as TMs) in SPE pipelines is often a less expensive (and arguably commercially more frequent) scenario. (Terumasa, 2007) combines RBMT with SPE to translate patent texts, and reports an improved score the NIST evaluation compared to that of RBMT alone.

In (Kuhn et al., 2010), the authors compare the two SPE approaches: the first using manually post-edited MT output and the second using the target side of the bilingual training data. They use Systran RBMT and Portage PBSMT systems, and combine them into a post-editing pipeline, with the RBMT system as first stage and the PBSMT system as the SPE system. The SPE system shows a gain of 10.2 BLEU points compared to the RBMT system alone, on a French-to-English translation task. However, the authors also show that a PBSMT system alone can reach results similar to those obtained by the post-editing pipeline.

The first mention of a pure SMT-based SPE pipeline (SMT + SPE) is likely to be in (Oflazer and El-Khalout, 2007), who in one of the experiments as part of their work on selective segmentation based models for English to Turkish translations, employ statistical post-editing (which they call model iteration). In the study conducted by (Béchara et al., 2011), two PBSMT systems are combined into a post-editing architecture. Two sets of experiments are presented. The first is a naive post-editing approach, were the output of the first SMT system is post-edited by the SPE

system without introducing additional information. The second is a source-context enriched SPE approach. Following the full PBSMT post-editing pipeline approach, (Rubino et al., 2012) use statistical post-editing to adapt out-of-domain machine translation systems to a specific domain and show that a generic MT system can be adapted through an automatic post-editing step.

In the first approach, SPE with manually post-edited MT output, the SPE directly addresses translation mistakes or inadequacies caused by the first stage MT system. By contrast, in the second approach, SPE with independently available bitext data, it is not guaranteed that a divergence between first stage MT output and the target side of the bilingual training data actually corresponds to a translation mistake by the first stage MT system. Our research focuses on this second scenario, in particular on whether (sometimes spectacular) SPE improvements in automatic evaluation scores correspond to actual improvements in translation quality verified by human evaluators. Our experiments follow the general design put forth by (Dugast et al., 2007) and (Kuhn et al., 2010), where the output of a Systran RBMT system on the source side of some bitext data as well as the target side of this bitext data is used to train a monolingual second-stage SPE system to post-edit the output of the RBMT system. The objective is to compare the output of the combined RBMT+SPE system with the RBMT system on its own, with an SMT system on its own trained on the bilingual training data, as well as with an SMT+SPE pipeline. We investigate whether the RBMT+SPE pipeline is actually better than Systran (or any of its other rivals: SMT and SMT+SPE), or just closer to the reference translation. In order to do this, we enlist the help of human evaluators.

3 Machine Translation Systems

In this section, we present the data and the translation systems used in our experiments. We also give details about the two SPE pipelines and the automatic evaluation metrics used to measure system performance.

3.1 Translation Memory Bitext Data

The data for our experiments are part of a French-English translation memory provided by Symantec. The data itself is in the domain of technical software user help information. We preprocessed the translation memory and removed all TMX markup and meta-information. We extracted 53,000 unique sentences from the translation memory, and from this data we randomly select 50,000 French-English sentence pairs as our training set. The sentences are between 1 and 98 words in length for English, and 1 and 100 words in length for French. The average sentence length in the training set is 13 words for English and 15 words for French, with a vocabulary size of 9,273 for the English side of the data, and 12,070 for French. Given the specific domain of the data, these are only 11 out-of-vocabulary (OOV) in the test set relative to the training data. The remaining sentences were split into a test set of 2000 sentences, and a development set of 1000 sentences. As we are working with a translation memory, all the sentences are unique. That is to say there are no repetitions in the data, and hence no overlap between the training, development, and test sets. Table 1 summarises the statistics of our data.

3.2 Rule-Based Translation

Despite the success of machine learning based and statistical approaches, rule-based machine translation systems still constitute a significant part of the current commercial MT landscape. RBMT systems work off built-in linguistic rules and bilingual dictionaries in order to construct

	French	English
TM Vocabulary Size	12,070	9,273
Training Vocabulary Size	11924	9159
Average Sentence Length	15	13
Range of Sentence Length	[1,100]	[1,98]

Table 1: Vocabulary and sentence length statistics for the French-English Translation Memory

translations for a given language pair. Wide-coverage systems rely on large-scale lexical and morphological, semantic, and syntactic information. Rule-based systems have been found to provide fluent and predictable quality translations.

As our rule-based machine translation system for the first stage MT in our experiments, we used the Systran Enterprise Server 6 production system, specifically customised with the use of 10K+ dictionary entries specific to the text type and domain of the Symantec translation memory data, as described in (Roturier, 2009).

3.3 Statistical Phrase-Based Machine Translation

Statistical machine translation builds statistical models based on the analysis of existing parallel corpora, both monolingual and bilingual. For our statistical machine translation system we used the PBSMT system Moses, 5-gram language models with Kneser-Ney smoothing trained with SRILM (Stolcke, 2002), the GIZA++ implementation of IBM word alignment model 4 (Och and Ney, 2003), with refinement and phrase-extraction heuristics as described in (Koehn et al., 2003). We used minimum error rate training (MERT) (Och, 2003) for tuning on the development set. During decoding, the stack size was limited to 500 hypotheses.

3.4 Statistical Post-editing

The first pipeline, which combines RBMT output with SPE (statistical post-editing) system, uses Systran to translate the entire source side of the TM-based training set, and the output together with the corresponding target side of the TM is then used as the training data for the SMT-based SPE system. The second-stage system therefore produces a monolingual translation based on the output produced by the first stage RBMT system and the target side of the TM training data (Figure 1).

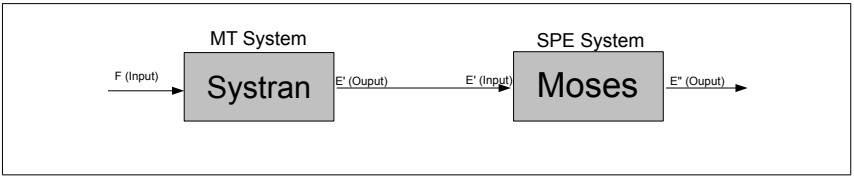


Figure 1: The RBMT+Moses pipeline, using the output of Systran as the input for the second stage SMT system (Moses)

The second pipeline uses SMT in both stages of the post-editing system. The first-stage PBSMT system is trained on the French to English parallel training data, and the output E' (MT output English) will be the source data for the second-stage (SPE) system. Once again the second-stage system produces a monolingual translation, this time using the output of the SMT system as its

input. The source side for the training data for the second-stage system is obtained by training another first-stage PBSMT system from French to English, using a 10-fold cross-validation approach on the French to English training set. This approach will ensure that we do not translate already seen data, and that the source side of the training set for the SPE system is as close in quality to the test set source as possible. Figure 2 illustrates the SMT+SPE pipeline.

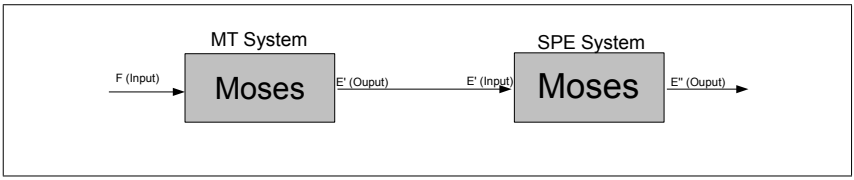


Figure 2: The Moses+Moses pipeline, using the output of Moses as the input for the second stage SMT system (Moses)

3.5 Automatic Evaluation Metrics

We used two metrics for automatic evaluation; BLEU (Bilingual Evaluation Understudy) and TER (Translation Edit Rate) (Papineni et al., 2002; Snover et al., 2006). Both of these metrics depend on a reference translation to estimate the quality of machine translated output. BLEU matches n-grams between the MT output and the reference translation, using n-gram precision with a brevity penalty as the score, as demonstrated in (1)

$$\text{BLEU}(n) = \prod_{1}^{n} PREC_i^{\frac{1}{n}} \cdot bp \tag{1}$$

where n is the order of n-gram, $PREC_i$ is the i-gram precision and bp is the brevity penalty. The brevity penalty is defined as (2):

$$bp = exp(max(\frac{len(Ref)}{len(Out)} - 1, 0)) \tag{2}$$

where $len(Ref)$ is the length of the reference and $len(Out)$ is the length of the output.
This n-gram matching scheme makes BLEU very sensitive to small changes in the output, and fails to capture linguistic variations, especially in the case where only one reference translation is being used. (Callison-Burch et al., 2008) show that that BLEU has a lower correlation with human judgement than metrics such as TER, which take into account linguistic resources and better matching strategies. Furthermore, BLEU is designed to evaluate MT output on a document level. For this reason, we have used S-BLEU (Sentence-Level BLEU) and TER to compare individual sentences.

TER is an Edit Distance-style evaluation metric that measures the amount of editing that a human post-editor would have to perform to change a system output so it matches the given reference translation. It calculates how many insertions, deletions, substitutions and sequence shifts are required to make the output identical to the reference. TER is defined in equation(3):

$$\text{TER} = \frac{\#INS + \#DEL + \#MOD + \#SHIFT}{len(Ref)} \tag{3}$$

	RBMT	SMT	RBMT+SPE	SMT+SPE
BLEU	23.26	65.43	64.63	65.14
TER	61.07	23.92	24.62	24.12

Table 2: BLEU and TER scores for the RBMT, SMT and the SPE systems

System	Ins	Del	Sub	Shift	TER
SMT	5.1	5.05	10.5	3.5	23.92
RBMT	17.04	4.39	30.24	9.3	61.07
SMT+SPE	5.47	4.95	10.1	3.56	24.61
RBMT+SPE	5.2	5.5	10.5	3.27	24.11

Table 3: Normalised number of translation errors for the RBMT, SMT, and SPE systems according to TER edit statistics

3.6 Automatic Evaluation Results

In order to evaluate the RBMT, SMT, RBMT+SPE and SMT+SPE approaches, we train and tune the SMT system on the French-to-English training and development sets, and decode the 2,000 sentence test set using this first-stage system. We also translate the complete training data using a 10-fold cross training approach (tuning on the same development set as above) to avoid translation of seen data to create the source side of the SPE system for the SMT+SPE pipeline. Furthermore, we translate the complete training data (as well as the test set data) using the Systran RBMT system, to create the source side training data for the SPE system in the RBMT+SPE pipeline. The target side of both SPE systems is provided by the target side of the training data from the TM. We evaluate automatically the post-edited translation of the test set using BLEU and TER. Despite the fact that the RBMT system was tuned to the TM data (section 3.2), Table 2 shows that SMT, RBMT+SPE and SMT+SPE outperform RBMT by more than 40 BLEU points absolute on our TM data set (with similar improvements for TER).[2] While the RBMT+SPE system improves over the RBMT output, it fails to improve over the pure SMT output, and performs similar to the SMT+SPE output in terms of the automatic evaluation scores.

Table 3.6 presents the number of average edits per sentence, based on the TER edit types. The errors are divided into four categories: insertion (Ins), substitution (Sub), deletion (Del) and shift. The numbers have been normalised using sentence length to make them comparable. The table suggests that applying SPE to the RBMT system achieves significant gains in the insertion and substitution categories, and to a lesser extent to the shift category. This reflects the fact that the SPE system can improve the pure RBMT translation in terms of better lexical choice and better reordering. Furthermore, the large number of substitutions and insertions in the RBMT system shows that the majority of the errors that account for the lower quality of the RBMT system are lexical. The number of deletions remains largely unaffected by the post-editing system, indicating that little information is actually lost during the second stage. Neither the RBMT+SPE nor the SMT+SPE systems achieve any significant gains over the pure SMT system.

4 Human Evaluation

In order to further investigate the quality of the RBMT+SPE output compared to the pure RBMT, pure SMT as well as the SMT+SPE output, we complement automatic evaluation metrics with a study involving human evaluators. Human evaluation can be an important source of

[2]Note again that there is no duplication between the test and training data extracted from the TM.

information, providing insight as to whether or not (and why) the SMT system actually performs better or worse than the RBMT+SPE pipeline, how these compare with the SMT+SPE output, and whether, and if so to what extent, the spectacular differences in automatic scores between the RBMT and the SMT and SPE pipeline systems actually correspond to human judgements.

4.1 The Evaluation Task

Evaluation was carried out by ten different translators of varied backgrounds. While none of them are professional translators, all of them have experience with machine translation or localisation. Six of these evaluators are native French speakers, and the others have a good grasp of French, evidenced by school and professional certificates. All of the evaluators speak English fluently.

The evaluators were asked to evaluate a pair of sentences from two of the four MT systems: pure SMT, pure RBMT and the two SPE pipelines RBMT+SPE and SMT+SPE. The evaluators were shown a source sentence (in French) and asked which of the two MT outputs (presented in random order) is a better translation, or if they are of equal quality. In order avoid biasing the evaluator, we did not provide a reference translation. The task spanned 200 sentences, and was available to be completed online. The subjects were paid for their time and were given a week to submit the task, which did not have to be completed in one sitting. Figure 3 shows a screenshot of the user interface with an example sentence included in the task. The evaluators generally rated the task as difficult, especially as the domain was highly technical and the sentences often fragmented and containing a large number of symbols and abbreviations.

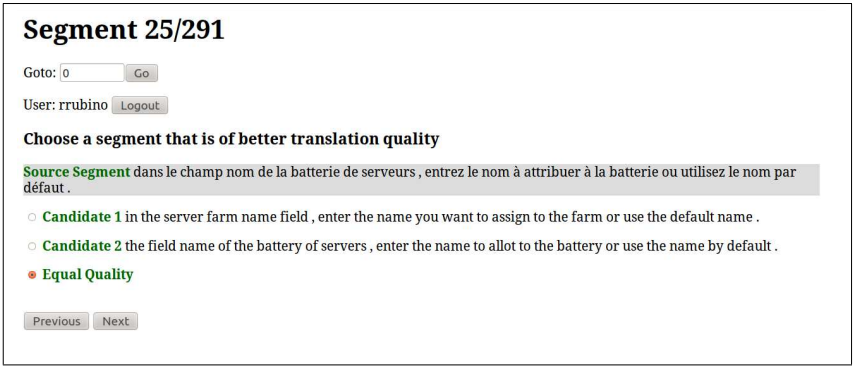


Figure 3: A screen-shot of the manual evaluation task

4.2 Annotator Agreement

Since a reasonable degree of agreement must exist to support the validity of our human evaluation experiment, we calculated pair-wise inter-annotator agreement between all of the different evaluators. For this agreement, we used Cohen's κ measure. κ is a more robust measure compared to simple percent agreement calculation, as it takes into account the

agreement occurring by chance. κ is defined by the formula in 4.

$$\kappa = \frac{Pr(a) - Pr(e)}{1 - Pr(e)} \tag{4}$$

Pr(a) is the proportion of times two annotators are observed to agree, and *Pr(e)* is the expected proportion of times the two annotators are expected to agree by chance. Agreement occurs when two annotators compare the same systems and agree on their rankings. In our case there are three possible choices; either one system is better than the other, or it is worse, or there is a tie. κ ranges between 0 and 1, with 1 indicating a higher rate of agreement, and 0 indicating low or no agreement.

According to (Landis and Koch, 1977), a moderate agreement falls between 0.4 and 0.6. A substantial agreement falls between 0.6 and 0.8, and 0.2 to 0.4 indicate a fair agreement, while anything below that is considered slight. Full results for all ten evaluators ($\kappa = 0.42$) are on the border between moderate to fair. As two of our evaluators scored an average agreement under 0.4, we discarded their results as weak. Without the outliers, our average agreement for the evaluators is 0.47. This amounts to low moderate agreement.

4.3 Human Evaluation Results

We tallied up the number of times that each system was chosen as "best system" based on the 200 sentences that were evaluated. The human annotators' results were normalised and divided by the number of annotators (in our case 8, as we discarded the outliers). In order to evaluate the document on a sentence-level, we used S-BLEU instead of BLEU. S-BLEU will still positively score segments that do not have highern-gram (n=4 in our setting) matching, unless there is no unigram match.

We compared the S-BLEU and TER scores for the sentences of each of the outputs, and tallied up the number of times each system was given the best score by either S-BLEU or TER. We then compared the results from the human evaluators with the S-BLEU and TER results. We found that while the automatic metrics seem to favour a pure SMT system, the human annotators were leaning more towards the combined rule based and post-editing system. Furthermore, while automatic evaluation metrics chose Systran as the best system less than 7% of the time, human evaluators chose it as the best system more than twice as often as S-BLEU or TER did. The results are further detailed in Table 4.

In order to assess the difficulty to compare different systems, we recorded the time each evaluator spent evaluating a translation pair. We assume that spending more time on an evaluation indicates that it is more difficult to select the best translation amongst the two alternatives. We report averaged results in Table 5. The results show that comparing the two stand-alone MT systems (RBMT and SMT) is hard because on average more than 20 seconds are spend to select the best translation. This is most likely due to the profound differences in terms of syntax and vocabulary between the SMT and the RBMT outputs. By contrast, when comparing SMT versus SMT+SPE, the time spent drops by nearly 10 seconds (on average). This is most likely due to the fact that SMT and SMT+SPE outputs are very similar, requiring less time to scan and judge. A similar trend can be observed comparing SMT with RBMT+SPE, where again outputs are more similar than between SMT and RBMT on their own. Finally, choosing between RBMT and RBMT+SPE requires the least amount of time. This is consistent with the observation that (according to the human evaluation) the quality difference between

	Human Evaluation	S-BLEU	TER
	SMT vs RBMT		
SMT	97	162	161
RBMT	52	16	9
Tie	51	26	30
	SMT vs RBMT+SPE		
SMT	28	125	124
RBMT+SPE	40	50	46
Tie	132	25	30
	SMT vs RBMT+SPE		
RBMT	40	16	11
RBMT+SPE	99	162	162
Tie	61	22	26
	SMT+SPE vs RBMT		
SMT+SPE	107	167	161
RBMT	46	11	9
Tie	47	25	30
	SMT+SPE vs RBMT+SPE		
SMT+SPE	27	46	46
RBMT+SPE	47	49	41
Tie	126	105	113

Table 4: Number of sentences chosen as "best" by each of the evaluations

RBMT and RBMT+SPE is the most pronounced, and therefore more "obvious" than in the other cases.

	SMT vs RBMT	SMT vs SMT+SPE	SMT vs RBMT+SPE	RBMT vs RBMT+SPE
Average time	23.4	14.12	12.61	9.52

Table 5: Average time spent (in seconds) by human evaluators on each system comparison.

5 Error Analysis

In order to obtain a better understanding of the translation quality gains between the RBMT system and the RBMT+SPE system, and to gain insight into why there are discrepancies between the manual and automatic evaluation results, we performed an additional manual sentence-level error analysis in a bid to reveal the advantages and disadvantages of the SPE pipelines compared with the RBMT and SMT systems.

Table 6 shows the detailed number of error types by system, based on the error typography provided by (Vilar et al., 2006). Our error analysis confirms what the TER edit statistics in Table 3.6 suggest, that most of the errors that account for the considerably lower quality of the RBMT system are lexical, both in terms of simple lexical choice and the repercussions of this on the phrasal level. Even though the RBMT system was tuned to the domain of the TM via domain specific lexical resources (Section 3.2), most of the errors appear to be due to the RBMT system's inability to pick the right term for the technical domain data set. However, compared to SMT and SMT+SPE, both the RBMT and RBMT+SMT system seem to produce a significantly lower number of grammatical errors, according to our evaluators. This is mostly obvious in the determiner and preposition categories, where combined, the SMT system produces three times as many errors as the RBMT system. Our results also show that while the SPE considerably

	RBMT	SMT	RBMT+SPE	SMT+SPE
Not Found Words	1.5	5	0	5
Simple Terms	34.5	10.5	6	9.5
Phrases	20.5	2.5	2	3
Meaning	20.5	2.5	2	3
Determiners	1	4.5	2	2.5
Prepositions	3	8.5	2.5	6
Tense	1.5	2.5	2.5	3
Number	0	1	1	1
Other Grammar	2.5	6.5	3.5	5.5
Punctuation	1	3.5	3.5	3.5
Word Order	7	4	4	4.5

Table 6: Normalised number and types of errors found in manual evaluation results

changes the error typography when applied to RBMT, reducing the overall number of errors, it has has a much smaller effect when applied to the SMT system. SMT+SPE fails to improve on a lexical choice where SMT has failed, and only marginally improved grammatical errors. Example 1 shows a very common RBMT lexical choice error. Errors such as these are almost always corrected in the statistical post-editing (SPE) phase.

Example 1

Source	options de planification de modification d'a pour **un travail de sauvegarde**
RBMT	options of planning of modification of has for **a work of backup**
RBMT+SPE	schedule options to change for **a backup job**
SMT	scheduling options has changed for **a backup job**
SMT+SPE	scheduling options has changed for **a backup job**
Reference	to change schedule options for for **a backup job**

Example 2 shows a similar case where the RBMT+SPE pipleline is superior when it comes to picking the right phrases within the correct domain. Due to the highly technical nature of the translation memory, the intended meaning is often lost if the wrong lexical choices are made.

Example 2

Source	pour installer **une version d évaluation**
RBMT	to install **a version of rating**
RBMT+SPE	to install **a trial version**
SMT	to install **a trial version**
SMT+SPE	to install **a trial version**
Reference	to install **an evaluation version**

On the other hand, RBMT often performs better when it comes to general grammar, especially in terms of prepositions and, to a lesser extent, determiners. This carries over to the RBMT+SPE system, which leads to a better grammatical quality than the pure SMT system (or the SMT+SPE pipeline, for that matter). Example 3 shows a common case where the preposition is missing from the Moses translation, but is inserted correctly in both the RBMT and RBMT+SPE translations.

Example 3

Source	pour ajouter le **le nom de compte** de connexion
RBMT	to add the name **of account** of login
RBMT+SPE	to add the name **of logon account**
SMT	to add the name **logon account**
SMT+SPE	to add the name **logon account**
Reference	to add the **logon account** name

Another interesting aspect concerns out-of-vocabulary (OOV) words. RBMT seems to be better at finding words than SMT (this is probably a reflection of the fact that the RBMT system used in our experiments was a production system tuned with a domain-specific 10k+ dictionary to the TM-based data-set), and even though these are not always perfectly correct words, they are sometimes fixed in post-editing, as seen in Example 4. As a result, RBMT and RBMT+SPE produce few if any out of vocabulary items given the output.

Example 4

Source	**enregistrera** l'image .iso idr amorçable ou non amorçable
RBMT	will **record** the or not bootable image .iso idr bootable
RBMT+SPE	will **save** the idr bootable or non-bootable .iso image
SMT	**enregistrera** the idr bootable or non-bootable .iso image
SMT+SPE	**enregistrera** the idr bootable or non-bootable .iso image
Reference	will **save** the bootable or non-bootable idr .iso image .

The results also show that in a few cases SMT+SPE can produce some grammatical improvements over the pure SMT system as well. Example 5 is one such case, where SPE applied to SMT corrected a grammatical error (the missing preposition **for**).

Example 5

Source	le nombre de secondes **pendant lesquelles le processus de restauration** ...
RBMT	the number of seconds **during which the process of restoration** ...
RBMT+SPE	the number of seconds **for the restore process** ...
SMT	the number of seconds **the restore process** ...
SMT+SPE	the number of seconds **for the restore process** ...
Reference	the number of seconds **for the restore process** ...

Conclusion and Perspectives

Previous research on automatic post-editing has shown spectacular improvements in automatic MT evaluation scores of RBMT + SPE pipelines over RBMT systems. In this paper we set out to answer two open questions for an SPE scenario that uses independently existing bitext training data, rather than specifically and manually corrected first stage MT output, as its training data:

- Is statistical post-editing (SPE) improving the output of the first-stage RBMT system, or is it just pushing the output closer to the reference translation?
- Does the RBMT+SPE pipeline improve the quality of the output over that of the pure SMT system as well as that of an SMT+SPE system trained on the same data set?

In order to answer these questions we used human evaluation from annotators who were unbiased by the reference translation. Our human evaluators agreed with the automatic evaluation metrics that the RBMT + SPE system does indeed perform better than the RBMT system on its own. Additionally, while they did not find the improvement as pronounced as the automatic evaluation metrics indicate, they consistently rated the RBMT + SPE system higher than the RBMT system by a factor of two.

While automatic evaluation metrics indicate that SPE does not improve RBMT systems over pure SMT system, a manual evaluation shows that human evaluators prefer the RBMT + SPE output over the pure SMT output. We conclude that this discrepancy is a result of S-BLEU and TER being biased towards the SMT system. Error analysis shows that SPE makes better lexical and phrasal choices, which leads to superior translation quality. We also observed that the human evaluators spend more time to select the best translation when the MT systems are very different (SMT and RBMT for instance), which reflects the underlying difficulty of the evaluation task.

We would like to use what we learnt about the nature of the errors and the strengths and weaknesses of the post-editing system to automatically predict which sentences can be corrected using SPE. Many sentences do not benefit from the post-editing phase, and a subset of sentences even degrade after post-editing. Using machine learning techniques, we plan to classify the output of the RBMT system, based on a variety of features, into two categories: those that benefit from SPE, and those that do not. Furthermore, we plan to train a classifier to choose better sentences of the two systems, RBMT+SPE and pure SMT, in order to reach the upper bound given by the two systems together. Furthermore, results obtained from the error analysis should enhance feature selection methods.

We will research ways to further refine statistical post-editing techniques for both RBMT and SMT systems. In previous work, we developed a contextualised SPE system that attempts to preserve the original context of the source material with a novel method of context modelling (Béchara et al., 2011). We intend to take this work further and refine our use of context information. We will also experiment with different system combinations: in addition to RBMT and PBSMT systems, we will utilise a hierarchical phrase based SMT system (Chiang, 2005).

Acknowledgments

This work is supported by Science Foundation Ireland (Grant No. 07/CE/I1142) as part of the Centre for Next Generation Localisation (www.cngl.ie) at Dublin City University. We would like to thank Symantec, in particular Dr. Johann Roturier and Dr. Fred Hollowood, for providing us with the data and with access to the Symantec Systran Machine Translation Production System. We would like to thank Dr. Stephen Doherty, from the School of Applied Language and Intercultural Studies at Dublin City University, for providing us with access to masters in translation students to act as our evaluators. We would also like to thank the reviewers for their many insightful comments, suggestions, and corrections.

References

Allen, J. and Hogan, C. (2000). Toward the Development of a Post editing Module for Raw Machine Translation Output: A Controlled Language Perspective. In *Proceedings of the Workshop on Controlled Language Applications (CLAW)*, pages 62–71.

Béchara, H., Ma, Y., and van Genabith, J. (2011). Statistical Post-Editing for a Statistical MT System. In *Proceedings of the MT Summit XIII*, pages 308–315.

Callison-Burch, C., Fordyce, C., Koehn, P., Monz, C., and Schroeder, J. (2008). Further Meta-Evaluation of Machine Translation. In *Proceedings of the Third Workshop on Statistical Machine Translation (WMT)*, pages 70–106.

Chiang, D. (2005). A Hierarchical Phrase-based Model for Statistical Machine Translation. In *ACL '05 Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*.

Dugast, L., Senellart, J., and Koehn, P. (2007). Statistical Post-Editing of SYSTRAN's Rule-Based Translation System. In *Proceedings of the Second Workshop on Statistical Machine Translation (WMT)*, pages 220–223.

Koehn, P., Hoang, H., Birch, A., Callison-Burch, C., Federico, M., Bertoldi, N., Cowan, B., Shen, W., Moran, C., Zens, R., Dyer, C., Bojar, O., Constantin, A., and Herbst, E. (2007). Moses: Open Source Toolkit for Statistical Machine Translation. In *Proceedings of the Association for Computational Linguistics (ACL)*, pages 177–180.

Koehn, P., Och, F., and Marcu, D. (2003). Statistical Phrase-Based Translation. In *Proceedings of the Human Language Technology Conference and the North American Chapter of the Association for Computational Linguistics (HLT/NAACL)*, pages 48–54.

Kuhn, R., Isabelle, P., Goutte, C., Senellart, J., Simard, M., and Ueffing, N. (2010). Automatic Post-Editing. *Multilingual*, 21(1):43–46.

Landis, J. and Koch, G. (1977). The measurement of observer agreement for categorical data. *Biometrics*, pages 159–174.

Och, F. (2003). Minimum Error Rate Training in Statistical Machine Translation. In *Proceedings of the Association for Computational Linguistics (ACL)*, pages 160–167.

Och, F. and Ney, H. (2003). A Systematic Comparison of Various Statistical Alignment Models. In *Proceedings of the Association for Computer Linguistics (ACL)*, pages 29(1):19–51.

Oflazer, K. and El-Khalout, I. D. (2007). Exploring Different Representational Units in English-to-Turkish Statistical Machine Translation. In *Proceedings of the Second Workshop on Statistical Machine Translation (WMT)*, pages 25–32.

Papineni, K., Roukos, S., Ward, T., and Zhu, W. (2002). BLEU: a Method for Automatic Evaluation of Machine Translation. In *Proceedings of the Association for Computational Linguistics (ACL)*, pages 311–318.

Potet, M., Esperança-Rodier, E., Blanchon, H., and Besacier, L. (2011). Preliminary Experiments on Using Users' Post-Editions to Enhance a SMT System. In *Proceedings of the European Association for Machine Translation (EAMT)*, pages 161–168.

Roturier, J. (2009). Deploying Novel MT Technology to Raise the Bar for Quality: A Review of Key Advantages and Challenges. In *Proceedings of the MT Summit XII*.

Rubino, R., Huet, S., Lefèvre, F., and Lenarés, G. (2012). Statistical Post-Editing of Machine Translation for Domain Adaptation. In *Proceedings of the European Association for Machine Translation (EAMT)*, pages 221–228.

Sadat, F., Johnson, J., Agbago, A., Foster, G., Kuhn, R., Martin, J., and Tikuisis, A. (2005). PORTAGE: A Phrase-Based Machine Translation System. In *Proceedings of the Workshop on Building and Using Parallel Texts: Data-Driven Machine Translation and Beyond*.

Simard, M., Goutte, C., and Isabelle, P. (2007). Statistical Phrase-based Post-editing. In *Proceedings of the Human Language Technology Conference and the North American Chapter of the Association for Computational Linguistics (HLT/NAACL)*, pages 508–515.

Snover, M., Dorr, B., Schwartz, R., Micciulla, L., and J., M. (2006). A Study of Translation Edit Rate with Targeted Human Annotation. In *Proceedings of Association for Machine Translation in the Americas (AMTA)*, pages 223–231.

Stolcke, A. (2002). SRILM - an Extensible Language Modeling Toolkit. In *Proceedings of the International Conference on Spoken Language Processing (ICSLP)*, pages 901–904.

Terumasa, E. (2007). Rule Based Machine Translation Combined with Statistical Post Editor for Japanese to English Patent Translation. In *Proceedings of the MT Summit XI Workshop on Patent Translation*, volume 11, pages 13–18.

Vilar, D., Xu, J., D'Haro, L. F., and Ney, H. (2006). Error Analysis of Statistical Machine Translation Output. In *Proceedings of the Fifth International Conference on Language Resources and Evaluation (LREC)*, pages 697–702.

Prague Dependency Treebank 2.5 – a revisited version of PDT 2.0

Eduard BEJČEK
Jarmila PANEVOVÁ Jan POPELKA
Pavel STRAŇÁK Magda ŠEVČÍKOVÁ
Jan ŠTĚPÁNEK Zdeněk ŽABOKRTSKÝ
Charles University in Prague, Faculty of Mathematics and Physics,
Institute of Formal and Applied Linguistics, Malostranské náměstí 25, Praha, Czech Republic
{bejcek,panevova,popelka,stranak,sevcikova,stepanek,zabokrtsky}@ufal.mff.cuni.cz

ABSTRACT
We present the Prague Dependency Treebank 2.5, the newest version of PDT and the first to be released under a free license. We show the benefits of PDT 2.5 in comparison to other state-of-the-art treebanks. We present the new features of the 2.5 release, how they were obtained and how reliably they are annotated. We also show how they can be used in queries and how they are visualised with tools released alongside the treebank.

TITLE AND ABSTRACT IN CZECH

Pražský závislostní korpus 2.5 – rozšířená verze PDT 2.0

Představujeme nejnovější verzi Pražského závislostního treebanku PDT 2.5, který bude poprvé vydán pod veřejnou licencí. Výhody PDT 2.5 ukážeme na srovnání s nejmodernějšími treebanky. Představíme nové vlastnosti verze 2.5, popíšeme, jak byly anotovány i jak spolehlivá tato anotace je. Ukážeme, jakými dotazy lze nové jevy hledat a jak se zobrazují v nástrojích dodávaných spolu s treebankem.

KEYWORDS: treebank, linguistic theory, PDT, annotation, syntax, semantics, multiword expressions, pair/group meaning, clauses.

CZECH KEYWORDS: treebank, lingvistická teorie, PDT, anotace, syntax, sémantika, víceslovné výrazy, souborovost, klauze.

Proceedings of COLING 2012: Technical Papers, pages 231–246,
COLING 2012, Mumbai, December 2012.

1 Introduction

The task of grammatical theories is to explicitly describe phenomena of language with the purpose of creating a description that analyses and/or generates language as natural as possible. The creation of a treebank then serves as an ultimate test of a linguistic theory.

Treebanks that have not been based on an elaborate theory which takes into account most phenomena of natural language start with a simple design. If they become popular, various additional, more-or-less ad hoc linked projects end up being piled upon the original simple design.

On the other hand, there are very complex theories that have been being developed for decades to take into account all of the possible phenomena of natural language but have not yet undergone the ultimate test of large-scale treebanking (e.g. Mel'čuk and Polguère, 1987).

In this paper we introduce the Prague Dependency Treebank 2.5 (PDT 2.5), the latest instance of a large treebank whose design is based on the Functional Generative Description. It is a step from PDT 2.0 towards PDT 3.0 (coming in 2013 with additional large-scale annotation of discourse, anaphora and more) which brings annotation of three new features finished so far and a number of corrections.

The rest of the paper is organised as follows: we introduce some basic facts about PDT 2.5 and the previous version of the treebank in Section 2. In Section 3, we provide some background on treebanking projects in general and compare PDT 2.5 with other popular treebanks. Next we present the new features of PDT 2.5: annotation of multiword expressions in Section 4, new semantic distinction of pair/group meaning of nouns in Section 5, and identification of clauses in Section 6. We summarise the state of the treebank and its general ecosystem in Section 7.

2 Prague Dependency Treebank 2.5 and the previous version

Functional Generative Description (FGD) is a relatively complex linguistic theory and as such it has provided many fundamental ideas that are directly reflected in the PDT design, e.g. multiple layers of linguistic description and the dependency approach to syntax based on the theory of valency (Sgall et al., 1986; Sgall, 1967). Nonetheless, FGD does not encompass all phenomena either, not even in the language core.

In this paper we focus on PDT 2.5, which is an updated release of PDT 2.0. For this new release, the data of PDT 2.0 have been enriched with annotation of three new phenomena (see Sections 4 to 6). Furthermore, some of the errors in the PDT 2.0 data have been corrected in the new release (they mostly involved morphological tags and lemmas[1] for personal names and abbreviations, yet some of these changes were also reflected on the higher layers). However, the design of the PDT 2.5 annotation as well as the size of the data are identical with PDT 2.0.

PDT 2.0 (Hajič et al., 2006) is a collection of Czech newspaper texts from 1990s with annotation added on four layers: on the word layer (w-layer), the source texts have been tokenized and segmented. The morphological layer (m-layer) provides a lemma and a positional tag for each token (word form or punctuation mark). On the analytical layer

[1] About 10 thousand morphological nodes were fixed.

(a-layer), a sentence is represented as a dependency tree with labelled nodes and edges, which correspond to surface-syntactic relations (such as subject, object etc.); one analytical node corresponds to exactly one morphological token. On the tectogrammatical layer (t-layer), the meaning of the sentence is represented as a dependency tree structure with additional features and constraints.

Tectogrammatical nodes (t-nodes) represent content words (including pronouns and numerals), whereas functional words such as prepositions have no separate node in the tree.[2] All t-nodes are labelled with t-lemmas, dependency relations (functors, such as an actor ACT, addressee ADDR or location specification LOC) and grammatemes (see Section 5). Furthermore, annotation of valency, coreference, and topic-focus articulation are all available in tectogrammatical trees as well (Mikulová et al., 2006).

The PDT data consist of 7,110 manually annotated textual documents containing 115,844 sentences with 1,957,247 tokens (word forms and punctuation marks). All these documents were annotated on the m-layer, 75 % of them were annotated on the a-layer (5,330 documents, 87,913 sentences, 1,503,739 tokens). 59 % of the a-layer data were annotated also on the t-layer (i.e. 45 % of the m-layer data; 3,165 documents, 49,431 sentences, 833,195 tokens).

As we are improving the PDT by providing more explicit and consistent annotation guidelines, we are also improving the theoretical framework of FGD. The same is true when we add analysis of phenomena not tackled by the original theory. The theoretical studies preceded the annotation stage.

2.1 Prague Markup Language

PDT uses the PML format (Pajas and Štěpánek, 2006) based on XML. Each token and node has been assigned a unique identifier; any layer built atop of another uses the identifiers from the lower layer as reference targets, effectively creating inter-layer links (of various types). Each node can be assigned an attribute-value structure, an *attribute* in short, that represents various grammatical categories.

Another advantage of the PML format is the availability of the framework surrounding it. The tools provided include the tree editor `TrEd` (Pajas and Štěpánek, 2008), the query language and engine PML-TQ (Pajas and Štěpánek, 2009, see also Figure 2) and a highly modular NLP system Treex (Popel and Žabokrtský, 2010).

3 Related work

During the past decade, plenty of treebanks have been published. New treebanks keep appearing at least bi-monthly.[3] There are some features, though, that set PDT 2.5 apart from most of them.

The most popular treebank of all times is the Penn Treebank (Marcus et al., 1993). It has been since extended by several projects: PropBank (Palmer et al., 2005), NomBank (Meyers et al., 2004), BBN Pronoun Coreference and Entity Type Corpus (Weischedel and Brunstein, 2005), and a few more.

[2] There are several exceptions of a technical nature. For instance, counterparts of coordinating conjunctions are included in the tree because they are used for the representation of coordinating constructions.

[3] LDC has published 5 new treebanks so far in 2012: `http://www.ldc.upenn.edu/Catalog/ByYear.jsp`

The pitfall of this process can be demonstrated on the Chinese Treebank (Xue et al., 2010), whose development followed the Penn Treebank pattern. The additional layers of annotation follow the stand-off principle, linking them to the original data. What remains problematic, though, is the format of these links: for example, both the proposition and word sense annotations use a "token number" to refer to a particular token in a sentence, but the latter only counts terminals with a surface form, while the former includes various added nodes as well (e.g. traces, pro's). Similarly, the coreference annotation (and named entity annotation, too) operates directly on the text, not taking the underlying phrase structure into account. Therefore, units that enter the coreference relations sometimes do not correspond to a continuous subtree of a tree.

As a result, it is substantially non-trivial to search for phenomena that involve several such layers (e.g. "list all the verbs at which a given named entity or a pronoun corefering to it can appear as Arg0").

The PML format (see Section 2.1), on the other hand, results in unambiguous interconnection of the annotation layers.

There are other projects aiming at standardisation of the solutions and conversion of old formats to new ones, cf. (Ide and Suderman, 2009). The solution used in the PDT is comparable to these efforts and standards (such as the LAF or TEI and its variants), but it has the added advantage of being supported by a complete suite of tools for annotation, search and processing mentioned earlier.

Finally, not all treebanks are freely available. Various license restrictions (and usage fees) exist. PDT 2.5 is now being distributed under the standard Creative Commons license (3.0-BY-NC-SA) allowing free access and distribution of additions and modifications.

4 Multiword expressions

Multiword expressions (MWEs) such as idioms, phrasemes, and multiword named entities are an important and sometimes overlooked part of natural languages. Usually they form a significant portion of the vocabulary, particularly in special domains where terminology is in play, but not only there: 16.3% of content words in the PDT are part of a MWE.

Multiword expressions are a boundary phenomenon on the interface of grammar and lexicon. We understand them, in accordance with Sag et al. (2002), Baldwin et al. (2003), Pecina (2009), and other authors, as phrases that contain some *idiosyncratic element* that differentiates them from normal expressions. This idiosyncratic element can be morphological, syntactic, or semantic. Although the annotation belongs to semantic layer, we have means for annotation on the other layers as well.

As a practical guideline for how idiosyncratic the expression must be to constitute a MWE, the most important criteria are the absence of *compositionality of meaning* and *word-for--word translation*. Neither of these criteria is absolute either by itself or together, but they are strong indicators, nonetheless. If these or any other secondary criteria compel a native speaker to add the expression to a dictionary because it requires an explanation, we consider it a MWE. For examples, see page 6.

Although some grammatical theories have accounted for MWEs decades ago (see e.g. Mel'čuk and Polguère, 1987), the annotation of MWEs is one of the least developed phenomena in treebanks. There were some MWEs annotated in PDT 2.0 (such as per-

sonal names or foreign phrases) and there are other treebanks that include named entities and/or MWEs to some extent, e.g. the German TüBa-D/Z or the Bulgarian BulTreeBank.[4]

In PDT 2.5, we annotate all occurrences of MWEs, including named entities (see below). We do not inspect various linguistic subtypes of MWEs in the treebank because we believe it is not the right place to analyse their grammatical attributes—only their instances should be identified in the treebank. Once that is done, a lexicon linked to their occurrences in corpora can be compiled and the MWEs (MWE types) can be analysed, taking into account all the information that can be acquired from the annotated occurrences in the data, as well as other resources. We have compiled a preliminary version of such a lexicon. It is complete with regard to lexemes occurring in the PDT 2.5 data and it is freely available.[5] The elaborate lexicographic analysis of all its entries, however, has yet to be performed. That is why the dictionary is not part of the PDT 2.5 release.

We distinguish a special type of MWEs: **named entities (NE)**.[6] In their case we are interested mainly in their *type* (see the list below) and their *basic form*. Since Czech is an inflectional language, a *basic form* of a MWE often differs from the form used in the sentence, but also from the sequence of basic forms (lemmas) of the individual words. This is illustrated by Examples 1 and 2. In the current release of PDT 2.5, basic forms have been manually added to some types of MWEs (see the full list below): lexemes, persons, locations, objects and some institutions. Treatment of NEs together with other MWEs is important, because syntactic functions are more or less arbitrary inside a NE (consider an address with phone numbers, etc.) and so is the assignment of semantic roles. That is why we need to be able to display each NE as a single node, just like we do it with MWEs in general. See details in Section 4.1.

Tectogrammatical layer is the layer of linguistic meaning, so the MWE annotation belongs there. MWEs can be more easily captured on the t-layer, because: (i) there are added nodes for words not present in the surface sentence (ellipses), (ii) each MWE constitutes a **continuous** subtree on the t-layer; such subtree is consequently collapsible and it can be represented as a single t-node, and (iii) the t-layer also does not feature nodes for auxiliary words,[7] which significantly simplifies the annotation process.

All the multiword expressions in a given sentence are stored in the attribute mwes of the root node of the tectogrammatical tree. The attribute mwes is a list, whose members represent MWEs in the tree. Each MWE contains an ID, a basic-form, a type and a list of identifiers of t-nodes that are a part of the MWE. The type of a MWE can have one of the following values:

- lexeme,[8]
- person (a name of a person or an animal),
- institution,
- object (e.g. a name of a book, a unit of measurement, a biological name of a plant or an animal),
- location,
- address,
- time,
- biblio (a bibliographic entry),
- foreign (a foreign expression),
- number (esp. a numerical range).

[4] See http://arbuckle.sfs.uni-tuebingen.de/en_tuebadz.shtml and (Osenova and Kolkovska, 2002).
[5] http://ufal.mff.cuni.cz/lexemann/mwe/
[6] An easier annotation of *single-word named entities* is left for future versions of PDT.
[7] Auxiliary words are instead accessible through attributes and links.
[8] "Conjunction of the lexical form and the individual meaning" (Filipec, 1994). Compare also "lexical unit"

Examples:

(1) *Prezident Havel by měl 15. července*[*] *na Pražském hradě*[†] *jmenovat třináct soudců Ústavního soudu*[‡] .
transl.: *President Havel is expected to appoint thirteen judges of the Constitutional Court on 15th of July at the Prague Castle.*

[*] *'on 15th of July'* – date, basic-form: "15. červenec" (nominative case)
[†] *'at Prague Castle'* (locative case) – location, basic-form: "Pražský hrad" (nominative case)
[‡] *'[of] Constitutional Court'* (genitive) – institution, basic-form: "Ústavní soud" (nominative)

(2) *Funkce ústavního soudce*[*] *je neslučitelná s členstvím v politických stranách*[†] .
transl.: *The role of a constitutional judge is incompatible with political party membership.*

[*] *'[of a] constitutional judge'* (genitive) – lexeme, basic-form: "ústavní soudce" (nominative)
[†] *'in political parties'* (locative, plural) – lexeme, basic-form: "politická strana" (nominative, singular)

4.1 MWE display and search

There are two modes of viewing the MWEs in TrEd: they can be seen either as coloured groups of t-nodes in a tectogrammatical tree (see Figure 3C), or they can be collapsed into a single node (see Figure 3B). When collapsed, children of the members of a MWE become children of the MWE node itself as we can see with *deficit* and its parent *miliarda* in Figure 3. In the "node group" mode the groups are drawn in different colours representing different types of MWEs. In Figure 3 (B) and (C) there is a subtree (*'a 33 billion budget deficit'*) with 2 MWEs (NE *'33 billion'* and a lexeme *'budget deficit'*)[9] in a compact collapsed form (B) and in a coloured group-view (C). Orange colour represents a multiword number and pink represents a lexeme in (C).

4.2 Annotation procedure

We annotated all occurrences of MWEs (including named entities, see below) at the tectogrammatical layer of PDT. A large part of the data was annotated in parallel. A table below shows how much data was annotated by 1, 2, or 3 annotators in parallel, compared to the size of the t-layer.

	number of annotators			$\sum$ annotated	$\sum$ parallel
	one	two	three	(100% of PDT t-layer)	(in % of PDT t-layer)
t-files	1,288	1,412	465	3,165	59
t-nodes	248,448	343,834	82,683	674,965	63

Table 1: Parallel annotation of data

The data produced by individual annotators is not part of PDT 2.5, but it is freely available at the project web page.[10] For the present release it was used to produce gold standard MWE annotation in the following manner: if the annotators agreed, the MWE was kept as gold. Disagreement was decided as follows:

of Cruse (1986).

[9] Multiword numeric entity is always annotated. The reason for annotation of "budget deficit" (translatable, as you can see) is non-compositionality: it is different than, e.g., "oxygen deficit", because there is no budget shortage, but shortage of money in the budget (or even an income shortage comparing to costs).

[10] http://ufal.mff.cuni.cz/lexemann/mwe/

In case a MWE was recognised by only one annotator, we kept it, since a test had shown that it was much more common for an annotator to miss a MWE than to annotate a false MWE. In case one annotator annotated a subset of the other's MWE, we kept the larger MWE. On the other hand, when one annotator chose several small MWEs covering the other's larger MWE, the smaller ones were kept.[11] The cases when the annotators created intersecting MWEs were judged by a third annotator, as were the cases when one annotator identified several subsets of the other's MWE but the subsets didn't cover the full extent of the large MWE.

5 The grammateme typgroup representing the pair/group meaning

In Czech, nouns typically have two sets of forms according to the grammatical category of number: singular forms and plural forms. Forms of the former set are used to denote a single entity (singularity meaning), plural forms express, in general, more than one entity (plurality meaning). Within the theoretical linguistic framework of FGD as well as in the annotation scenarios of PDT 2.0 and PDT 2.5, the semantic opposition of singularity and plurality is represented by the values sg vs. pl of the grammateme number; grammatemes are attributes of nodes of the tectogrammatical tree, which capture the semantically relevant morphological categories.

In addition to the existence of nouns accompanied in the lexicon with the feature "singulare tantum", which blocks the semantic opposition of sg vs. pl (e.g. *kamení 'stones'*) and "plurale tantum", where sg and pl are expressed by the same form (e.g. dveře 'door/doors'), there are nouns in Czech that have both singular and plural forms but their plural forms are used to refer to a pair or to a typical group of entities rather than to a plurality of them. For instance, the plural *ruce 'arms'* denotes a pair or several pairs of arms rather than several upper limbs, the form *boty 'shoes'* denotes a pair or several pairs of shoes, the form *klíče 'keys'* means a bundle or more bundles of keys. The meaning is referred to as the "pair/group meaning" in the present paper.

As the pair/group meaning is compatible with most Czech concrete nouns and it manifests in some peculiarities as to the compatibility of the particular nouns with numerals,[12] we proposed to treat the pair/group meaning as a grammaticalized meaning constituting a new grammatical category of Czech nouns (Panevová and Ševčíková, 2011).

5.1 Grammateme typgroup

For the purpose of including the pair/group meaning into the tectogrammatical annotation of PDT 2.5, a new grammateme typgroup was added to the existing set of 15 grammatemes used in PDT 2.0. For the typgroup grammateme, three values were defined: group for the pair/group meaning, single for the meaning of single entities, and nr ("not recognised") for unresolvable cases.

The pair/group meaning is closely related to the meanings of the number category. In connection with the annotation of the grammateme typgroup, values of the grammateme

[11] Because it is typically a case like the composer and a symphony annotated together as a concert performance.

[12] The counting of pair/group nouns requires using a set numeral instead of a cardinal one. This is a strong argument in favor of considering the pair/group meaning a grammatical category. Cf. the set numeral *dvoje 'two sets'* in the example *Máme dvoje sklenice – na bílé a červené víno. 'We have two-sets of glasses – for the white and for the red wine'* vs. the cardinal numeral *dvě 'two'* if counting single entities in the sentence *Postavil na stůl dvě sklenice. 'He put two glasses on the table'.*

number as implemented in the PDT 2.0 data had to be changed in some cases. For instance, if the plural form was identified as denoting a pair or group, the value pl (assigned to the node representing this form in PDT 2.0) was changed to sg in the PDT 2.5 data.

5.2 Manual annotation of the typgroup grammateme with selected nouns

In the PDT 2.5, the grammateme typgroup was assigned semi-automatically with all nouns; the manual annotation concerned the nouns for which a higher frequency of the pair/group meaning was expected, the rest of the nouns was assigned a value of the typgroup grammateme automatically.

Occurrences for manual assignment were selected on the basis of a list of tectogrammatical lemmas (t-lemmas) of prototypical pair/group nouns. Nouns which co-occur with a set numeral in the PDT 2.0 and in the SYN2005 (ÚČNK, 2005) corpus data were analyzed as good candidates for this list. The list was further enriched using grammar books and theoretical studies on number in Czech as well as linguistic introspection. In the resulting list, 141 Czech nouns were involved, only 67 of them with 618 instances of plural forms were found in the PDT 2.5 data. Most of the nouns belong to one of the following groups:

- nouns denoting body parts occurring in pairs or groups (*uši 'ears'*, *prsty 'fingers'*, *vlasy 'hair'*),
- nouns denoting clothes and accessories for these body parts (*náušnice 'earrings'*, *rukavice 'gloves'*),
- nouns denoting family members such as *rodiče 'parents'*, *sourozenci 'siblings'*, and
- nouns denoting objects of everyday use and foods sold or used in typical amounts (*klíče 'keys'*, *sirky 'matches'*, *cigarety 'cigarettes'*, *sušenky 'biscuits'*).

The plural forms to be annotated were extracted from the data together with a short, both preceding and following context. In order to make the annotation task as simple as possible, the annotators did not specify the values of both the typgroup and number grammatemes, but they were asked to choose one of the annotation choices 1 to 6; the correspondences between the annotation choices and the grammateme values are described in Table 2. All 31 files were annotated manually by two annotators (native Czech speakers) in parallel during four months, the annotation was preceded by a short training period. The language intuition of native speakers played a crucial role in the annotation process. The annotators agreed on 464 (75.1%) out of 618 instances annotated, with a Cohen's Kappa score of 0.67. After the manual parallel annotation, instances of disagreement were adjudicated by a third annotator and the instances on which annotators agreed were revised in order to check the correctness and consistency of the annotation. The frequency of the choices in the revised annotation is summarized in the last column of Table 2.

5.3 Automatic assignment of the typgroup grammateme to the remaining nouns

Nouns which were not in the list (and consequently in the manual annotation) were assigned a value of the typgroup grammateme fully automatically. A simple, two-step procedure was provided for the automatic annotation: in the first step, nouns accompanied

Annotation choice	Grammateme values typgroup	number	# of instances (percentage)	
1 - plurality	single	pl	133	(21.5%)
2 - one pair/group	group	**sg**	230	(37.2%)
3 - several pairs/groups	group	pl	30	(4.9%)
4 - one pair/group or several pairs/groups	group	**nr**	154	(24.9%)
5 - cannot be resolved	nr	**nr**	70	(11.3%)
6 - —	*to indicate a mistake*		1	(0.2%)

Table 2: Manual annotation: annotation choices and corresponding combinations of the values of the grammatemes typgroup and number and their frequency in the manually annotated data. The number values marked in bold were changed from the pl value (as available in the PDT 2.0 annotation) to the marked value, influenced by the annotation of the pair/group meaning.

Grammateme values typgroup	number	# of instances
single	sg	185086
single	pl	59912
single	nr	10232
group	sg	237
group	pl	35
group	nr	153
nr	nr	66

Table 3: Combinations of values of the grammatemes typgroup and number and their frequency in the PDT 2.5 data.

with a set numeral *jedny 'one-pair/group'* (except for pluralia tantum) were assigned the value group of the grammateme typgroup and the value of the grammateme number was changed to sg in this connection; if the noun collocated with a set numeral of a higher numeric value (*dvoje 'two-pairs/groups-of'*, *troje 'three-pairs/groups-of'* etc.), the value group was filled in the grammateme typgroup whereas the grammateme number remained unchanged (i.e. pl). Secondly, all the other nouns were assigned the value single in the grammateme typgroup, the value of the grammateme number was not changed in these cases, compared to the original (PDT 2.0) annotation.

Combinations of the values of the grammatemes typgroup and number in the full PDT 2.5 data and their frequency is displayed in Table 3.

6 Automatic annotation of clause segmentation

Analytical trees in PDT 2.5 have been enriched with annotation of clause segmentation. Clauses are grammatical units out of which complex sentences are built. A clause typically corresponds to a single proposition expressed by a finite verb and all its arguments and modifiers (unless they constitute clauses of their own).[13] Annotation of clauses can be

[13]Given that Czech is a pro-drop language (pronouns in subject positions are often dropped, since their gender, number and person values are already expressed by verb inflection), this definition based on finite verbs matches

used for training clause boundary identifiers, which are generally supposed to be helpful in a number of NLP tasks such as parsing (since most dependencies do not cross clause boundaries), text summarisation (for instance, relative clauses might contain information of lesser importance and thus are more likely to be removable), machine translation (most reordering patterns are to be applied inside clauses), and speech applications (clause boundaries often imply prosodic boundaries).

We believed that clause boundaries could be identified automatically with very high reliability, since gold-standard morphological and more importantly, analytical annotation had already been available. Therefore clause boundaries were annotated manually only in a small portion of the PDT data and this annotation was used for developing a rule-based clause-identification procedure. To make the annotation consistent across the whole data, all the clause annotation distributed in PDT 2.5 was generated by this procedure; the original manually annotated samples are not included in PDT 2.5.

6.1 Basic conventions for clause representation

Technically, clause boundaries are represented by the dedicated attribute `clause_number` added to analytical nodes. If two analytical nodes in a tree share the same non-zero value of this attribute, then they belong to the same clause. Zero value of this attribute is reserved for boundary tokens, i.e. tokens that are located on the boundary of two clauses and cannot be unequivocally assigned to either of these clauses. Boundary tokens are typically various types of punctuation marks or coordinating conjunctions.[14]

Coindexing by the dedicated attribute is rendered by colours in the PDT 2.5 clause segmentation samples below:

(3) *U sochy básníka seděl vlasatý mladík a⋆ hrál Vysockého písně.†*
transl.: *There was a hairy guy sitting at a statue of a poet playing Vysockij's songs.*
⋆ Clause boundary is formed by the coordinating conjunction between the two clauses.
† Sentence boundary is manifested by the final punctuation.

(4) *Pokud jde o kupní smlouvu a⋆ všechny náležitosti s ní spojené,† musí si to zařídit a⋆ zaplatit strany samy.*
transl.: *Considering the buying contract and all related requirements, it has to be set and paid by contracting parties themselves.*
⋆ The coordinating conjunction that joins *sentence constituents* belongs to the clause.
† Clause boundary manifested by the punctuation symbol.

(5) *Lidé na nás tehdy chodili, aby⋆ se odreagovali od přítomného režimu.*
transl.: *People in those days used to attend our sessions, so that they could lay off the present government.*
⋆ The subordinating conjunction belongs to the subordinate clause.

(6) *Posunovač, který prý vstoupil do kolejiště, aniž se rozhlédl, je nyní v nemocnici⋆ .*
transl.: *The switchman that is said to enter the railyard without even looking around is in the hospital.*
⋆ The matrix clause is split into two parts by the embedded relative clause, which is further modified by the dependent clause.

(for Czech) the traditional notion of a clause as a group of words having a subject and a predicate.

[14] Note that subordinating conjunctions are systematically annotated as part of the respective dependent clause. The reason for this decision lies in their linguistic properties: subordinating conjunctions in Czech make an integral part of the dependent clause and, if omitted, the clause might become ungrammatical.

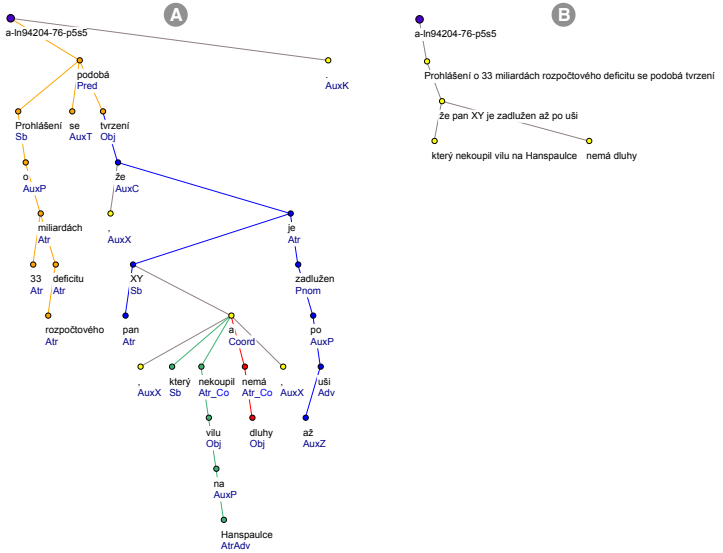


Figure 1: Two ways to visualise sentence segmentation in TrEd. Translation in Figure 3.

Clause segmentation can be comfortably visualised in the tree editor TrEd (see Figure 1) in two styles: either in full (unfolded) or in folded trees. In the former case, the tree topology is displayed as usual and clause segmentation is signalled by node and edge colours (see Figure 1A). In the latter case, the set of nodes belonging to one clause collapses to a single node which represents the whole clause (see Figure 1B).

6.2 Annotation procedure for clause segmentation

The automatic clause identification procedure can be outlined as follows:

1. Clause seeds are identified. Every occurrence of a finite verb form (the POS tag identifies finiteness reliably) is marked as a distinct clause seed.
2. Seeds forming a compound verb are joined together. Seeds with the analytical function of an auxiliary verb (AuxV) cannot constitute a clause on their own.
3. The tree is recursively traversed (post-order) and each coordination head is temporarily added to the clause of its rightmost member that already belongs to a clause.
4. Clause completion step. The tree is traversed recursively and the children that do not yet belong to any clause are typically added to the clause of the parent node (special handling of coordinations is needed here), or to their nearest left or right sibling that already constitutes a clause.
5. All potential boundary nodes are excluded from the clauses and their clause membership is re-estimated. The criteria is based mostly on the linear order of tokens but attention is also paid to the tree structure.

```
t-root [
    atree.rf $aroot,
    2+x member mwes [  ],
    descendant t-node [
        gram/typgroup = "group"
] ];

a-root $aroot := [
    1+x descendant a-node [
        clause_number = 3
] ];
```

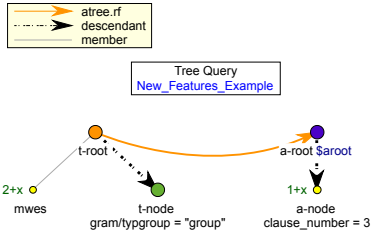


Figure 2: The PML-TQ query used to obtain Figure 3 in a textual and in a graphical form. It searches for a sentence with at least three clauses, two MWEs, and one word with the pair/group meaning.

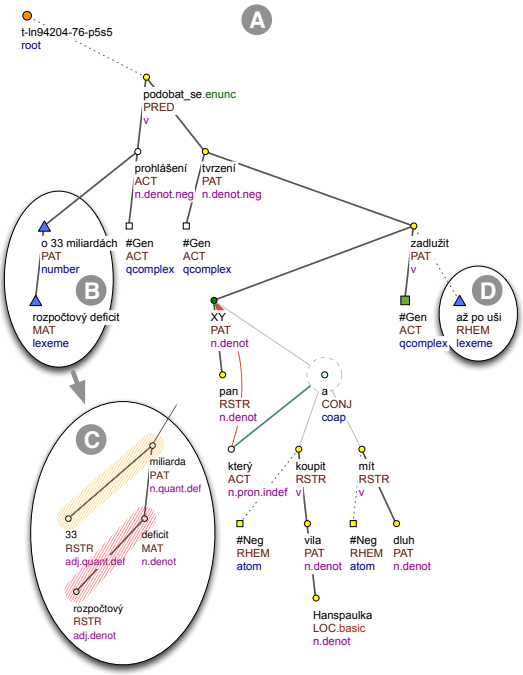


Figure 3: An example tree showing all the new features of PDT 2.5 as shown in TrEd: *Prohlášení o 33 miliardách rozpočtového deficitu se podobá tvrzení, že pan XY, který nekoupil vilu na Hanspaulce a nemá dluhy, je zadlužen až po uši**.
transl.: *The statement on 33 billion [of] budget deficit is similar to the statement that Mr. XY, who didn't buy a luxury villa and doesn't owe to anyone, is up to his ears* in debt.*
* pair meaning

6.3 Evaluation and application on PDT 2.5 data

For the purpose of evaluation, we obtained data from a pilot project on annotating sentence structure, whose methodology is thoroughly formulated in (Lopatková et al., 2012). The project provided us with a valuable collection of 2505 manually annotated sentences. We use these gold-standard sentences for evaluation of our automatic clause-identification procedure. Despite being a subset of the PDT data, the manually annotated sentences are not shipped with PDT 2.5 due to reasons explained below; for clause boundaries, all the PDT 2.5 data are annotated automatically.

Mostly because of the different scope of the project, we have adopted a slightly different annotation scheme. Let us summarise the original concepts and emphasise the differences.

The theory behind the pilot project (Lopatková et al., 2012) is centred on *segmentation charts*. Prior to manual annotation, tokenised and morphologically annotated sentences are automatically split into individual *segments*. All punctuation marks and coordinating conjunctions serve as *segment boundaries*. A single clause then consists of one or more segments. This scheme is viable given the very strict rules for punctuation in Czech – there must be some kind of a boundary between two finite verb forms, be it a sentence boundary, punctuation or a conjunction. The task of the annotators was to identify individual clauses, i.e. to group the segments forming a single clause, and to assign the appropriate level of embedding, thus allowing the distinction between coordination and super- or subordination. The usage of analytical layer during the annotation was intentionally quite limited. Only the analytical functions of tokens were used to help the annotators decide on the correct level of embedding and to disambiguate if more readings of a particular sentence were possible.

Unlike the manual annotation, the automatic clause-identification procedure does not rely on the boundary segments and extensively uses analytical trees. There are three key differences in the annotation rules:

(a) The automatic procedure does not attempt to assign levels of embedding, as the interclausal relations are explicitly captured in the analytical tree.
(b) Segment boundaries delimiting segments within the scope of a single clause are annotated as part of the clause, so that the distinction between coordination of sentence members and coordination of clauses is made obvious.
(c) A parenthetical expression is not considered a separate clause unless it contains a finite verb form.

Especially the last rule created the need of further post-processing of the gold-standard data, to make automatic evaluation possible. During the post-processing, parenthetical expressions were automatically merged with their surrounding clauses. In the original manually annotated data there are 2,505 sentences divided into 5,311 clauses. After post-processing, the number of clauses drops to 4,948.

The evaluation was performed on the basis of clauses using standard precision, recall, and f-measure metrics, reaching values 0.973, 0.978, and 0.975 respectively.[15] This confirmed the initial hypothesis that a highly reliable segmentation can be induced from the already available dependency annotations. As for the few remaining wrongly recognized

[15] Each automatically recognised clause was considered correct if and only if there is a clause in the manually annotated data spanning the very same set of nodes.

boundaries, the error analysis has shown that they have no single dominating cause that could be easily fixed. Such sentences are often difficult to annotate even for a human, for instance because of elipsis or intricate interplay of hypotactic and paratactic structures.

The automatic clause-identification procedure was used to annotate all sentences provided with gold-standard analytical trees in PDT 2.5, which amounts to 87,913 sentences. The procedure identified 153,434 clauses.

7 Conclusion

The Prague Dependency Treebank has been used as a model for several other treebanks, showing that both the general linguistic model of FGD and the technical realisation of PDT using PML are flexible and generic. They are not limited to a particular language, or a language family. By now there are at least five treebanks[16] annotated in the "PDT style".[17]

We have shown that PDT is exceptional in the richness of the information it provides. PML fits this richness well and thus all the PML-based tools such as the `TrEd` editor and the PML-TQ tree-query language (Pajas and Štěpánek, 2009) can be seamlessly used with PDT. PDT 2.5 is the most complex release of PDT to date. It is an intermediate step on the way to PDT 3.0, which will add even more annotation (discourse and extended anaphora, for example). It also contains corrections of more than 10,000 technical and annotation errors found in the previous release.

In Sections 4 through 6 we have presented major new features of PDT 2.5: what they are, how they were obtained, and what is the resulting quality and reliability. In Figures 2 and 3 there is an example of a complex query involving all of these features using the PML-TQ search tool and a result found together with its visualizations.

PDT 2.5 and all of the tools mentioned above are freely available (not just) for research purposes under standard, permissive licenses at `http://ufal.mff.cuni.cz/pdt2.5` and in the LINDAT-Clarin repository at `http://lindat.cz`. The Prague Dependency Treebank 2.5 itself has a citable persistent identifier `http://hdl.handle.net/11858/00-097C-0000-0006-DB11-8`.

Acknowledgement

This work has been using language resources developed and/or stored and/or distributed by the LINDAT-Clarin project of the Ministry of Education of the Czech Republic (project LM2010013). The work on this project was supported by the grants P406/2010/0875, P406/12/P175, and P406/10/P193 of the Grant Agency of the Czech Republic.

[16] Prague Dependency Treebank, Prague Arabic Dependency Treebank, Prague Czech-English Dependency Treebank, Index Thomisticus Treebank, Slovak Treebank (analytic layer).

[17] The markup language PML is even more generic. In fact, it has no direct connection to FGD and can be used to represent treebanks in any linguistic formalism (phrase-based or dependency, or any other variety based on trees in the basic graph-theory sense). Currently almost thirty treebanks are available in PML and can be queried online using our PML-TQ server. We have yet to meet a treebank that can't be converted to PML.

References

Baldwin, T., Bannard, C., Tanaka, T., and Widdows, D. (2003). An empirical model of multiword expression decomposability. In *Proceedings of the ACL 2003 workshop on Multiword expressions*, pages 89–96, Morristown, NJ, USA. Association for Computational Linguistics.

Cruse, D. A. (1986). *Lexical Semantics*. Cambridge University Press.

Filipec, J. (1994). Lexicology and lexicography: Development and state of the research. In Luelsdorff, P. A., editor, *The Prague School of Structural and Functional Linguistics*, pages 163–183, Amsterdam/Philadelphia. J. Benjamins.

Hajič, J., Panevová, J., Hajičová, E., Sgall, P., Pajas, P., Štěpánek, J., Havelka, J., Mikulová, M., Žabokrtský, Z., and Ševčíková Razímová, M. (2006). Prague Dependency Treebank 2.0. CD-ROM. Linguistic Data Consortium.

Ide, N. and Suderman, K. (2009). Bridging the gaps: Interoperability for GrAF, GATE, and UIMA. In *Proceedings of the Third Linguistic Annotation Workshop*, ACL-IJCNLP '09, pages 27–34, Suntec, Singapore. Association for Computational Linguistics.

Lopatková, M., Homola, P., and Klyueva, N. (2012). Annotation of sentence structure: Capturing the relationship between clauses in Czech sentences. *Language Resources and Evaluation*, 46(1):25–36.

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

Mel'čuk, I. A. and Polguère, A. (1987). A formal lexicon in The Meaning-Text Theory (or how to do lexica with words). *Computational Linguistics*, 13(3-4):261–275.

Meyers, A., Reeves, R., Macleod, C., Szekely, R., Zielinska, V., Young, B., and Grishman, R. (2004). The NomBank Project: An Interim Report. In Meyers, A., editor, *HLT-NAACL 2004 Workshop: Frontiers in Corpus Annotation*, pages 24–31, Boston, Massachusetts, USA. Association for Computational Linguistics.

Mikulová, M., Bémová, A., Hajič, J., Hajičová, E., Havelka, J., Kolářová, V., Kučová, L., Lopatková, M., Pajas, P., Panevová, J., Razímová, M., Sgall, P., Štěpánek, J., Urešová, Z., Veselá, K., and Žabokrtský, Z. (2006). Annotation on the tectogrammatical level in the Prague Dependency Treebank. Annotation manual. Technical Report 30, ÚFAL MFF UK, Prague, Czech Rep.

Osenova, P. and Kolkovska, S. (2002). Combining the named-entity recognition task and NP chunking strategy for robust pre-processing. In *Proceedings of The First Workshop on Treebanks and Linguistic Theories (TLT2002)*.

Pajas, P. and Štěpánek, J. (2006). XML-based representation of multi-layered annotation in the PDT 2.0. In Hinrichs, R. E., Ide, N., Palmer, M., and Pustejovsky, J., editors, *Proceedings of the LREC Workshop on Merging and Layering Linguistic Information (LREC 2006)*, pages 40–47, Genova, Italy.

Pajas, P. and Štěpánek, J. (2008). Recent Advances in a Feature-Rich Framework for Treebank Annotation. In Scott, D. and Uszkoreit, H., editors, *The 22nd International Conference on Computational Linguistics - Proceedings of the Conference*, volume 2, pages 673–680, Manchester, UK. The Coling 2008 Organizing Committee.

Pajas, P. and Štěpánek, J. (2009). System for querying syntactically annotated corpora. In *Proceedings of the ACL-IJCNLP 2009 Software Demonstrations*, pages 33–36, Suntec, Singapore. Association for Computational Linguistics.

Palmer, M., Gildea, D., and Kingsbury, P. (2005). The Proposition Bank: An Annotated Corpus of Semantic Roles. *Computational Linguistics*, 31(1):71–105.

Panevová, J. and Ševčíková, M. (2011). Jak se počítají substantiva v češtině: poznámky ke kategorii čísla. *Slovo a slovesnost*, 72:163–176.

Pecina, P. (2009). *Lexical Association Measures: Collocation Extraction*, volume 4 of *Studies in Computational and Theoretical Linguistics*. Institute of Formal and Applied Linguistics, Prague, Czech Republic.

Popel, M. and Žabokrtský, Z. (2010). TectoMT: Modular NLP framework. In Loftsson, H., Rögnvaldsson, E., and Helgadottir, S., editors, *Lecture Notes in Artificial Intelligence, Proceedings of the 7th International Conference on Advances in Natural Language Processing (IceTAL 2010)*, volume 6233 of *Lecture Notes in Computer Science*, pages 293–304, Berlin / Heidelberg. Iceland Centre for Language Technology (ICLT), Springer.

Sag, I. A., Baldwin, T., Bond, F., Copestake, A., and Flickinger, D. (2002). Multiword expressions: A pain in the neck for nlp. In *Third International Conference, CICLing*.

Sgall, P. (1967). *Generativní popis jazyka a česká deklinace*. Academia, Praha.

Sgall, P., Hajičová, E., and Panevová, J. (1986). *The Meaning of the Sentence in Its Semantic and Pragmatic Aspects*. Academia/Reidel Publ. Comp., Praha/Dordrecht.

ÚČNK – Institute of the Czech National Corpus (2005). Czech National Corpus - SYN2005. `http://www.korpus.cz`. Prague.

Weischedel, R. and Brunstein, A. (2005). BBN Pronoun Coreference and Entity Type Corpus. CD-ROM. Linguistic Data Consortium.

Xue, N., Jiang, Z., Zhong, X., Palmer, M., Xia, F., Chiou, F.-D., and Chang, M. (2010). Chinese treebank 7.0. CD-ROM. Linguistic Data Consortium.

Deriving a Lexicon for a Precision Grammar from Language Documentation Resources: A Case Study of Chintang

Emily M. BENDER[1] *Robert SCHIKOWSKI*[2]
Balthasar BICKEL[2]
(1) Department of Linguistics, University of Washington
(2) Seminar für Allgemeine Sprachwissenschaft, Universität Zürich
ebender@uw.edu, robert.schikowski@uzh.ch, balthasar.bickel@uzh.ch

ABSTRACT
Language documentation projects typically invest a lot of effort in creating digitized lexical resources, which are used in the creation of dictionaries and in the glossing of collected texts. We present and evaluate a methodology for repurposing such a lexical resource developed for Chintang (ISO639-3: ctn), a language of Nepal, for use with a precision implemented grammar developed in the DELPH-IN formalism. The target lexicon, when combined with a set of morphological rules, achieves 57% type-level coverage and 50% token-level coverage of held-out texts, while maintaining a feature-level accuracy F-measure of 70%. As lexicon development is typically one of the most expensive aspects of creating a precision grammar, this represents a significant savings of effort.

TITLE AND ABSTRACT IN GERMAN

Ableitung des Lexikons für eine Präzisionsgrammatik aus dokumentationslinguistischen Ressourcen anhand einer Fallstudie zum Chintang

Typische Sprachdokumentationsprojekte investieren viel Zeit in den Aufbau digitaler lexikalischer Ressourcen, die für die Erstellung von Wörterbüchern und für die Glossierung von Korpustexten genutzt werden können. Dieser Vortrag stellt eine alternative Verwendung eines elektronischen Wörterbuchs vor, das für das Chintang (ISO639-3:ctn), eine bedrohte Sprache Nepals, entwickelt wurde. Die Kombination dieses Wörterbuchs mit einer nach dem DELPH-IN-Formalismus entwickelten Präzisionsgrammatik in Form morphologischer Regeln kann erste Texte auf der Type-Ebene zu 57% und auf der Token-Ebene zu 50% abdecken, wobei auf der Merkmalsebene ein F-Maß von 70% gewahrt wird. Da der Aufbau lexikalischer Ressourcen zu den zeitintensivsten Komponenten der Entwicklung einer Präzisionsgrammatik gehört, bringt diese Methode eine signifikante Zeitersparnis mit sich.

KEYWORDS: lexical acquisition, grammar engineering, endangered languages, low-resource languages, language documentation.

KEYWORDS IN GERMAN: Lexikonerstellung, Grammar Engineering, bedrohte Sprachen, Sprachen mit geringen Ressourcen, Sprachdokumentation.

Proceedings of COLING 2012: Technical Papers, pages 247–262,
COLING 2012, Mumbai, December 2012.

1 Introduction

Endangered languages represent an especially urgent type of low-resource languages: Not only do they generally lack computational resources, but they also unfortunately have the property that the window in which to create such resources is small and closing. Thus to the extent that any computational resources are created, they are particularly valuable, and if they can be repurposed for applications beyond their original target, this extends their value further.

This paper describes a case study in the repurposing of a lexical resource for an endangered language, Chintang. The original lexical resource is a Toolbox[1] lexicon developed to assist in the glossing of collected texts and in the development of a conventional dictionary. We present a methodology for automatically translating this lexicon into one which can be used as part of a precision grammar for the language, suitable for both parsing and generation and eventually in applications including machine translation. The grammar is a starter grammar generated by the LinGO Grammar Matrix grammar customization system (Bender et al., 2010), and includes an initial implementation of Chintang verbal morphology. As a starter grammar, it still has very limited coverage. However, even broad coverage grammars rely heavily on the size and quality of their lexicons. Our focus here is therefore on the extent to which the existing resources for Chintang can be used to bootstrap a lexicon for this implemented grammar.

This paper is structured as follows: In Section 2, we provide background on the Chintang language, the project which is documenting it, and the relevance of precision grammars for language documentation. Section 3 describes our methodology for creating the Grammar Matrix-compatible lexicon on the basis of the Toolbox lexicon and the implementation of the morphology. We evaluate the lexical coverage of the resulting grammar over held-out texts in Section 4. Section 5 situates this work with respect to related initiatives.

2 Background

2.1 Chintang Language

Chintang (ISO639-3: ctn) is a Kiranti language spoken in the hills of Eastern Nepal. The next bigger city close to the village of Chintang is Dhankuta, which is a six hours footwalk away. The local economy is centered around agriculture, both for subsistence and for trade.

The Chintang language belongs to the large Sino-Tibetan (Tibeto-Burman) family. The exact position of the Kiranti languages within this family is unclear (Ebert 2003). Within Kiranti, Chintang belongs to the Eastern branch, which is characterized by the development of the preglottalized stops reconstructed for Proto-Kiranti by Michailovsky (1994) to aspirated stops.

The number of speakers of Chintang is generally estimated to be around 5000 (e.g. Bickel et al. 2007), and this is in accordance with the speakers' own estimations. There are no reliable official data. Most speakers are bilingual, speaking Nepali, the national language of Nepal, as the second language. Many in addition speak Bantawa, a big neighboring Kiranti language. Chintang is still being learned by children (Stoll et al., 2012), but language knowledge and transmission are clearly on the decline, especially in the more easily accessible parts of the village.

[1] http://www.sil.org/computing/toolbox/index.htm

2.2 Research projects and resources

Research on Chintang started with the Chintang and Puma Documentation Project (CPDP), which ran from 2004 to 2009. Since 2009, research has been continued by a group of several collaborative projects together referred to as the Chintang Language Research Program (CLRP).[2] Several native speakers of Chintang and Nepali were employed during the projects to make recordings, conduct interviews, and to transcribe and translate recordings. Presently there is an office at the Tribhuvan University, Kathmandu, where five transcribers and translators are constantly working on the corpus, but no new recordings are being made.

The corpus comprises about 280 hours of video recordings, the majority of which have been transcribed by now (250 hours, containing 1,130,000 words; Bickel et al., 2009ff). Transcribed sessions are first translated from Chintang to Nepali and then from Nepali to English. The English translation is an important aid for the final step of glossing, which is done by student assistants studying linguistics. So far approximately 620,000 words have been glossed. Additional annotations are added to parts of the corpus depending on the needs of individual projects. Examples include the annotation of pointing gestures or of referential properties such as identifiability.

The compilation of the corpus is tightly coupled with the Chintang dictionary, which presently has about 9,000 words. The electronic version was created along with the corpus, so all words in the corpus are in the dictionary. Some systematic elicitation work to cover semantic fields that do not frequently come up in everyday conversation was carried out in 2010, and a printed version for the speaker community was published in 2011 (Rāī et al. 2011). The electronic dictionary keeps growing as more and more words are glossed. New words collected by the glossers are integrated into the main dictionary twice a year.

Both the corpus and the dictionary are in Toolbox format. In Toolbox, files are divided into structurally similar records (utterances in the case of the corpus, entries in the case of the dictionary). Each record consists of several lines where each line starts with a so-called field marker indicating the type of information (e.g. phonological words, morphemes, morpheme glosses) followed by content (see Section 3 below for details). It is possible to align the tiers thus defined, enabling composite searches (e.g. "find all morphemes of the shape *cekt* which have been glossed as 'speak'"). Since a major revision of the dictionary in 2010, all dictionary entries have IDs, which are inserted and aligned with morphemes upon glossing. This makes it possible to automatically look up detailed information for each corpus morpheme in the dictionary.

The entries in the dictionary include stem forms, alternate forms, glosses in English and Nepali, as well as some grammatical information. In particular, the dictionary lists coarse-grained part of speech (drawn from a set of 30 tags) as well as fairly detailed information about the syntactic and semantic valence of verbs, that is, the number of arguments expected, the cases for each argument, and an indication of which argument(s) the verb agrees with. This information is encoded as a string. (1) gives the valence information for *bhend* 'loosen', indicating that this verb takes two arguments. The most agent-like argument ("A") is marked with ergative case, the other (patient-like, "P") with nominative, and the verb will be inflected to agree with both of them.

(1) `\val A-ERG P-NOM V-a(A).o(P)`

[2] `http://www.spw.uzh.ch/clrp`.

This information reflects rich linguistic knowledge, the product of the analysis done by the annotators, and it is digitized. However, it is not really machine interpretable. While Toolbox can assist with morphological parsing (and thus with glossing of sentences), it does not make use of such syntactic information for syntactic parsing.

2.3 Precision Grammars for Language Documentation

Precision grammars are machine-readable sets of rules developed by hand to capture linguistic generalizations. Large-scale precision grammar projects have been carried out in a variety of linguistic frameworks, including HPSG (Pollard and Sag, 1994; Flickinger, 2000) (described further below), LFG (Kaplan and Bresnan, 1982; Butt et al., 2002)) and TAG (Joshi et al., 1975). DELPH-IN-style[3] HPSG grammars map surface strings to semantic representations in the format of Minimal Recursion Semantics (MRS; Copestake et al., 2005), and are reversible, i.e., suitable for use in both analysis (strings-to-MRS) and generation (MRS-to-strings).

Precision grammars can be deployed in transfer-based machine translation (e.g., Lønning et al., 2004), grammar checking applications (e.g., Suppes et al., 2012), and other NLP applications which benefit from a strong distinction between grammatical and ungrammatical strings (e.g., in generation) and/or detailed semantic representations. While broad coverage precision grammars can be expensive to build, the alternative of treebank-derived grammars presupposes resources which don't typically exist for endangered languages and are themselves costly to create. Furthermore, precision grammars, by locating analytical decisions in specific rules, can be more easily updated than treebanks, as more is understood about the language being described. Using the methodology of the Redwoods project (Oepen et al., 2004), precision grammars can be used to create treebanks which can be kept up to date with the grammar as it evolves. Both precision grammars and their associated treebanks can be valuable resources in language documentation (Bender et al., 2012).

2.4 The LinGO Grammar Matrix

As noted, precision grammars are time-consuming to develop. However, because similar structures recur across languages, the development time for new grammars can be reduced by repurposing grammar code developed for other languages. This is the idea behind multilingual grammar engineering projects, including the LinGO Grammar Matrix (Bender et al., 2002, 2010), ParGram (Butt et al., 2002; King et al., 2005), PAWS (Black and Black, 2009), and GF (Ranta, 2007). The Grammar Matrix stores a core grammar which includes (partial) analyses hypothesized to be cross-linguistically applicable, including basic phrase structure rule types for combining heads with different types of dependents, as well as an implementation of semantic compositionality, i.e., constraints which relate the semantic representation associated with a phrase to the semantic contributions of its daughters. In addition, the Grammar Matrix provides a series of libraries of analyses of cross-linguistically variable phenomena. These analyses are accessed through a web-based questionnaire which elicits a linguistic description of a language from a linguist and outputs a corresponding set of grammar files describing phrase structure rules, lexical rules, and lexical entries.

The information provided by the linguist is encoded in a plain text 'choices' file, where each 'choice' is a simple attribute-value pair. The customization system interprets the choices to output grammar files. The grammar files are encoded in the framework of Head-Driven Phrase

[3] `http://www.delph-in.net`

Structure Grammar (HPSG; Pollard and Sag, 1994), providing representations in the format of Minimal Recursion Semantics (Copestake et al., 2005) and are compatible with the DELPH-IN suite of grammar development and deployment tools, including the LKB (Copestake, 2002).

For the purposes of this work, the most important aspects of the Grammar Matrix are its support for the creation of lexical rules, which handle the ordering, basic form, and syntactico-semantic contributions of affixes, and its set of lexical types. Both the lexical rules and lexical types pair forms with complex feature structures. These feature structures encode syntactic and semantic information and are compatible with the feature structures for phrase structure rules, meaning that the lexical entries can be used as part of a grammar capable of both paring and generation.

The resources of the Grammar Matrix represent another rich source of linguistic knowledge, but in this case, the knowledge is not specific to a particular language. In order to create a grammar for a particular language, they need to be paired with information about that language: The forms and lexical meanings of individual words, their valence patterns, and the forms and effects of individual affixes. While affixes generally form a relatively small closed class, lexicons are another matter. The goal of this work is to see how effectively we can use the existing lexicographic work of the CLRP to flesh out a Grammar Matrix-derived lexicon for Chintang.

3 Methodology

3.1 Matrix Lexicons and Toolbox Lexicons

As described above, Toolbox lexicons are structured by user-designed fields (marked with initial tags) that store information including the orthography of a form, its gloss, example sentences, and any other information the lexicon developers would like to collect. In the case of the CLRP, this includes part of speech and detailed valence information (case and semantic roles). These are each encoded as a string in the value of the associated tag.

A lexicon for a DELPH-IN style grammar associates orthographic forms with complex feature structures representing morphological, syntactic and semantic information, encoded in such a way that this information can interact with lexical and phrase structure rules to license syntactic analyses of full sentences which furthermore embed compositionally created semantic representations. The relationship between the strings and these complex feature structures is mediated by lexical types which bear the constraints that describe the feature structures. The lexical types, in turn, are arranged into a multiple inheritance hierarchy so that each constraint need be stated only once and can be inherited by all lexical entries which require it.

```
\lex kond
\id 179
\psrev v
\val A-ERG P-NOM V-a(A).o(P)
\ge search; look.for
\dt 22/Feb/2011
```

Figure 1: Sample Toolbox entry from CLRP

A sample Toolbox entry is shown in Figure 1 while Figure 2 illustrates the corresponding Matrix entry. (Both are abbreviated, to focus on the most relevant information.) In Figure 1, the value of the tag `\lex` encodes the stem, `\ge` gives an English gloss, `\psrev` the part of speech, and `\val` the detailed valence information. Other fields not shown in the figure encode alternate forms of the entry, examples, and glosses in Nepali.

The Matrix entry in Figure 2, is a typed feature structure. The type of the whole structure is *trans-verb-lex*. This type provides (or inherits from its supertypes) most of the constraints on the entry. The only constraints provided directly in the lexical entry are the STEM value (*kond*, corresponding to `\lex` in Figure 1) and the PRED value *_search;look.for_v_rel*, i.e., the predicate symbol for the semantic relation associated from this entry. This is built on the basis of the `\ge` field of the Toolbox entry.

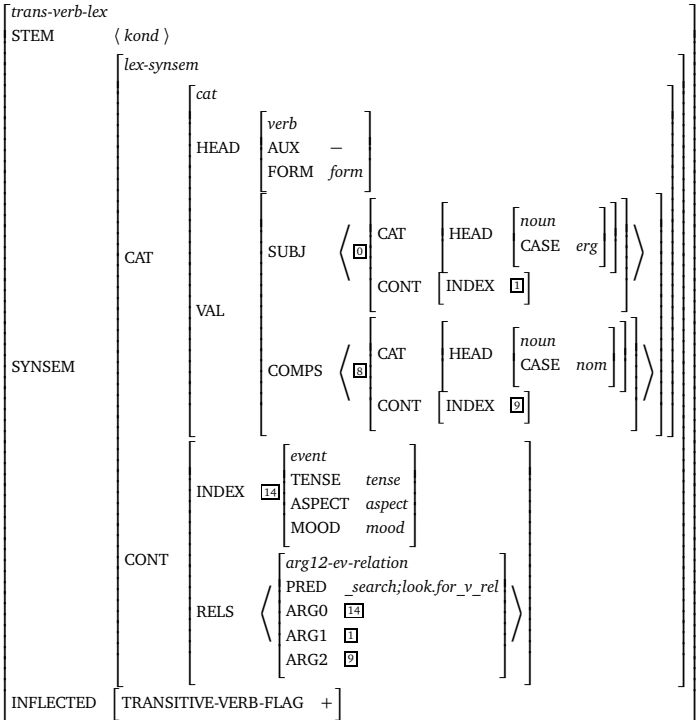


Figure 2: Sample Matrix entry corresponding to Figure 1

Turning to the information contributed by *trans-verb-lex*, the HEAD value indicates that this is a verb (and will head verbal projections such as VP and S), that it is not an auxiliary ([AUX −]), and that its form value is as yet underspecified ([FORM *form*]). When this lexical entry is inflected as either a finite verb or a non-finite verb, the FORM value will be constrained accordingly. The INDEX value is linked to the ARG0 of the relation contributed by the verb, and has underspecified values for TENSE, ASPECT, and MOOD. These, too, can be filled in via lexical rules for affixes that mark these values.

The VAL information indicates that this verb is seeking a subject and a complement, both headed

by nouns, where the subject must be in the ergative case and the object in the nominative.[4] Furthermore, the INDEX values of each are linked to the ARG1 and ARG2 positions in the semantic predicate, respectively. The number of arguments and the linking to the semantic roles is part of the cross-linguistic definition of a transitive verb in the Matrix. The constraints that both arguments are NPs (rather than, say, PPs) and the information about case come from specializations to the transitive verb type defined for Chintang by hand through the Grammar Matrix customization system questionnaire.

Finally, the feature INFLECTED is related to the morphotactic system. The value of this feature is a bundle of further 'flag' features (Goodman, 2012) tracking whether certain lexical rules have or have not applied, in order to encode dependencies between lexical rules and between lexical rules and lexical types. Here, we have shown only the TRANSITIVE-VERB-FLAG feature, whose + value will ensure that affixes throughout the affix chain will only be those that are compatible with transitive verbs.

3.2 Import of Lexical Entries from Toolbox Lexicons

The Grammar Matrix customization system provides facilities for the definition of lexical types. This is in principle unbounded: the user can define, for example, types for both common and proper nouns, as well as types for nouns of different genders and types for verbs with different case frames. The user can specify constraints on these types through the customization system (e.g., constraints on noun gender or case frames). Many other constraints, particularly those concerned with semantic composition, are inherited from the Matrix core grammar.

We extended the Grammar Matrix customization system to include a subpage that allows the user to define mappings between sets of properties encoded in a Toolbox lexical entry and user-defined lexical types. Figure 3 gives an example. This type maps entries from the Toolbox lexicon which are specified to have the part of speech 'v' and the valence 'S-NOM V-s(S)' to the type 'verb1'. This type is defined on another page of the customization system questionnaire to describe intransitive verbs with nominative case on their sole argument. Types inherited from the Matrix core grammar provide the constraints that contribute a one-place semantic predicate and link the sole syntactic argument to the semantic argument. The import facility creates the name symbol for that semantic predicate on the basis of the gloss or alternatively of the orthography of the stem (as specified by the user).

Since Toolbox allows users to define their own tags, the extension we designed for the Grammar Matrix customization system does not make any assumptions about the name or number of tags which will be relevant to each import class. Users fill in the name of the tag in the 'Toolbox tag' field for each tag-value pair, and can add arbitrarily many tag-value pairs with the 'Add' button. Another part of the page allows the user to specify the location of a Toolbox file to import from and upload it. Though Chintang was the initial test case for this import facility, there is (to our knowledge) nothing specific to Chintang nor the CLRP in the design of the system. It is available for use through the Grammar Matrix customization system's web-based

[4] The Grammar Matrix uses the names SUBJ and COMPS for the valence features, but makes relatively few assumptions about which properties accrue to the argument in SUBJ as opposed to those in COMPS cross-linguistically. For example, as case and agreement are both handled lexically, the system is flexible enough to model even tripartite case and agreement, where the sole argument of intransitives is handled differently from either argument of transitives (Drellishak, 2009). Similarly, grammar developers using the Grammar Matrix customization system can define multiple different classes within transitive and intransitive which behave differently with respect to agreement and/or case.

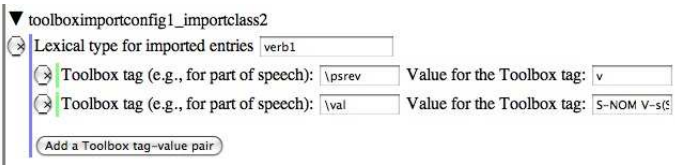


Figure 3: Sample Toolbox import class

questionnaire.[5]

The Grammar Matrix core grammar provides support for a wide variety of semantic valences. However, the present system only exposes simple intransitive and transitive valences to the customization page. As soon as the customization system is updated to expose more valence possibilities in the definition of lexical types, types using such valence possibilities will be available as targets for the import of corresponding Toolbox lexical entries, without any further updates to the extension we created. For the purposes of this study, however, we are limited to nouns and simple transitive and intransitive verbs.

The final 'choices' file for the Chintang grammar specifies import configurations for common nouns and two types each of transitive and intransitive verbs: native Chintang verbs which take the full range of inflectional morphology and verbs borrowed from Nepali which must co-occur with an auxiliary. This results in imported lexical entries for 4,741 common nouns, 282 native Chintang intransitive verbs, 142 borrowed Nepali intransitive verbs, 285 native Chintang transitive verbs, and 190 borrowed Nepali transitive verbs. Thus 5,640/9,034 (62%) of the entries in the Toolbox lexicon resulted in entries for the Matrix grammar, including 899/1,440 (62%) of the verbs. The most frequent remaining part of speech categories in the Toolbox lexicon file are adverbs (866), adjectives (515), interjections (377), and affixes (286).[6][7]

3.3 Implementation of Morphology

Chintang has a relatively complex morphological system, especially for verbal inflection. Schikowski (2012) identifies 12 suffix positions following verbs. On the other side of the stem, a verb root may take up to 4 prefixes, and additionally endoclitics can appear inside the prefix chain. One special type of prefix is that found in bipartite stems, where a specific prefix is idiosyncratically selected by the verb and is in effect part of the stem, though not always realized contiguously with the rest of the stem. To add to the complexity, the prefixes do not occur in a fixed order (Bickel et al., 2007). Beyond that, a single verbal word can contain up to four verb roots, each of which can host prefixes and suffixes. In these 'verb chains', any given prefix can appear only once, while suffixes are frequently repeated (Bickel et al., 2007; Schikowski, 2012). Finally, there is a host of morphophonological effects, some categorical

[5] `http://www.delph-in.net/matrix/customize/matrix.cgi`

[6] The Toolbox lexicon includes words from four different languages (Chintang, Nepali, Bantawa, and English), as words from all four of these languages appear in the collected data. These numbers reflect the full Toolbox file, and so are not directly representative of only Chintang. For example, Chintang has only two adjectives; the rest of the adjective entries come from other languages.

[7] These part of speech counts are based on the `\ps` field in the lexicon. The import of lexical entries was done based on the `\psrev` field. This field does not cover as many words as the older `\ps` field, but has been thoroughly reviewed.

and some variable, which change the surface form of any given morpheme depending on its phonological context (as well as on sociolinguistic factors).

The Grammar Matrix customization system provides extensive support for the definition of the morphotactic and morphosyntactic aspects of lexical rules, i.e., the order in which morphemes appear, co-occurrence restrictions between morphemes, and the syntactico-semantic constraints associated with each (Goodman, 2012). For this study, we defined a set of lexical rules through this system on the basis of the prose description of Chintang morphology in Schikowski 2012 and consultation between Schikowski (field linguist) and Bender (grammar engineer). Here we briefly describe the phenomena handled by this rule set.

There are a total of 160 verbal lexical rules (grouped into 54 position classes) and 24 nominal lexical rules (6 position classes) in the implementation. The position classes define the order of the affixes, including whether they are prefixes or suffixes and their relative order to other prefixes/suffixes. We handle verb chains by treating only the first verb root as an actual root in the model; this is facilitated by the fact that the first V position in a verb chain has the widest lexical variation. The current system allows up to three verb roots per chain, with the verb roots appearing in the second and third position treated as affixes. The position classes for each of these positions contain 32 rules, one for each verb root which can appear in non-initial position. The prefixes and suffixes which intervene between roots in the verb chain are treated as separate lexical rules for suffixes. This duplication of the lexical rule types is partially responsible for the high overall total of lexical rules.[8]

Elsewhere in the choices file we have specified information about the case system (number of cases and their names), possible values of tense, aspect and mood, possible values of person and number (including inclusive/exclusive distinctions in first person dual and plural), and other similar information. This allows us to model or partially model the syntactico-semantic effects of 131 of the 160 rules. The features targeted by these rules are shown in Table 1.[9] Examples of affixes whose syntactico-semantic effects are not modeled include the causative marker, possessive prefixes on nouns, and verb chain elements indicating the direction of motion or resulting position of a participant in the event. This information is not modeled because it is not directly supported by the customization system. The grammar output by the customization system is suitable for further hand-development, however, and nothing in HPSG theory or the DELPH-IN formalism would prevent encoding such information.

There are a few other ways in which this model of Chintang morphology is incomplete. First, it should be noted that we are abstracting away from most of the morphophonology by targeting the underlying representation given in the Toolbox files (both lexicon and corpus), rather than the transcription. Second, we are not modeling the phenomenon of free prefix order. This is possible in the DELPH-IN formalism but not supported by the Grammar Matrix customization system. Again, it would be relatively straightforward to modify the grammar to accommodate this, but we have chosen to focus our evaluation on the grammar as produced by the customization system. While the customization system can handle bipartite stems, and the facility for importing lexical entries from Toolbox anticipates this, we are not modeling them

[8] HPSG's type hierarchy in principle would allow us to define the morphosyntactic effects of these rules just once and cross-classify those types with the types encoding the position class information. The Grammar Matrix customization system interface, however, does not allow cross-classification of position class types with other kinds of types, so this kind of generalization must await hand-editing of the grammar files output by the customization system.

[9] Note that NEGATION isn't really a feature but rather a flag that causes the customization system to create a lexical rule which adds negation to the verb's semantic representation (Crowgey, ip).

Feature	# Rules	Notes
PERNUM	103	Person and/or number of verb's dependent or of noun
CASE	17	
FORM	35	Form (finite/non-finite) of verb
TENSE	27	
ASPECT	5	
MOOD	17	
NEGATION	8	

Table 1: Features constrained by lexical rules

at this time. Finally, while verb chains can in principle have four verbal roots, this model only allows up to three, because the four-root forms are extremely rare.

4 Evaluation

We created a choices file for Chintang which gives general grammatical information as well as definitions for lexical classes and lexical rules as described above. In addition, this choices file defines 11 closed-class lexical entries: 10 pronouns and one auxiliary. We used three sample narratives (totaling 2,906 word tokens) from the corpus as development data to refine the choices file. This process involved creating a grammar from the choices file using the customization system, loading the grammar into the LKB grammar development environment (Copestake, 2002) and processing the utterances in the narratives with the grammar using the [incr tsdb()] grammar profiling platform (Oepen, 2001). [incr tsdb()] provides facilities for browsing both results and errors encountered during parsing. These were used to identify forms that were not being handled appropriately. We then used the grammar exploration tools provided by the LKB to diagnose the source of the problem and then updated the choices file accordingly.

We then selected an additional four narratives to use as test data. The narratives range in length from 200 to 489 tokens (total: 1,453) and represent a range of domains: 'Durga_Exp' is a biographical monologue; 'pear_6-1' is a Pear Story (Chafe, 1980) elicited by asking the speaker to recount a story shown in a short, non-verbal film; 'story_rabbit' is a story about a clever rabbit who escaped a tiger; and 'choku_yakkheng' is a recipe for cooking nettle curry. We extracted the morpheme segmented line of each line in the narratives. An example is shown in (2), where the second line is the line we are targeting.

(2) thupro wassace uyuwakte pho
thupro wassak-ce u-yuŋ-a-yakt-e pho
many bird-NS 3NSS/A-live-PST-IPFV-IND.PST REP
'There lived many birds.' [ctn] story_rabbit.005

The performance of the grammar was evaluated in two ways. First, we evaluated coverage at both the type and token level over the test narratives. Table 2 gives the results. A word was counted as 'covered' if the grammar assigned it a morphological analysis that the grammar considered fully inflected. As shown in the table, the grammar found analyses for at least 50% of both word types and tokens across all the narratives, with the exception of 'story_rabbit' where the token-level coverage was only 35%. The ambiguity numbers in Table 2 reflect the

average over those words which had at least one analysis. These numbers reflect low ambiguity, with the maximal analyses per word form being only 8.

Narrative	total		# analyzed		% analyzed		avg ambiguity	
	type	token	type	token	type	token	type	token
Durga_Exp	206	489	120	265	58	54	1.24	1.14
choku_yakkheng	152	331	89	184	59	56	1.26	1.20
pear_6-1	206	433	105	203	56	51	1.20	1.62
story_rabbit	85	200	43	69	51	35	1.37	1.23
All	568	1453	324	721	57	50	1.40	1.27

Table 2: Coverage of customized grammar over test narratives

To get a sense of the accuracy of the resulting grammar, we randomly selected 10 word types from each of the four narratives (while ensuring that no word type was selected from more than one narrative). We used the LKB to parse each of these word types and compared the information in the resulting feature structure to the information in the gloss of the first instance of that word type in the narrative it was chosen from. We calculated precision and recall for each piece of grammatical information in the gloss and the feature structure.[10] In cases where the grammar found more than one analysis, we chose the best match to the gloss. The results are shown in Table 3.

Narrative	Total gold attributes	Precision	Recall	F-measure
Durga_Exp	16	.48	.88	.62
choku_yakkheng	21	.57	.62	.59
pear_6-1	31	.71	.92	.80
story_rabbit	29	.62	.83	.71
Total	97	.61	.82	.70

Table 3: Accuracy of customized grammar over 40 word types from test narratives

A large portion of the precision errors in this evaluation relate to cases where the grammar interprets the non-marking of some category as informative. For example, nouns that do not bear any affix for number as marked as singular in the grammar, and nouns not bearing any affix for case as nominative. These disagreements between the grammar and the glosses are counted as errors in Table 3, as the glosses are taken as the gold standard for this evaluation. However, the glosses reflect a systematic decision by the CLRP to not mark the contribution of zero morphemes. In most of these cases, the grammar is likely correct. Finally, verbs inflected for tense are considered to be finite by the grammar, and this is reflected in a (syntactic) feature FORM in addition to the semantic feature TENSE. The glosses mark non-finiteness explicitly, but do not mark finiteness separately from tense. Default singular number on nouns accounts for 19 errors, default nominative case 15, and finite form 9, of a total of 52 errors in precision. The 18 errors in recall are primarily due to cases where the intended lexical root is not available in the grammar but a homophone is.

Finally, we performed an error analysis to get a sense of the range of reasons a word form might

[10]For the gloss, we included all information provided. For the feature structure, we included only the predicate symbol from the root and any information that is added by some lexical rule in the grammar.

not be analyzed by the current grammar. We randomly selected 10 word forms from the four narratives that were not assigned analyses by the grammar. The failure of analysis of these 40 forms can be attributed to the following causes:

- [29 forms] Stems not imported to the grammar, because they don't match any of the import classes. These stems include verbs taking three arguments, adverbs, numerals, demonstratives, and other function words.
- [2 forms] Stems that are not in the version of the Toolbox lexicon used to import from.
- [4 forms] Affixes not implemented in the grammar.
- [5 forms] Other problems with the grammar, such as not allowing for case stacking and not allowing the affix order attested.

In general, we find the results of this evaluation encouraging: They suggest that the methodology presented here is effective at repurposing the results of the work on the Toolbox lexicon towards additional computational linguistic ends. Furthermore, the error analysis points the way towards effective means of improving the resulting grammar further, including fixing the specific errors with affixes that were identified, broadening the classes of verbs handled, adding adverbs, and creating lexical entries for high frequency closed-class words by hand.

5 Related Work

This work is similar in spirit to Bender's (2008) development of an implemented grammar for Wambaya (ISO639-3: wmb) based on the Grammar Matrix and a descriptive grammar. However, that work focused on hand-development of the grammar and included a manually entered lexicon, in contrast to our work on automatically populating the lexicon for the implemented grammar.

Other work applying grammar engineering and shared resources (including typological information) to endangered or other resource poor languages includes the Parser and Writer for Syntax system (PAWS; Black and Black, 2009) and Linguist's Assistant (Beale, 2011). We are not aware of any work addressing lexicon repurposing for these systems, but methodology analogous to what we propose in this paper should be applicable to them as well.

More generally, our work is situated within a broader context of reuse of lexical resources across formalisms and across systems. Other work along these lines includes the work of Kamei et al. (1997) and Bond et al. (2009) on making it possible to share user dictionaries across different MT systems, that of Bond et al. (2008) on repurposing a variety of resources (both WordNets and other lexical resources) in order to create a WordNet for Japanese and that of (McConville and Dzikovska, 2007) on creating lexical entries for a TRIPS grammar (Dzikovska, 2004) on the basis of FrameNet (Baker et al., 1998).

Conclusion and perspectives

The target lexical entries for a precision grammar derived from the Grammar Matrix are much more complex than the information explicitly encoded in even a thorough Toolbox lexicon. The work of developing the Toolbox lexicon is, however, the hard part. In this paper we have shown how it is possible to use a language-independent (i.e., explicitly multi-lingual) tool to leverage

the effort and linguistic analysis encoded in a Toolbox lexicon to create the kind of resource required for a machine-readable, precision grammar.

However, it is important to note that this is only a first step. Previous work building medium to large scale grammars with the DELPH-IN technology, including the broad-coverage English Resource Grammar (Flickinger, 2000, 2011) and a medium-sized grammar for Wambaya (Bender, 2008) suggest that it should indeed be possible to build a substantial grammar fragment for Chintang that uses this lexicon. The Wambaya grammar is especially pertinent for two reasons: first, like Chintang, it represents an application of the Grammar Matrix to a language not considered in its initial development, and second, its lexical types are based on the same general supertypes as those developed here for Chintang. Nonetheless, every language is different, and it is not possible to know without building the grammar whether the lexical types will be compatible with the specific grammatical phenomena attested in Chintang. We intend to develop such a grammar to test this in future work.

Acknowledgments

CPDP was funded by the VW Foundation as part of the DoBeS program (Grant No. II/79092 to Bickel) and CLRP by the VW Foundation (Grant No. II/81 730 to S. Stoll), the German Research Foundation (Grants No. BI 799/5-1 and BI 799/9-1 to B. Bickel) and the Swiss National Science Foundation (Grant Nr. 100012_140881 to Bickel).

In addition, this material is also based upon work supported by the National Science Foundation under Grants No. 0644097 and 1160274 to Bender. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

We are grateful to David Wax for assistance in integrating the software for importing lexical entries from Toolbox to the Grammar Matrix into the Grammar Matrix's web-based questionnaire.

References

Baker, C. F., Fillmore, C. J., and Lowe, J. B. (1998). The Berkeley FrameNet project. In *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics, Volume 1*, pages 86–90, Montreal, Quebec, Canada. Association for Computational Linguistics.

Beale, S. (2011). Using linguist's assistant for language description and translation. In *Proceedings of the IJCNLP 2011 System Demonstrations*, pages 5–8, Chiang Mai, Thailand. Asian Federation of Natural Language Processing.

Bender, E. M. (2008). Evaluating a crosslinguistic grammar resource: A case study of Wambaya. In *Proceedings of ACL-08: HLT*, pages 977–985, Columbus, Ohio. Association for Computational Linguistics.

Bender, E. M., Drellishak, S., Fokkens, A., Poulson, L., and Saleem, S. (2010). Grammar customization. *Research on Language & Computation*, pages 1–50. 10.1007/s11168-010-9070-1.

Bender, E. M., Flickinger, D., and Oepen, S. (2002). The grammar matrix: An open-source starter-kit for the rapid development of cross-linguistically consistent broad-coverage precision grammars. In Carroll, J., Oostdijk, N., and Sutcliffe, R., editors, *Proceedings of the Workshop*

on Grammar Engineering and Evaluation at the 19th International Conference on Computational Linguistics, pages 8–14, Taipei, Taiwan.

Bender, E. M., Ghodke, S., Baldwin, T., and Dridan, R. (2012). From database to treebank: Enhancing hypertext grammars with grammar engineering and treebank search. In Nordhoff, S., editor, *Electronic Grammaticography* (In press). University of Hawaii Press, Hawaii.

Bickel, B., Banjade, G., Gaenszle, M., Lieven, E., Paudyal, N., Rai, I., Rai, M., Rai, N. K., and Stoll, S. (2007). Free prefix ordering in Chintang. *Language*, 83(1):43–73.

Bickel, B., Gaenszle, M., Rai, N. K., Lieven, E., Banjade, G., Bhatta, T. N., Paudyal, N., Pettigrew, J., Rai, I. P., Rai, M., Schikowski, R., and Stoll, S. (2009). Audiovisual corpus of the chintang language, including a longitudinal corpus of language acquisition by six children, plus a trilingual dictionary, paradigm sets, grammar sketches, ethnographic descriptions, and photographs. *DOBES Archive*, `http://www.mpi.nl/DOBES`.

Black, C. A. and Black, H. A. (2009). PAWS: Parser and writer for syntax: Drafting syntactic grammars in the third wave. In *SIL Forum for Language Fieldwork*, volume 2.

Bond, F., Isahara, H., Kanzaki, K., and Uchimoto, K. (2008). Boot-strapping a WordNet using multiple existing WordNets. In *Proceedings of LREC 2008*, pages 1619–1624.

Bond, F., Okura, S., Yamamoto, Y., Murata, T., Uchimoto, K., Kato, M., Shimazu, M., and Suzuki, T. (2009). Sharing user dictionaries across multiple systems with UTX-S. In *Second International Workshop on Intercultural Collaboration (IWIC-2009)*, Stanford CA.

Butt, M., Dyvik, H., King, T. H., Masuichi, H., and Rohrer, C. (2002). The parallel grammar project. In Carroll, J., Oostdijk, N., and Sutcliffe, R., editors, *Proceedings of the Workshop on Grammar Engineering and Evaluation at the 19th International Conference on Computational Linguistics*, pages 1–7.

Chafe, W. (1980). *The Pear Stories: Cognitive, Cultural, and Linguistic Aspects of Narrative Production*. Ablex Pub. Corp., Norwood NJ.

Copestake, A. (2002). *Implementing Typed Feature Structure Grammars*. CSLI Publications, Stanford, CA.

Copestake, A., Flickinger, D., Pollard, C., and Sag, I. A. (2005). Minimal recursion semantics: An introduction. *Research on Language & Computation*, 3(2–3):281–332.

Crowgey, J. (ip). The syntactic exponence of sentential negation: A model for the LinGO grammar matrix. Master's thesis, University of Washington.

Drellishak, S. (2009). *Widespread But Not Universal: Improving the Typological Coverage of the Grammar Matrix*. PhD thesis, University of Washington.

Dzikovska, M. O. (2004). *A Practical Semantic Representation for Natural Language Parsing*. PhD thesis, University of Rochester, Rochester NY.

Ebert, K. (2003). Kiranti languages: an overview. In Thurgood, G. and LaPolla, R., editors, *The Sino-Tibetan languages*, chapter 31, pages 505–517. Routledge, London/New York.

Flickinger, D. (2000). On building a more efficient grammar by exploiting types. *Natural Language Engineering*, 6 (1) (Special Issue on Efficient Processing with HPSG):15 – 28.

Flickinger, D. (2011). Accuracy v. robustness in grammar engineering. In Bender, E. M. and Arnold, J. E., editors, *Language from a Cognitive Perspective: Grammar, Usage and Processing*, pages 31–50. CSLI Publications, Stanford, CA.

Goodman, M. W. (2012). Generation of machine-readable morphological rules from human-readable input. Unpublished ms, University of Washington.

Joshi, A., Levy, L. S., and Takahashi, M. (1975). Tree adjunct grammars. *Journal of Computer Systems Science*.

Kamei, S., Itoh, E., Fujii, M., Hirai, T., Saitoh, Y., Takahashi, M., Hiyama, T., and Muraki, K. (1997). Shareable formats and their supporting environments for exchanging user dictionaries among different MT systems as part of AAMT activities. In *MT Summit VI*.

Kaplan, R. M. and Bresnan, J. (1982). Lexical-Functional Grammar: a formal system for grammatical representation. In Bresnan, J., editor, *The Mental Representation of Grammatical Relations*. MIT Press.

King, T. H., Forst, M., Kuhn, J., and Butt, M. (2005). The feature space in parallel grammar writing. *Research on Language and Computation, Special Issue on Shared Representations in Multilingual Grammar Engineering*, 3(2):139–163.

Lønning, J. T., Oepen, S., Beermann, D., Hellan, L., Carroll, J., Dyvik, H., Flickinger, D., Johannessen, J. B., Meurer, P., Nordgård, T., Rosén, V., and Velldal, E. (2004). LOGON. A Norwegian MT effort. In *Proceedings of the Workshop in Recent Advances in Scandinavian Machine Translation*.

McConville, M. and Dzikovska, M. O. (2007). Extracting a verb lexicon for deep parsing from FrameNet. In *ACL 2007 Workshop on Deep Linguistic Processing*, pages 112–119, Prague, Czech Republic. Association for Computational Linguistics.

Michailovsky, B. (1994). Manner vs. place articulation in the kiranti initial stops. In Kitamura, H., Nishida, T., and Nagano, Y., editors, *Current issues in Sino-Tibetan linguistics*, pages 766–772. National Museum of Ethnology, Osaka.

Oepen, S. (2001). [incr tsdb()] — Competence and performance laboratory. User manual. Technical report, Saarland University, Saarbrücken, Germany.

Oepen, S., Flickinger, D., Toutanova, K., and Manning, C. D. (2004). LinGO Redwoods. A rich and dynamic treebank for HPSG. *Journal of Research on Language and Computation*, 2(4):575–596.

Pollard, C. and Sag, I. A. (1994). *Head-Driven Phrase Structure Grammar*. Studies in Contemporary Linguistics. The University of Chicago Press and CSLI Publications.

Rāī, N. K., Rāī, M., Paudyāl, N. P., Schikowski, R., Bickel, B., Stoll, S., Gaenszle, M., Banjade, G., Rāī, I. P., Bhaṭṭa, T. N., Sauppe, S., Rāī, R. M., Rāī, J. K., Rāī, L. K., Rāī, D. B., Rāī, G., Rāī, D., Rāī, D. K., Rāī, A., Rāī, C. K., Rāī, Ś. M., Rāī, R. K., Pettigrew, J., and Dirksmeyer, T. (2011). *Chintāṅa-Nepālī-Ãgrejī śabdakośa tathā vyākaraṇa*. Chintang Language Research Programme, Kāṭhmāḍauṁ.

Ranta, A. (2007). Modular grammar engineering in GF. *Research on Language & Computation*, 5:133–158.

Schikowski, R. (2012). Chintang morphology. Unpublished ms, University of Zürich.

Stoll, S., Bickel, B., Lieven, E., Banjade, G., Bhatta, T. N., Gaenszle, M., Paudyal, N. P., Pettigrew, J., Rai, I. P., Rai, M., and Rai, N. K. (2012). Nouns and verbs in chintang: children's usage and surrounding adult speech. *Journal of Child Language*, 39:284 – 321.

Suppes, P., Flickinger, D., Macken, B., Cook, J., and Liang, T. (2012). Description of the EPGY Stanford University online courses for mathematics and language arts. In *International Society for Technology in Education Annual (ISTE) 2012 Conference*, San Diego CA.

Quantifying Semantics Using Complex Network Analysis

Chris Biemann *Stefanie Roos* *Karsten Weihe*
Computer Science Department
Technische Universität Darmstadt
Hochschulstr. 10
64289 Darmstadt, Germany
{biem,weihe}@cs.tu-darmstadt.de, stefanie.roos@cased.de

ABSTRACT
Though it is generally accepted that language models do not capture all aspects of real language, no adequate measures to quantify their shortcomings have been proposed until now. We will use n-gram models as workhorses to demonstrate that the differences between natural and generated language are indeed quantifiable. More specifically, for two algorithmic approaches, we demonstrate that each of them can be used to distinguish real text from generated text accurately and to quantify the difference. Therefore, we obtain a coherent indication how far a language model is from naturalness.
Both methods are based on the analysis of co-occurrence networks: a specific *graph cluster measure*, the *transitivity*, and a specific kind of *motif analysis*, where the frequencies of selected *motifs* are compared. In our study, artificial texts are generated by n-gram models, for $n = 2, 3, 4$. We found that, the larger n is chosen, the narrower the distance between generated and natural text is. However, even for $n = 4$, the distance is still large enough to allow an accurate distinction.
The motif approach even allows a deeper insight into those semantic properties of natural language that evidently cause these differences: polysemy and synonymy.
To complete the picture, we show that another motif-based approach by Milo et al. (2004) does not allow such a distinction.
Using our method, it becomes possible for the first time to measure generative language models deficiencies with regard to semantics of natural language.

KEYWORDS: quantitative linguistics, network analysis, motif analysis, co-occurrence networks, language models.

Proceedings of COLING 2012: Technical Papers, pages 263–278,
COLING 2012, Mumbai, December 2012.

1 Introduction

Language models are used in many text processing systems (e.g. machine translation, document classification, language generation etc.) and are undoubtedly a standard building block of natural language processing. However, there exist hardly any methods that characterize language models quantitatively, in order to measure their deficiencies with respect to real language: the commonly used perplexity measure is known to be insensitive to semantic aspects (Chang et al., 2009). To this end, we propose an automatic approach that not only can distinguish language-model-generated text from real text, it also quantifies their distance. Moreover, we can quantify language model shortcomings with respect to two semantic phenomena of natural language, namely polysemy and synonymy. The ability to measure, to what extent language models in fact model these and other characteristics of real language, is a prerequisite for improving language models to closer adhere to the many-layered structure of natural language.

Before laying out the details of our approach, we would like to briefly sketch the general idea: If we accept that different language models capture natural language semantics to different extents, and if we had a measure that indicates and quantifies this extent, then this measure enables us to drive the development of language models towards a better reflection of semantics. We propose such a measure, which is based on the assumption that the structure of the co-occurrence graph of a (real or generated by a language model) text reflects semantic adequateness. Computing and comparing different characteristics of these co-occurrence graphs allows us to quantify the differences.

This paper summarizes a series of computational studies to characterize the uniqueness of co-occurrence networks from real natural language, as opposed to co-occurrence networks from artificial "natural" language. We developed two testing methods to decide whether a corpus of text was written by humans or generated by a language model. For that, we analyze the structural difference of the co-occurrence networks induced by real text and generated text, respectively. As generative language models, we chose the 2-gram, 3-gram, and 4-gram models.

We examine co-occurrence graphs.[1] In that, we consider two types of co-occurrence: sequential occurrence between neighbors and co-occurrence within a whole sentence. Our measures are based on two network metrics. Traditionally, the structures of language networks are analyzed either on the global level or on the level of single nodes and edges (e.g. the degree distribution). Our first method is based on a global clustering metric, the *transitivity*.

Our second method is based on *motif analysis*. This approach addresses an intermediate level, where a structural entity is composed of a small number of nodes and edges. Networks are compared by counting the number of times a certain *k-node motif*, i.e. a small graph of k nodes, appears as an induced subgraph. In case of the directed sequential co-occurrence graphs, 3-node motifs are analyzed (see Fig. 1), whereas the bidirectional sentence co-occurrence networks are compared based on their 4-node motifs (see Fig. 2).

In case of sentence-based co-occurrence, each of these methods enables us to distinguish natural from generated text and to quantify the differences in a reasonable way. Such a quantification can be viewed as an indicator how far a language model is from naturalness. In fact, it turns out that this indicator conforms well to what one would expect: higher-order n-gram models generate a better approximation of real texts. However, no n-gram model is able to capture long-

[1] In accordance with the established terminology, we will use the terms *graph* and *network* synonymously, and choose the appropriate word in view of the respective context.

range semantic dependencies well, which we will exploit in our analysis techniques. Moreover, we will show that motifs can be related to specific semantic properties of natural languages that do not occur in *n*-gram generated text and hence explain the observed differences to a large extent. Here, the above-mentioned domain-specific phenomena are of a semantic kind. In particular, we show that two specific phenomena, *polysemy* and *synonymy*, are reflected by the counts of two motifs, the *chain motif* and the *box motif* (#2 and #4 in Fig. 2).

In summary, the presented work follows a new, successful, path and opens new, promising, perspectives on the analysis of language models. This is the first application of motif analysis to language networks and their underlying semantics. So, besides the specific computational results and the novel method to obtain them, the presented work is also relevant due to this general methodological innovation.

The paper is organized as follows. In Section 2, we briefly review the most relevant related work. Then, in Section 3, we will describe our approach in full detail. In Section 4, we present and discuss the conjectures that structure our research. In Section 5, we present our results, and finally discuss future perspectives in Section 6.

2 Related Work

2.1 Network Analysis

There is a large scientific body of methods and applications of network analysis (Aggarwal and Wang, 2010; Aggarwal, 2011). Graph mining – the art of detecting and analyzing patterns and structures in graphs – is the specific focus of the surveys (Cook and Holder, 2006; Fortunato, 2010).

It seems reasonable to classify network analysis techniques by the level of granularity they address. Elementary statistical measures such as the node degree distribution operate on the level of single nodes and edges. In the opposite extreme case, on the global level, the structure of a network is captured in a single (scalar) numerical value. Examples for global measures are the average shortest path length, the diameter, as well as simple characteristics such as node and edge count. See the above-mentioned surveys for a systematic discussion.

For our analysis, a global clustering metric, the *transitivity*, is considered, however, our main focus is on *motif analysis*. *Motif analysis* addresses an intermediate level: local structures consisting of a small number of nodes and edges. Networks are compared by comparing the number of occurrences of selected motifs.

Motif analysis has first been investigated in computational biology (Shen-Orr et al., 2002) and has since been applied to a variety of network types in biology and biochemistry (Schreiber and Schwöbbermeyer, 2010). The underlying insight is that biological and biochemical dynamics are statistically related to the occurrence of small functional blocks, which have specific structures. This insight is well captured by motif signatures, and in fact, many computational studies reveal significant relations. Due to this success, it did not take long time until this technique has been applied to networks from other domains. For example, (Milo et al., 2002, 2004) compare networks from biology, electrical engineering, natural language and computer science and find that the motif signatures from different domains are so different that they may serve as a "fingerprint" of the respective domain.

The idea of functional blocks applies in domains beyond biology and biochemistry as well, surprisingly, even in social networks. In (Krumov et al., 2011), we analyzed citation networks,

which we modeled as undirected graphs on the authors. An edge indicates at least one joint publication. In a sense, the citation numbers of individual publications within an occurrence of a motif can be aggregated to a citation number of the entire occurrence. We considered four natural ways for aggregation. Roughly speaking, the main result of (Krumov et al., 2011) is this: the average citation number of the box motif, taken over all occurrences, is statistically significantly larger than expected. This effect occurs for all four ways of aggregation. A deeper look revealed that certain occurrences of the *box motif* (#4 in Fig. 2) explain this result: two "seniors," A and B, have jointly published, A has published with a "junior" C, B with a junior D, and C and D have joint publications as well, but neither A with D nor B with C. Among these occurrences, the ones that serve as "bridges" in the network in a certain sense are particularly responsible for the observed effect.

In view of the outlook (Section 6), we further mention recent work that uses the concept of motifs for other purposes than network analysis. (Krumov et al., 2010a,b) developed an algorithm to optimize the structure of peer-to-peer networks based on local operations only. Each node manipulates the local structure in its vicinity in order to thrive the local motif signature towards the average local motif signature of an optimal network.

2.2 Complex Networks of Natural Language

The structure of natural language networks has been extensively investigated, see e.g. (Masucci and Rodgers, 2006) and references therein.

(Ferrer-i-Cancho and Solé, 2001) have shown that co-occurrence networks of natural language are scale-free small world graphs. Whereas scale-freeness seems to be a consequence of the Zipfian word-frequency distribution (Biemann, 2007), Steyvers and Tenenbaum (2005) find the small-world property in co-occurrence networks and lexical-semantic resources, which indicates that co-occurrence networks reflect semantic properties.

There is only very little work on operationalizing complex network analysis for natural language processing applications. (Pardo et al., 2006) evaluate the quality of automatic summaries by analyzing the degree distributions of networks generated from words at different fixed offsets in the text, and (Amancio et al., 2012) characterize texts for authorship attribution by quantifying their consistency index, which is measured by the number of authors that use content words in a sequence. A related work is (Köhler and Naumann, 2010), where segments of words with increasing length and frequency are used to characterize texts of different authors.[2] We are not aware of any other research that uses network analysis to assess the quality of language models trained from real text.

3 Methodology

Our results have been generated in a three-step process: First, the text needs to be selected, respectively generated, before the graphs can be derived from the texts according to a parameterizable strategy. In the last step, our proposed metrics are evaluated on these graphs. These three steps are explained in detail in Sects. 3.1–3.3, respectively.

[2]**Note:** (Köhler and Naumann, 2010) also use the term 'motifs', but they refer to the aforementioned sequences of words, not to subgraphs as in our work.

3.1 Text Basis

Text corpora For our experiments, we use corpora of different languages of one million sentences each, provided by LCC[3] (Biemann et al., 2007). We use the same corpus of real language for training the n-gram model and for comparison. For comparison between real and generated language, we generate text according to the same sentence length (number of tokens) distribution as found in the respective real language corpus, since we have found in preliminary experiments that co-occurrence network structure is dependent on the sentence length distribution. We have found in further experiments, that the general picture of results is stable for corpora of different sizes, starting from about ten thousand sentences.

Text generation with n-gram models For the scope of this work, we chose n-gram models, which are the standard workhorses of language modeling. A language model assigns a probability to a sequence of words, based on a probabilistic model of language. This can be used to pick the most probable/fluent amongst several alternatives, e.g. in a statistical machine translation system (Koehn, 2010). An n-gram language model (cf. (Manning and Schütze, 1999)) over sequences of words is defined by a Markov chain of order $(n-1)$, where the probability of the next word only depends on the $(n-1)$ previous words, and the probability of a sentence is defined as $P(w_1...w_k) = \prod_{i=1..k} P(w_i|w_{i-1}..w_{i-n+1})$. We add special symbols, BoS and EoS, to indicate sentence beginning and end. Then we generate sentences word by word, starting from a sequence of $(n-1)$ BoS-symbols, according to the probability distribution over the vocabulary. As soon as the EoS symbol is generated, we generate the next sentence. Probabilities are initialized by training on the respective corpus of real text (see above) from the relative counts, i.e. $P(w_i|w_{i-1}..w_{i-n+1}) = count(w_i..w_{i-n+1})/count(w_{i-1}..w_{i-n+1})$. In our study, we used n-gram models with $n \in \{2,3,4\}$ (in some contexts, we additionally consider $n = 1$ for completion).

Shortcomings of n-gram models are obvious: no long-range relations are modeled explicitly, thus n-gram models produce locally readable but semantically incoherent text. This study is, to our knowledge, the first attempt to quantify this phenomenon. Despite their simplicity, n-gram models still excel in NLP applications (cf. (Ramabhadran et al., 2012)).

In NLP applications, n-gram models are usually subject to smoothing and back-off techniques (cf. (Manning and Schütze, 1999)). Smoothing is necessary to account for unseen words, which is not an issue for generation. We only present results for language models without back-off in this work, although we did some experiments with texts generated from n-gram models with back-off estimated through deleted estimation. Note that we found no substantial differences to text generated without back-off.

3.2 Network Construction

Next, we describe the construction of a complex network from a text corpus of (real or generated) language. The nodes of the derived graphs correspond to the m most frequent words in the considered text. An edge from node A to B exists if the word corresponding to A *co-occurs*, i.e. occurs together in a well-defined context, with the word corresponding to B significantly often. Different kinds of co-occurrence contexts are considered, as well as significance thresholds and graph sizes m.

[3] see `http://corpora.informatik.uni-leipzig.de/`

Network size The number of nodes m in the graph, corresponding to the most frequent words in the considered text, was set to be $5,000$, as to match the commonly assumed size of the core vocabulary of a language (Dorogovtsev and Mendes, 2001). In preliminary experiments, we have verified that, qualitatively, our results are stable across vocabulary sizes between $1,000$ and $20,000$, as long as the *most frequent* words are considered.

Co-occurrence contexts We consider two different kinds of contexts: co-occurrence as immediate neighbors in a sequence, and co-occurrence within a sentence (sequences as limited by BoS and EoS). Thus, for each corpus of text, composed of sentences, we can compute the co-occurrence graph by connecting word nodes with edges, if words co-occur. Edges are directed in the case of neighbor-based co-occurrence, and undirected for the sentence-based case. It is known (Biemann et al., 2004) that sentence-based co-occurrences, besides capturing collocations, often reflect semantic relations and capture topical dependencies between words.

Significance threshold Since mere co-occurrence results in a large number of edges and very dense networks, we apply a significance test that measures the deviation of the actual co-occurrence frequency from the co-occurrence frequency that would have been observed if the two co-occurring words would be distributed independently. Here, we use the log-likelihood test (Dunning, 1993) to prune the network: We only draw edges between word nodes, if the words co-occur with a significance value above a certain threshold. For our experiments, we used a threshold of 10.83[4] . During preliminary experiments, we have found the reported results to be stable across a wide range of significance thresholds. See (Biemann and Quasthoff, 2009) for an analysis of global properties of significant co-occurrence graphs of natural language. The co-occurrence graph was computed using the TinyCC[5] corpus production engine (Quasthoff et al., 2006).

3.3 Network Analysis

Transitivity Let $G = (V, E)$ be an undirected graph. A *closed triangle* is a set of three nodes such that all three possible edges do exist. On the other hand, a *triple* is any set of three nodes and two edges (in other words, a chain of two edges). The *transitivity* of $T(G)$ of G is three times the total number of closed triangles divided by the total number of triples, as defined by (Newman et al., 2002). This can be calculated by iterating over every node v and counting the triangles and triples in which v is incident to two edges:

$$T(G) = \frac{\sum_{v \in V} \delta(v)}{\sum_{v \in V} \binom{k(v)}{2}} \tag{1}$$

with $\delta(v) = |\{u, v, w\} \in V, \{\{u, w\} \in E \text{ and } \{v, u\} \in E \text{ and } \{v, w\} \in E\}|$, and $k(v)$ the degree of v.

Motif analysis A k-node motif is a small connected graph of k nodes. An *occurrence* of a motif M in a network $G = (V, E)$ is a set $V' \subseteq V$ of nodes such that the subgraph of G induced by V' is isomorphic to M.[6] For a set of motifs, the *motif signature* is the vector of number of instances

[4] which corresponds to an error level of 0.1% of falsely rejecting the hypothesis that words co-occur independently

[5] available for download at `http://wortschatz.uni-leipzig.de/~cbiemann/software/TinyCC2.html`

[6] For a graph $G = (V, E)$ and a node set $V' \subseteq V$, the *subgraph* of G *induced* by V' is the unique graph (V', E'), where $E' \subseteq E$ is the set of all edges of E with both endnodes in V'. Note that this really means *all* edges. In fact, if E' is only required to contain *some* of the edges with both endnodes in V', (V', E') is usually called just a subgraph, not *the*

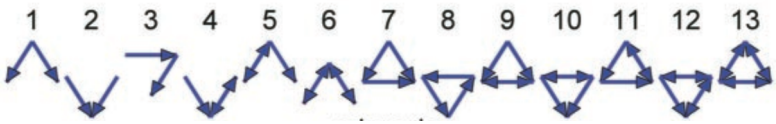


Figure 1: Directed motifs of size 3 as used in (Milo et al., 2004)

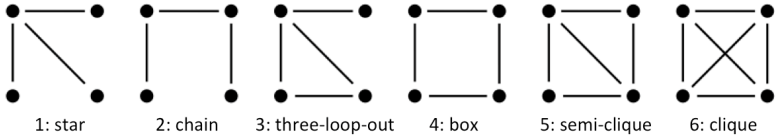


Figure 2: Undirected motifs of size 4 with names used throughout this paper

of each motif in G (typically, but not exclusively, all motifs in the set have the same number of nodes). Alternatively, the fraction of each motif to the total number of motifs is frequently considered, i.e. for m motifs M_1,...,M_m with counts $c(M_1)$,...,$c(M_m)$, the *motif signature* is the vector $(s(M_1), ..., s(M_m)$ with

$$s(M_i) = \frac{c(M_i)}{\sum_{j=1}^{m} c(M_j)}$$

To compare graphs of different sizes and edge counts, we generally present fractions instead of absolute counts. Our results are mainly presented in the form of xy-diagrams, mapping the motif to the corresponding frequency. Due to the high diversity in frequency, a logarithmic scale is used on the y axis. For presentational purposes and in accordance with the literature, we connect the dots in the plots, although they represent discrete values.

Throughout this paper, we consider two kinds of motifs, directed 3-node motifs (see Figure 1) and undirected 4-node motifs (see Figure 2). For directed and undirected graphs, respectively, these are the smallest meaningful motifs for our purposes. In fact, undirected 3-node motifs are triangles and triples and, hence, implicitly covered by our transitivity analysis. The motif counts were computed efficiently with the MotifAnalysis[7] software.

4 Conjectures

In the following, the term *(network) parameter* can refer to either the transitivity or a component of the motif signature.

Distinction We conjecture that a significant difference between the network parameters from real and generated text can be observed. Informally speaking, this conjecture is affirmatively substantiated within the scope of our computational studies if there is at least one network parameter such that for each language, the three values from generated text are on the same side of the value from natural text (either all smaller or all larger), and the distance between the former three values and the latter value is substantial.

induced subgraph.

[7] available for download at `https://github.com/stef-roos/MotifAnalysis`

Quantification We conjecture that the network parameters induce a reasonable quantitative measure how far a language model is from naturalness. In our studies, the investigated language models are the 2-gram, 3-gram, and 4-gram models. It is quite reasonable to say, the larger n is, the closer the n-gram model is to naturalness. Therefore, the conjecture is affirmatively substantiated within the scope of our computational studies, if we find at least one network parameter whose values for the n-gram models show a strictly monotonous convergence behavior towards the value for natural language, and this behavior is consistent throughout all languages.

Relation to semantics We conjecture that some of the motifs substantiate the first two conjectures, and that these motifs also allow a deeper insight into the semantic reasons for the observed differences. This conjecture is affirmatively substantiated within the scope of our computational studies, if we can identify semantic phenomena that (1) occur in natural text more often than in generated text and (2) significantly increase respectively decrease the number of occurrences of some motifs.

Relation to local syntax We conjecture that the motif profile of the neighbor-based graph does reflect local syntactic dependencies. A comparison of the motif profiles of the neighbor-based graphs should quantify the extent, to which language models capture local syntactic dependencies between adjacent words. Since n-gram models are trained on short word sequences, we conjecture that there is no difference between real and n-gram generated text with respect to local syntax for $n > 1$.

5 Results

With regard to sentence co-occurrence, the computational results in Sections 5.1 and 5.2 confirm that transitivity fulfills the first and second conjecture, and motif analysis fulfills all three conjectures. The local syntactic conjecture, based on neighborhood co-occurrence, is proven valid in our computational studies, as detailed in Section 5.3.

5.1 Distinction and Quantification

Transitivity Table 1 shows the transitivity values of the sentence-based co-occurrence networks for six languages, in each case for the 2-gram, 3-gram, and 4-gram models and for real natural language. The gap between natural text and any generated text is nowhere smaller than 15%. This substantiates the first conjecture. Evidently, the values for the n-gram models converge strictly monotonously towards the value of natural language in each case, which substantiates the second conjecture.

As an explanation, we attribute this to missing links in n-gram networks that result from the deficiency of such models to capture semantic coherence. The linguistic interpretation of transitivity is the following: if two words A and B co-occur significantly, and A occurs significantly with a third word C, what is the probability that B and C also co-occur significantly? Semantic cohesion (Halliday and Hasan, 1976) means that a text, thus a sentence, is about a certain topic, and there are several sentences that refer to the same topic in corpora. Topics manifest themselves in a certain set of words that will be used frequently together to describe this topic, which results in cliques in the co-occurrence network. While n-gram models capture topical relations between words if they co-occur within a short distance, they miss semantic relations between words that occur at long distances.

Clique motif The differences in the relative shares of the clique motif (#6 in Fig. 2) are even stronger. In fact, Table 2 shows the relative number of cliques in n-gram generated text normalized by the relative number of cliques in real text. The gap between natural text and any generated text is dramatic for $n = 2, 3$ and still always greater than 38% for $n = 4$. The Lithuanian language is the only exception to monotonous convergence. However, even for this language, the discrepancy is greater than 35% for $n = 3$. Except for the Lithuanian language, strictly monotonous convergence is evident, and most of the convergence steps are quite large. In summary, the first conjecture is completely substantiated by this particular motif as well, and the second conjecture is substantiated to a very large extent.

Language	**Real**		**2-gram**		**3-gram**		**4-gram**	
	$T(G)$	rel. $T(G)$	$T(G)$	rel. $T(G)$	$T(G)$	rel. $T(G)$	$T(G)$	rel. $T(G)$
English	0.1533	1.0	0.0729	0.4757	0.0886	0.5781	0.0937	0.6111
German	0.1255	1.0	0.0700	0.5573	0.0841	0.6701	0.1057	0.8420
French	0.1468	1.0	0.0652	0.4440	0.0773	0.5263	0.1047	0.7133
Indonesian	0.1789	1.0	0.0883	0.4936	0.1227	0.6858	0.1479	0.8263
Farsi	0.2143	1.0	0.0764	0.3565	0.1116	0.5207	0.1557	0.7265
Lithuanian	0.1615	1.0	0.0893	0.5530	0.1216	0.7529	0.1289	0.7981

Table 1: Transitivity $T(G)$ in absolute and relative terms to real language networks for six languages, comparing networks from real text with networks from n-gram generated text

To exemplify this, the closed neighborhood graph of "Monday" for its 20 most significant co-occurrences, which is the subgraph consisting of all edges involving "Monday" and the edges between all involved nodes, is depicted in Figure 3 for our real and n-gram networks of English. While collocations like "Monday evening" are present in all graphs, words like "Football" and "Saturday" do not get connected in the 3-gram graph: although they are generated significantly frequently with "Monday", this happens in different generated sentences, whereas they significantly co-occur in real language. Further, the density of these graphs is monotonically increasing with n and highest for real language.

5.2 Semantic Conjecture

Recall the concept of *functional blocks* from Section 2. Next we will show that, in our context here, the chain and the box motif are functional blocks in quite an analogous sense.

Figure 4 shows the motif profiles of networks on a log-scale. In Fig. 4 (left upper), we depict the

Language	**Real**		**2-gram**		**3-gram**		**4-gram**	
	abs	rel	abs	rel	abs	rel	abs	rel
English	0.1090	1.0	0.0339	0.3113	0.0387	0.3548	0.0407	0.3734
German	0.0735	1.0	0.0321	0.4364	0.0342	0.4659	0.0437	0.5944
French	0.1002	1.0	0.0192	0.1902	0.0284	0.2841	0.0484	0.4838
Indonesian	0.1706	1.0	0.0336	0.1971	0.0672	0.3939	0.1041	0.6104
Farsi	0.2668	1.0	0.0250	0.0937	0.0492	0.1843	0.1107	0.4149
Lithuanian	0.1755	1.0	0.0474	0.2699	0.1139	0.6487	0.1078	0.6143

Table 2: Percentage of clique motifs (#6) in absolute (abs) and relative (rel) terms to real language for six languages, comparing networks from real text with networks from n-gram generated text

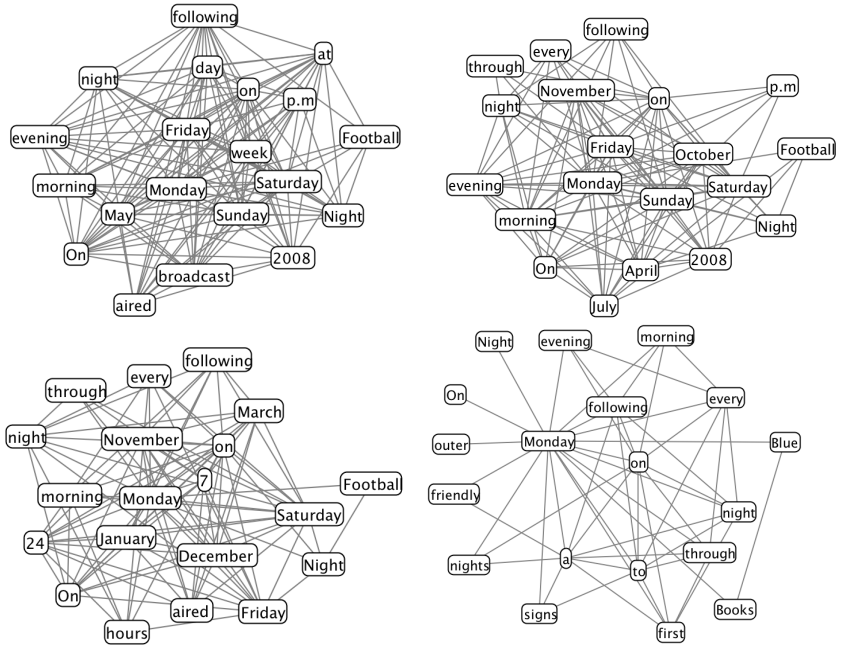


Figure 3: Neighborhood graphs of "Monday" in the English networks for real (upper left), 4-gram (upper right), 3-gram (lower left) and 2-gram (lower right) language, which exemplify the deficiency of n-gram models to capture long-range semantic relations

motif profiles for English networks of real and generated language for $n = 1, 2, 3, 4$; The other plots in Fig. 4 shows the profiles for all other languages for $n = 2, 3, 4$. It is clearly visible that real language networks exhibit fewer star (#1) motifs and a higher amount of all other motifs. Differences for the chain (#2) and the box (#4) motifs are especially pronounced. Examining instances of these motifs more closely, we are able to link these differences to properties of natural language semantics, which will be explained more thoroughly in the remainder.

Polysemy and chain motif *Semantic polysemy* refers to the phenomenon that a word, denoted as a string of characters, can have different denotations in different contexts, e.g. "board" as an assembly or a piece of wood. In real sentences, words are not co-occurring at random, but usually revolve around a certain topic. Thus, it is not likely to find the word "wood" in a sentence that talks about a "board of directors", and sentences about wooden planks usually do not contain the word "chairman".

In co-occurrence networks, polysemy leads to chains: ambiguous words connect words that are not connected to each other, and act as a bridge between different topical word clusters. In a chain of length four, one more word from a topical cluster is observed, which does not connect to the polysemous word since it seems that their occurrences are deemed rather independent

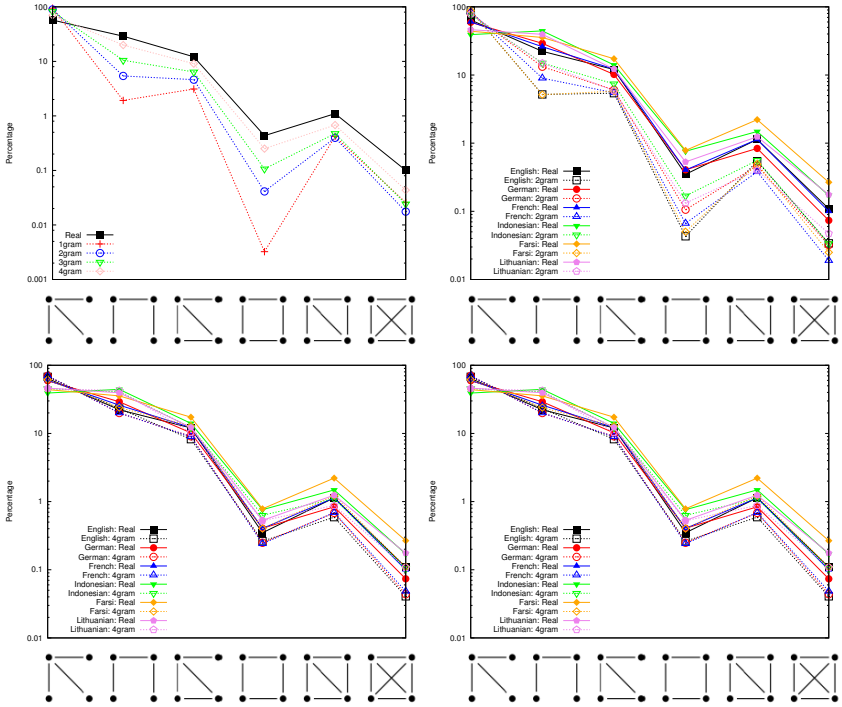


Figure 4: Motif profiles for real and generated text networks based on sentence co-occurrence. Upper Left: motif signature for English, comparing real language to $n = 1, 2, 3, 4$. Upper Right: Comparison Real vs. 2-gram for six languages. Bottom Left: Comparison Real vs. 3-gram for six languages. Bottom Right: Comparison Real vs. 4-gram for six languages. High congruence across languages of different language families demonstrates language independence of our analysis method

by the significance measure.

Enumerating the chain motif instances of the English real network, we exemplify this point with the following chains (the ambiguous word is emphasized in each line):

- total - km^2 - **square** - root
- Democrats - **Social** - Sciences - Arts
- Number - **One** - Formula - Championship
- Abraham - **Lincoln** - Nebraska - Iowa

N-gram models are oblivious to these sense distinctions. Thus, nothing prevents e.g. a 3-gram model from generating e.g. a sequence "Abraham Lincoln , Nebraska" with high likelihood,

confusing the two senses of "Lincoln" as a last name and a city. In the co-occurrence network, this can result in a connection between "Abraham" and "Nebraska", which decreases the chain motif count. The remaining chains of *n*-gram networks, on the other hand, consist mostly of highly frequent words that occur next to each other, e.g. "slowly started on finals", "personal taste good advice". These are also present in the real language network. We observe a much smaller number of chains formed of words of lower frequencies in *n*-gram generated text. Note that it is neither the case that all polysemous words cause chains, nor do all words in the central positions of a chain exhibit lexical ambiguity – differences in chain motif counts rather quantitatively measure the amount of such polysemy than qualify as an instrument to find single instances.

Hence, the lower amount of chain motifs can be explained by the creation of links that are not present in real language. On the first glance, this should lead to a higher clustering, contradictory to the results for motifs #3,5,6. However, as explained in Section 5.1, the clustering of *n*-grams is drastically lower than for real language. Although some of the 4-nodes sets that represent boxes or chains in real languages form (semi-)cliques in *n*-grams, instances of motif #3,5,6 in real languages are replaced by stars in *n*-gram graphs more frequently.

From these observations it becomes very clear that chain motif counts reflect polysemy. The lower *n* is chosen in the generating *n*-gram model, the smaller is the modelling context for ambiguities, resulting in lower chain motif counts.

Synonymy and box motif *Synonymy* means that different words refer to the same concept. Two words are synonyms if they can be used interchangeably without changing the meaning, but there are also rather syntactic variants of words that refer to the same concept, such as nominalizations of adjectives or verb forms of different inflections.

In natural language, the principle of *parsimony* leads to the effect that the same concept is rarely referred to several times in the same sentence. In fact, synonyms usually do not co-occur, but they share a large number of significant co-occurrences – an observation that leads to the operationalization of the distributional hypothesis (Miller and Charles, 1991). When two such concepts are mentioned together frequently since they belong to the same topic, this leads to box motifs, as the following examples from the English real language network illustrate:

- - Ancient - Greek - ancient - Greece -
- - winning - award - won - price -
- - Ph.D - his - doctorate - University -
- - said - interview - stated - " -
- - wrote - articles - published - poems -

We observe different kinds of word pairs for the same concept: synonyms like (award, price), same word stem within or across word classes like (winning, won) or (Greek, Greece), and artifacts of punctuation or spelling (ancient, Ancient) or (interview, "), Thus, box motifs capture a very loose notion of synonymy: "interview" and the double quote " e.g. both refer to a (indirect or direct) speech act.

Again, *n*-gram models are not aware of concepts and references to them, so there is no mechanism that prevents the *n*-gram model from generating sentences that refer to the same concept several times or even use the same word repeatedly. This possibly results in a connection

between those pairs, reducing the box motif count. Box motifs that can be found in both real and n-gram language are again resulting from local sequences of highly frequent words that are possibly circular, not necessarily from the same contexts. Examples include "desktop cover art background", "hall nearby on church" and "these will ask why".

These observations lead to the conclusion that synonymy of natural language leads to box motifs in sentence-based co-occurrence networks, and the difference in the box motif count quantifies the amount of capability of the language model to inhibit the generation of words that refer to concepts which already have been mentioned. N-gram models have this capability only for a very limited context, which again increases with higher n.

5.3 Local Syntactic Conjecture

To assess whether n-gram models really grasp local syntactic dependencies, we examine the motif profile of directed motifs of size 3 on the neighbor-based co-occurrence graph as described in Section 3. In this, we follow Milo et al. (2004), who unfortunately do not specify their procedure of graph construction from language in detail. Figure 5 shows the directed motif profiles of real text and 2-gram-generated text for four languages, and for English for $n = 2, 3, 4$. As Milo et al. (2004) observe, different languages have a very similar motif profile, and the corresponding graphs belong to the same superfamily of networks, i.e. the mostly bipartite networks. We also observe that there is no systematic difference between the neighbor-based networks of real language and generated language, even for $n = 2$. This substantiates our fourth conjecture: n-gram models do in fact grasp local syntactic dependencies very well.

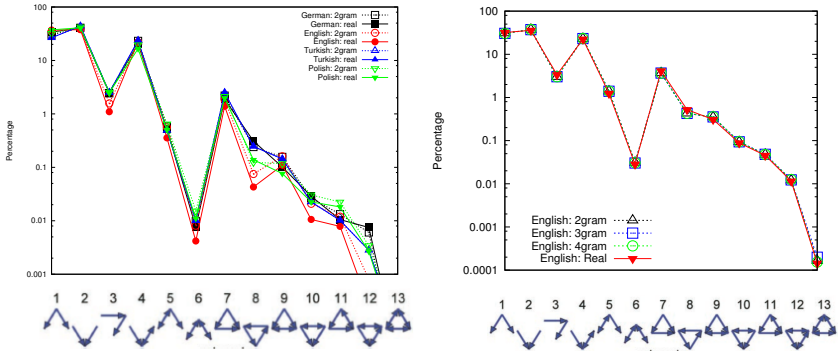


Figure 5: Motif profile of the directed neighbor-based co-occurrence graphs for real and 2-gram generated language (left), respectively for English with n-gram generated text for $n = 2,3,4$ (right), showing a high inter-language agreement and the inability of this measure to distinguish real from generated language

This renders the result of Milo et al. (2004) as not being very influential for natural language research, since the neighbor-based motif profile of real language can be generated by a highly deficient 2-gram language model.

6 Conclusion and Outlook

The methods presented in this paper open new ways to the assessment of the quality of language models, as reflected in the first and second conjecture. We showed that both a global graph metric, transitivity, and an intermediate metric, motif signatures, quantify the difference between natural and generated language. With our third conjecture, we go even one step further by presenting a way to relate semantics to network structure. Looking at the motif signature of real and generated language, we can identify differences due to polysemy and synonymy, which are not adequately modeled by *n*-grams.

Our analysis builds on the fact that generation with language models is not tied to any target application, and generative language models that do not have mechanisms to ensure cohesion will fail to show the same patterns as real language, especially regarding semantic properties such as synonymy and polysemy. In applications like Machine Translation, where language models are used to rank alternatives rather than free generation and are thus bound to the cohesive structure of the source language text, the shortcomings discussed in this paper do not necessarily impede the performance on the task. In fact, preliminary experiments involving comparisons of real translations with automatic translations of the same text did not result in motif profile differences. However, as e.g. (Tan et al., 2012) point out, there is a need for more coherent language models even in these applications: e.g. speech-to-text in noisy environments might greatly benefit from better language models.

Our computational studies with regard to co-occurrence graphs based on sentences and neighboring words indicate that language models based on *n*-grams reflect local syntax well, but fail to model semantic cohesion and topicality. Further, these language models do not have means of regulation for referring to the same concept several times. While these results confirm the common intuition of *n*-grams, we present the first study to actually quantify this deficiency.

The presented series of experiments are but a first step towards a more systematic analysis of the relation between the global characteristics of language and the structure of co-occurrence networks. Varying the notion of co-occurrence, for example, to involve positional offsets, could possibly unveil grammatical differences for a quantitative typology of languages. Further work should include more sophisticated language models such as the syntactic topic model (Boyd-Graber and Blei, 2008), which explicitly models topicality. The restriction to a subset of word classes, for example, nouns or verbs, or to class-based *n*-gram models (Brown et al., 1992) may also be interesting.

All of these ideas still address the *analysis* of language models. As mentioned in Section 2, the concept of motifs has recently been used for a constructive purpose. We anticipate that this change of perspective is also promising in the realm of language networks and may well guide the design of new, semantically more adequate, language models.

References

Aggarwal, C. C. (2011). *Social Network Data Analytics*. Kluwer.

Aggarwal, C. C. and Wang, H. (2010). Managing and mining graph data. *Database*, 40:487–513.

Amancio, D., Oliveira Jr., O., and da Costa. L.F. (2012). Using complex networks to quantify consistency in the use of words. *Journal of Statistical Mechanics: Theory and Experiment*, 2012(01):P01004.

Biemann, C. (2007). A random text model for the generation of statistical language invariants. In *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; Proceedings of the Main Conference*, pages 105–112, Rochester, New York.

Biemann, C., Bordag, S., and Quasthoff, U. (2004). Automatic acquisition of paradigmatic relations using iterated co-occurrences. In *Proceedings of LREC 2004*, Lisbon, Portugal.

Biemann, C., Heyer, G., Quasthoff, U., and Richter, M. (2007). The Leipzig Corpora Collection - monolingual corpora of standard size. In *Proceedings of Corpus Linguistic 2007*, Birmingham, UK.

Biemann, C. and Quasthoff, U. (2009). Networks generated from natural language text. In Ganguly, N., Deutsch, A., and Mukherjee, A., editors, *Dynamics On and Of Complex Networks*, Modeling and Simulation in Science, Engineering and Technology, pages 167–185. Birkhäuser Boston.

Boyd-Graber, J. and Blei, D. M. (2008). Syntactic topic models. In *Neural Information Processing Systems*.

Brown, P. F., Pietra, V. J. D., deSouza, P. V., Lai, J. C., and Mercer, R. L. (1992). Class-Based n-gram Models of Natural Language. *Computational Linguistics*, 18(4):467–479.

Chang, J., Boyd-Graber, J., Wang, C., Gerrish, S., and Blei, D. M. (2009). Reading tea leaves: How humans interpret topic models. In *Neural Information Processing Systems*.

Cook, D. J. and Holder, L. B. (2006). *MIning Graph Data*. Wiley.

Dorogovtsev, S. N. and Mendes, J. F. F. (2001). Language as an evolving word web. *Proceedings of The Royal Society of London. Series B, Biological Sciences*.

Dunning, T. (1993). Accurate methods for the statistics of surprise and coincidence. *Computational Linguistics*, 19(1):61–74.

Ferrer-i-Cancho, R. and Solé, R. V. (2001). The small world of human language. *Proceedings of The Royal Society of London. Series B, Biological Sciences*, 268(1482):2261–2265.

Fortunato, S. (2010). Community detection in graphs. *Physics Reports*, 486:75–174.

Halliday, M. A. K. and Hasan, R. (1976). *Cohesion in English*, volume 1 of *English Language Series*. Longman.

Koehn, P. (2010). *Statistical Machine Translation*. Cambridge University Press, New York, NY, USA, 1st edition.

Köhler, R. and Naumann, S. (2010). A syntagmatic approach to automatic text classification. statistical properties of F- and L-motifs as text characteristics. In Grzybek, P., Kelih, E., and Mačutek, J., editors, *Text and Language*, pages 81–89. Praesens Verlag, Vienna.

Krumov, L., Andreeva, A., and Strufe, T. (2010a). Resilient peer-to-peer live-streaming using motifs. In *11th IEEE World of Wireless, Mobile and Multimedia Networks (WoWMoM)*, pages 1–8.

Krumov, L., Fretter, C., Müller-Hannemann, M., Weihe, K., and Hütt, M.-T. (2011). Motifs in co-authorship networks and their relation to the impact of scientific publications. *European Physical Journal B*, 84(4):535–540.

Krumov, L., Schweizer, I., Bradler, D., and Strufe, T. (2010b). Leveraging network motifs for the adaptation of structured peer-to-peer-networks. In *GLOBECOM*, pages 1–5.

Manning, C. D. and Schütze, H. (1999). *Foundations of statistical natural language processing*. MIT Press, Cambridge, MA, USA.

Masucci, A. P. and Rodgers, G. J. (2006). Network properties of written human language. *Phys. Rev. E*, 74:026102.

Miller, G. A. and Charles, W. G. (1991). Contextual Correlates of Semantic Similarity. *Language and Cognitive Processes*, 6(1):1–28.

Milo, R., Itzkovitz, S., Kashtan, N., Levitt, R., Shen-Orr, S., Ayzenshtat, I., Sheffer, M., and Alon, U. (2004). Superfamilies of evolved and designed networks. *Science*, 303(5663):1538–1542.

Milo, R., Shen-Orr, S., Itzkovitz, S., Kashtan, N., Chklovskii, D., and Alon, U. (2002). Network Motifs: Simple Building Blocks of Complex Networks. *Science*, 298(5594):824–827.

Newman, M. E. J., Watts, D. J., and Strogatz, S. H. (2002). Random graph models of social networks. *Proc. Natl. Acad. Sci. USA*, 99(suppl. 1):2566–2572.

Pardo, T. A. S., Antiqueira, L., Nunes, M. G. V., Oliveira Jr., O. N., and da F. Costa, L. (2006). Using complex networks for language processing: The case of summary evaluation. In *Proceedings of the International Conference on Communications, Circuits and Systems (ICCCAS'06) - Special Session on Complex Networks*, pages 2678–2682, Gui Lin, China. UESTC Press.

Quasthoff, U., Richter, M., and Biemann, C. (2006). Corpus portal for search in monolingual corpora. In *Proceedings of the fifth international conference on Language Resources and Evaluation, LREC*, pages 1799–1802, Genoa, Italy.

Ramabhadran, B., Khudanpur, S., and Arisoy, E., editors (2012). *Proceedings of the NAACL-HLT 2012 Workshop: Will We Ever Really Replace the N-gram Model? On the Future of Language Modeling for HLT*. Montréal, Canada.

Schreiber, F. and Schwöbbermeyer, H. (2010). *Statistical and Evolutionary Analysis of Biological Network Data*, chapter Motifs in biological networks, pages 45–64. Imperial College Press/World Scientific.

Shen-Orr, S., Milo, R., Mangan, S., and Alon, U. (2002). Network motifs in the transcriptional regulation network of Escherichia coli. *Nature genetics*, 31(1):64–68.

Steyvers, M. and Tenenbaum, J. B. (2005). The Large-Scale Structure of Semantic Networks: Statistical Analyses and a Model of Semantic Growth. *Cognitive Science*, 29(1):41–78.

Tan, M., Zhou, W., Zheng, L., and Wang, S. (2012). A scalable distributed syntactic, semantic, and lexical language model. *Computational Linguistics*, 38(3):631–671.

Improvements to Training an RNN parser

Richard J. BILLINGSLEY *James R. CURRAN*
School of Information Technologies
University of Sydney
NSW 2006, Australia
{richbill,james}@it.usyd.edu.au

ABSTRACT
Many parsers learn sparse class distributions over trees to model natural language. Recursive Neural Networks (RNN) use much denser representations, yet can still achieve an F-score of 92.06% for right binarized sentences up to 15 words long. We examine an RNN model by comparing it with an abstract generative probabilistic model using a Deep Belief Network (DBN). The DBN provides both an upwards and downwards pointing conditional model, drawing a connection between RNN and Charniak type parsers, while analytically predicting average scoring parameters in the RNN. In addition, we apply the RNN to longer sentences and develop two methods which, while having negligible effect on short sentence parsing, are able to improve the parsing F-Score by 0.83% on longer sentences.

KEYWORDS: Parsing Recurrent Neural Network Restricted Boltzmann Machine.

Proceedings of COLING 2012: Technical Papers, pages 279–294,
COLING 2012, Mumbai, December 2012.

1 Introduction

Fast and accurate constituent parsing is critical for many natural language processing systems (Curran et al., 2007), as it enables rich information to be extracted from text.

Charniak (2000) parsed text by learning to compute top down production probabilities. The Stanford (Klein and Manning, 2003) and Berkeley (Petrov and Klein, 2007) parsers represent a parse as a sub-class distribution on a tree, and maximize the entropy of predicting label probabilities for each node. Advanced parsers use techniques like training on unlabeled text (McClosky et al., 2006), and parse re-ranking (Charniak and Johnson, 2005) to improve their accuracy.

To achieve such accuracy requires conditional probabilities to be computed correctly far up the parse tree, and requires plenty of sub-class labels (typically over 4000) to produce a sufficiently fine grained analysis that accommodates subtle distinctions in text. The resulting transition matrices become large very quickly.

Titov and Henderson (2007) used vectors of real numbers to represent words in a history based model with a mean field representation. Such word vectors have the advantage of maintaining expressive power with much lower dimensionality, typically just 100 dimensions. Garg and Henderson (2011) extended this model to use a Restricted Boltzmann Machine (RBM, Hinton et al., 2006) representation.

Socher et al. (2010) constructed a parse tree using word vectors in a Recursive Neural Network (RNN, Bengio et al., 1994), and applied a similar unsupervised approach for sentiment analysis (Socher et al., 2011).

We focus on the RNN models of Socher, which perform well on short (up to 15 word) sentences, achieving a parse F-score of 92.06% on right binarized text, and we obtain a baseline of 83.94% by applying this to all sentences.

We develop an abstract fully generative model of parsing by considering a Deep Belief Network (DBN, Hinton et al., 2006), and by examining the mean field approximation, contrast this with the conditional RNN model to discover underlying properties of the scoring function.

Motivated by insights from the DBN model, we use an RBM to implement a better model for scoring production probabilities. While this has negligible affect on short sentences, it achieves a 0.83% gain in parsing performance on long sentences.

Noting that the RNN parser's CKY performance drops over large trees, we also develop a novel method of applying gradient methods during evaluation time that improves the parsing F-score by 0.38% for long sentences.

While demonstrating these methods over an RNN model, the gradient method can broadly be applied to a wide range of conditional tree-based models.

2 Background

The Charniak (2000) parser represents a parse probability as the product of the top down production probabilities of a parse. One intuition is that the head node represents the entire sentence, and lower nodes are the probabilities of expressing a span of the text in each possible way. This makes each parse a downwards pointing conditional graphical model, and as explained by Charniak (1997), parser performance increases as more conditional information is used in calculating these production probabilities.

The more recent Stanford (Klein and Manning, 2003) and Berkeley (Petrov and Klein, 2007) parsers use increasingly fine grained sub-class schemes to convey rich information to each parse node to increase performance.

2.1 Berkeley Parser

The Berkeley parser solves the graphical model using the inside-outside algorithm to calculate the distribution of latent subcategories from transition probabilities, β:

$$\beta(A_x \to B_y C_z) := \frac{\#\{A_x \to B_y C_z\}}{\sum_{y'z'}\{A_x \to B_{y'} C_{z'}\}} \tag{1}$$

The algorithm is an application of the Expectation Maximization algorithm (Dempster et al., 1977) for a tree based graphical model. Each tree node maintains a distribution of being in each class with probability A_x. The Berkeley parser uses about 4000 different classes, requiring the optimization algorithm to learn 16 million transition probabilities. This makes computation slow. Other parsers like the C&C parser (Clark and Curran, 2004) speed up this process by effectively limiting the transition space through operator rules that reduce the transition space.

2.2 Word Vectors

Instead of representing the sparse high dimensional distributions of word selection, classes and transitions directly, the word-vector approach attempts to encode such sparse distributions into a much shorter (say 100 dimensions) dense vectors of latent states.

This method was used in the adjacent field of language modeling (Chen and Goodman, 1996), which aims to predict the smoothed frequency of n-gram distributions. Using Neural Networks in a method similar to Principle Component Analysis (Pearson, 1901), Mnih and Hinton (2007) show a log-bilinear model having a low perplexity in predicting the last word of an n-gram. This approach effectively encodes words into same-sized word vectors which can be combined to maximally represent n-gram distributional information through a neural network.

2.3 Neural Networks

Neural Networks are the natural extension of logistic regression that can be used to transform word-vectors into sub-class distributions.

Given a dataset of inputs X and their corresponding outputs Q, a simple logistic regression predicts an output $q_x \in Q$ for each input $x \in X$ by learning a matrix W such that

$$p_x = P(q_x|x;W) = \sigma(Wx) \tag{2}$$

where σ is the sigmoid or hyperbolic tangent function. Learning consists of training W to minimize the loss error function $E = \sum_{x \in X} E_x$, commonly using the square or cross entropy loss

$$E_x = (q_x - p_x)^2 \text{ or } E_x = q_x \log(p_x) + (1 - q_x)\log(1 - p_x) \tag{3}$$

which can be solved through Stochastic Gradient Descent (SGD, Bottou, 2010) by subtracting from W the gradient of E_x for each training example, multiplied by a learning rate.

Stacking logistic regression layers into neural networks (Rumelhart et al., 1986) is possible by passing back the gradient of the error using the derivative chain rule, but learning may become slow if both the outputs and inputs to a logistic regression are latent. This can be overcome by pre-training with Restricted Boltzmann Machines (RBM, Hinton et al., 2006).

2.4 Neural Network Parsing

Costa et al. (2003) previously used a Neural Network for parsing. Titov and Henderson (2007) achieved successful results, with an F-score of 89.3% on sentences of up to 15 words of the WSJ dataset Marcus et al. (1993) by implementing a recurring neural network, predicting the constituents and label decisions at each step based on text features and previous decisions.

Garg and Henderson (2011) used RBM in a similar approach to dependency parsing. Here, while the prediction step stays the same, the learning method is adjusted. Instead of trying to learn $p_x = P(q|x;W)$ over the dataset $\{X,Q\}$, the goal is to learn the generative probability of $P(x,q;W)$. Garg and Henderson implements a recurrent model, with weight biases derived from previous parse decisions, achieving 89% dependency parser score on short sentences.

3 Recurrent Neural Network Parsing

Socher et al. (2010) used a Recurrent Neural Network (RNN) that represented a parse tree consisting of real-valued node vectors from which the various sub-class distributions and parse decisions were computed through logistic regression classifiers. The conditionally independent nature of the elements of each node vector allows the normally sparse sub-class distributions to be compressed into a length of just 100 dimensions.

3.1 Leaf Layer

At the lexical leaf node layer, each tokenized word in the text is represented by a word vector t_i that is supplied from a pre-generated word-to-vector table L. This table L (the Lexicon) is generated through a distributional similarity process (Collobert and Weston, 2008) using back-propagation through a series of transformations of word and feature distributions.

The text is padded with a pair of start-of-sentence and a pair end-of-sentence tokens, each pair with their own vector. A direct 300→100 feature word-to-leaf logistic regression combines the 100 dimensional word-vectors for the current word t_i, previous word t_{i-1} and next word t_{i+1}, to generate the 100 dimensional real-valued vector v_i that acts as a leaf-vectors for the parse tree, as shown on the left of Figure 1. The word-to-leaf logistic regression learns a 300x100 element matrix Y, which is composed of three 100x100 element matrices Y_1, $Y_2$and Y_3, so that the leaf vectors v_i for each word are calculated by:

$$v_i = \sigma(Y_1 t_{i-1} + Y_2 t_i + Y_3 t_{i+1}) \tag{4}$$

For example, in Figure 1, the leaf node a above the word "John" would be calculated as:

$$v_a = \sigma(Y_1 t_{[start]} + Y_2 t_{[John]} + Y_3 t_{[eats]}) \tag{5}$$

As well as predicting parse information, the leaf-vectors predict Part of Speech (POS) tags which, when given a gold standard POS tag during training, provides an additional error gradient to help learn the word-to-leaf logistic regression layer.

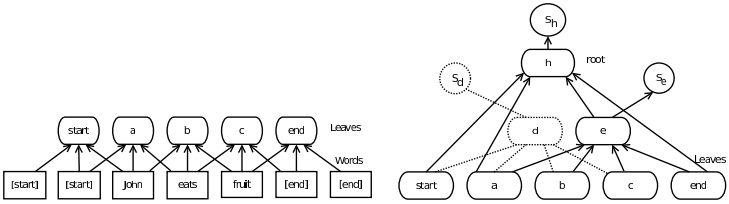


Figure 1: (left) Word to leaf network for text "John eats fruit" and (right) parse tree for parse $T_1 = a, (b, c)$ and (dotted) $T_2 = (a, b), c$ showing scoring nodes s_d, s_e and s_h. For clarity the predictions for the POS and phrase classes are not shown, however these are logistic regression classifications directly from the tree leaf nodes, and tree non-leaf nodes respectively.

3.2 Non Leaf Layer

The non-leaf node (i.e., parent node) vectors, p, of the tree are predicted using a 400→100 feature child-to-parent logistic regression layer. This logistic regression takes four lower tree node vectors as inputs. Two of these input vectors c_1 and c_2 are the vectors of p's left and right children, which may themselves be either parent or leaf node vectors. The other two input vectors are the leaf-node vectors v_1 and v_2 of the two words that directly surround the span of text represented by p. The child-to-parent logistic regression uses a 400x100 element matrix W, which is composed of four 100x100 element matrices W_1, W_2, $W_3$and W_4, so that the parent vectors p are calculated by:

$$p = \sigma(W_1 c_1 + W_2 c_2 + W_3 v_1 + W_4 v_2) \tag{6}$$

In Figure 1, the parent node h for the parse "John (eats fruit)" would be calculated as:

$$p_h = \sigma(W_1 v_a + W_2 p_e + W_3 v_{start} + W_4 v_{end}) \tag{7}$$

As well as being used to predict parent vectors higher in the parse tree, the parent vectors are used to predict node class labels (like VP or NP), and a real-valued score feature s_p which is used to predict whether a particular parent node is part of the gold-parse tree. The score is calculated using the output of a 100→1 logistic regression classifier, with parameter R:

$$s = \sigma(r) \text{ where } r = Rp \tag{8}$$

3.3 Parse Generation

Parse trees are built from the leaves up. The leaf nodes vectors are first calculated, and then parent vectors are calculated for each pair of adjacent leaf nodes. These parent vectors are then scored using the node-to-score classifier.

For subsequent layers up the parse tree, each new parent vector is computed using each combination of potential children vectors that might constitute the new parent vector. These new parent vectors are each scored using the node-to-score classifier. The computed score is added to the scores for sub-trees below, and the CKY algorithm (Kasami, 1965) is used to store and select the highest scoring overall tree.

The RNN's continuous vector model does not have a direct class representation, so there are no equivalence classes as needed for a packed chart. To overcome this, the RNN uses a beam to store a selection of high-scoring parses and their node vectors. A beam size of one was used in most experiments, as larger beam sizes gave little improvements. This is consistent with Socher et al. (2010)'s experiments and observations.

3.4 Updating the Weights

The parameters for each regression layer are updated using gradient descent, back propagating the error gradient through both the gold and the generated tree. The error function used is a hinge loss error gradient for the predicted scores s_i.

$$\frac{dE}{dr} = \begin{cases} -1 & \text{predicted, not gold} \\ 0 & \text{predicted+gold} \\ 1 & \text{gold, not predicted} \end{cases} \tag{9}$$

For gold trees, the gradient is 0 if the computed parse includes the gold node, and 1 if it does not. This is added to the gradients for the logistic regression POS tagger and node category tagger. For computed trees, the error for the predicted score s is 0 if the node is included in the gold parse, and -1 if it is not.

The errors for each tree are propagated down through each tree to the leaves, to update the word-to-leaf classifiers and eventually update the word vectors in the lexicon L. Note the gradient is calculated on r, not s (i.e. before the sigmoid function is applied), which has the effect of making it an entropy based gradient.

4 Generative Model

The RNN model is an upwards pointing conditional model in which the parents are conditioned on the children. This ignores aspects of a generative nature. By contrast, the intuition of Charniak shows a parse can be generated top down through a conditional generative process from a root node. To bridge these two models, we develop a novel expressive generative model (using a Deep Belief Network) that is conditional both up and down a tree, and examine the similarity of the equations with those of an RNN.

We are also motivated towards using a generative top-down conditional model because of the intuition that the speaking and writing process starts with ideas, which are reformulated into phrases and expressed in words. This process supports the notion that words are conditionally chosen based on ideas which would be found higher in the tree. A counter argument might exist that for the listener, the ideas heard are selected conditionally on the words of the text, however, the words and ideas were chosen by the speaker, so this is a weaker argument.

This bidirectional approach does not require these arguments to be mutually exclusive, as learning to speak and learning to listen might be shared aspects of one bidirectionally conditional model, transforming ideas into words, and back again.

4.1 Hypothetical Model

First we consider a (completely impractical) hypothetical model of parsing that follows the top-down approach, so we can later constrain it and compare it with the RNN model.

In this hypothetical model, the structure and meaning of any sentence can be represented by a 100 dimensional root vector. This root vector generates two child vectors, one representing a span of text on the left, and one representing a span on the right, and each of these child vectors continue generating their own two children vectors until some stopping process makes leaf vectors that generate words.

In this way, the process implements a top down generative conditional probability model (Charniak, 2000) in which the children are conditionally dependent on their parents.

To parse of a piece of text, samples from every possible root node vector generate every possible tree and their resulting texts. These trees and latent variables are then selected only where the generated text equals the supplied text. From this sample, the most likely parse is determined to be the most likely tree that generated the text.

The parameters of this hypothetical model would be tuned to create an optimum distribution at the root node, and an optimum conditional child generation distribution and stopping process to best model the parses of the text.

4.1.1 Constrained Model

We take this hypothetical model, and explore whether applying the constraints found in a Deep Belief Network (DBN, Hinton et al., 2006) would severely limit the model's expressive power.

A DBN consists of an RBM which generates a stationary distribution vector of binary values that acts as the complementary prior (CP, Hinton et al., 2006) for a downwards pointing directed acyclic graph (DAG) of binary vectors, each conditional on the parent. The complementary prior is defined by Hinton et al. to be the prior distribution that contains exactly the correct correlations necessary to enable the posterior parent distribution to be factorial (independent of one-another) given the child vector.

To implement a parse tree using a DBN, we firstly require that elements of each latent node-vector p (including the root vector) only take values 0 and 1. These vectors could potentially predict any other latent multi-class variable from p through soft-max classifiers, so a binary constraint should not be too onerous.

Secondly, we require that the root node be able to generate a stationary distribution that is optimum to generate the text. We note that RBMs can flexibly generate any distribution of child vectors, limited only by the number of hidden units (the dimension of p) (Le Roux and Bengio, 2008). For modeling text, this should be sufficient, since the text is independent of the trees when given the leaf vectors' distributions.

Thirdly, we require that the DBN generates a tree with downwards pointing conditional distributions. The DBN will naturally support the conditional distribution with function F:

$$P(c_1, c_2|p) = F(p, c_1, c_2) \tag{10}$$

and the probability of generating a tree T given its root will be:

$$P(T) = \prod_{nodes\ p \in T} F(p, c_1, c_2) \tag{11}$$

Finally, however, using a DBN within a tree structure presents the problem of maintaining a complementary prior at every parent node throughout the tree. Typically, the DAG used

with RBMs only recover the original complementary prior at every alternate layer. This can be overcome by initializing W with a symmetric weight matrix with $W = W^T$.

A second issue is that as we travel down the tree, the total vector count grows, so an issue of double-counting the prior arises. The prior must be a prior of both its children together, and not each child taken separately. While constructing a Deep Boltzmann Machine (DBM), Salakhutdinov and Hinton (2009) discuss how to manage the asymmetric application of DBN so the downward posterior has twice the dimensionality of the upwards posterior.

4.1.2 Parent Probability

The main feature of the DBN is that inference can be performed quickly to calculate the parent vector given the children vectors. Given a tree T, this allows us to calculate up the tree, even though it is through a downwards pointing conditional model, so the ith element of p can be predicted by:

$$P(p^{(i)} = 1|c_1, c_2) = \sigma(c_1 W_1 + c_2 W_2)^{(i)} \tag{12}$$

It is important to note that here c_1, c_2 and p are binary sampled values, not their probabilities or averages, whereas the resulting probabilities are distributions. However, used as a mean-field approximation, Equation (12) is essentially the same as that of the RNN. The DBN overcomes this limitation by sampling at each step within the tree.

4.1.3 Tree Probability

For each word sequence, there are multiple tree structures that could generate the same text, each with varying probabilities. Occasional examples exist like 'I saw that gas can explode' having two different parses of similar probability. To select the most likely parse, we must compare the generative probability of the two trees.

When two parses differ only at some parent node h, and the trees beneath h are of the form $T_1 = a, (b, c)$ and $T_2 = (a, b), c$ (see Figure 1) and d is the parent of (a, b) and e is the parent of (b, c), then we approximate the relative probability each tree was generated as follows:

Given the supplied text under nodes a, b, c being respectively x, y, z, the conditional vector distributions can be calculated through sampling going up each tree. In both T_1 and $T_{2,}$, when we arrive at nodes a, b and c, they will each have the same sampled distributions, as below a, b, and c, the two trees are identical. The mean field approximations for the most probable combinations of each vector a, b, c will be near their expected values $\bar{a} = E(a|x)$, $\bar{b} = E(b|y)$ and $\bar{c} = E(c|z)$.

Given no information, the prior probability of h will be the complementary prior, so taking a mean field approximation, the probability T_1 generates the text will be:

$$P(x, y, z|T_1) = \sum_h \sum_d P(h, \bar{a}, d) P(\bar{b}, \bar{c}|d) P(x|\bar{a}) P(y|\bar{b}) P(z|\bar{c}) \tag{13}$$

Ignoring the common terms $P(x|a)$ etc, we get:

$$P(x, y, z|T_1) \propto \sum_{hd} P(h, \bar{a}, d) P(\bar{b}, \bar{c}|d) \tag{14}$$

If again we make the mean field approximation that $P(\bar{b}, \bar{c}|d)$ only has probability mass at the point $E(\bar{b}, \bar{c})$, and apply Bayes rule ignoring common terms we get:

$$P(T_1|x, y, z) \propto P(\bar{a}, d) \tag{15}$$

4.1.4 Scoring the Tree Probability

The relative log probabilities for T_1 and T_2 is (where d and e are as shown in Figure 1):

$$\log(P(a, d)) : \log(P(e, c)) \tag{16}$$

So for children u, v of an arbitrary parent node h, where the parent node h has the distribution of the complementary prior, we can model the log probability $\log(P(u, v))$ by summing each one of the $2^{|h|}$ combinations of h using:

$$\log(\sum_h P(h, u, v)) = \log(\sum_h \exp(hW_1u + hW_2v + \gamma h + \alpha u + \beta v)) - \log(Z) \tag{17}$$

where α, β and γ are the biases to be learned with W_1 and W_2. We can ignore $\log(Z)$ as it is found in both T_1 and T_2. A common rearrangement is to multiply out the contributions made for each element $h^{(i)}$ of h being a 0 and a 1 (writing $\sum_i g^{(i)}$ as the sum of g's elements):

$$= \alpha u + \beta v + \sum_i \log(1 + \exp(\gamma + W_1u + W_2v))^{(i)} \tag{18}$$

$$= \alpha u + \beta v - \sum_i \log(1 - \sigma(\gamma + W_1u + W_2v)))^{(i)} \tag{19}$$

$$\log(P(u, v)) = \alpha u + \beta v - \sum_i \log(1 - h_i) \tag{20}$$

where $h_i = P(h^{(i)} = 1)$. Since u and v are factorial in h, the resulting log probability is also largely factorial in h making it suitable for a regression layer. As the RNN operates with mean-field values, it can learn $\log(P(u, v))$ through regression based on h using:

$$s(h) = \sigma(Rh) \simeq \log(P(u, v)) \propto \log(P(T_1)) \tag{21}$$

In this way, the approximate conditional probability of each parent is calculated upwards using the RNN model, and the log probability of each parse can be estimated by summing the expected values of s, and learned through back-propagation.

4.1.5 Estimating the Scoring Function

We can estimate the value of s by taking the Taylor expansion of the log probability of $\log(P(u, v))$ in terms of h_i:

$$log(P(u,v)) \simeq C + \sum_i -\log(1-h_i) \simeq C + \sum_i h_i + O(h_i^2) \tag{22}$$

The Taylor expansion of the logistic function used by the classifier by contrast is:

$$\sigma(Rh) = C + \sum_i \frac{1}{4} R^{(i)} h_i + O(h_i^2) \tag{23}$$

Comparing the linear h_i terms in Equations (22) and (23) would suggest an expected value of $R^{(i)} = 4$

It is interesting to note that in our experiments with the RNN model, the average value of $R^{(i)}$ was 1.77. However, the implementation of the RNN used the tanh function, instead of the sigmoid function σ found in the abstract DBN model for the node to node matrices W_1 and W_2 which generate the input to the scoring function. Since $\tanh(x)$ outputs values in twice the range as $\sigma(x)$, this gave the RNN model an effective average value for $R^{(i)}$ of 3.54, within 13% of the predicted value.

4.1.6 Reducing Divergence

In practice, the mean-field distribution diverges from the true combinatoric binary distribution. This becomes most noticeable when a node's distribution is sharply constrained by its own parent node, and is the result of each node being conditioned only on its children, and not on its parents. To reduce this effect, conditioning each node's probability on leaf-node vectors adjacent to the span (as in Equation 6) helps bring in some of the conditional information of the parent's node vector, since the parent node-vector is itself conditioned on its two children, one being the original node, the other being an ancestor of one of the adjacent leaf node.

4.2 RBM to Improve Modeling S

The RNN can be thought of as a mean-field approximation of the DBN model that learns to model the log probability of $P(T_1)$ through Equation (21). The R term implies that s is computed through a logistic regression.

Although the DBN abstract model indicates logistic regression on h should provide a good solution of $P(T_1)$, the mean-field approximation diverges sufficiently enough that incorporating contextual features makes a significant improvement to the results (Socher et al., 2010). This suggests that a superior scoring classifier might also further improve the results. Recalling that RBMs are better at modeling general distributions, and that when given the complementary prior, the probability of $P(T_1)$ takes the form of Equation (17), this motivates that $s(p)$ would be better modeled by a Restricted Boltzmann Machines (RBM).

RBMs have been used before in parsing (Garg and Henderson, 2011). They aim to model a generative probability $P(x,h) \propto \exp(xWh)$ and are often trained through Gibbs sampling, one layer at a time. Layer-wise training is harder for a recursive model, however, they can be used to model the distribution of the scoring function $s(p)$ in the approximate RNN model:

$$s(p) \propto \sum_h \exp(hUp + ap + bh) \tag{24}$$

4.2.1 Configuration

RBMs can be used in several configurations for modeling the probability of $s(p)$.

One method involves using two RBMs, one with energy functions $E_1 = h_1 U_1 p$ to model the probability that the parent node is part of the correct parse, and one with $E_2 = h_2 U_2 p$ to model the probability that it is not. In this case, the probability the parent node is part of the parse is given by $\exp(E_1)/(\exp(E_1) + \exp(E_2))$.

Another configuration uses a single RBM, $E = hMp$ and regression $s = \sigma(Vh)$. This model most naturally extends to a three layer neural network.

The discriminative RBM (Schmah et al., 2009) which we use goes one step further, by assuming that s is itself a latent feature of the RBM which has energy $E = hMp + hVs$. This means that $P(s = 1|p) = \sigma(Vh)$, while $P(h_i = 1|p) = \sigma(Mp + Vs)$. By including the Vs term in the calculation of $P(h_i = 1)$, it enhances the expressiveness of the model.

4.2.2 Training

Instead of using Contrastive Divergence (CD, Hinton et al., 2006) to learn the parameters of M and V, we perform discriminative RBM training, since the dimensions of s are so small. This is done by calculating the RBM twice, once for $s = 0$ and once for $s = 1$. These results can be combined to give a value of $P(s = 1|p)$, and the gradient can be used to update M and V.

We train this model for s as a post-process to training the RNN, while holding fixed the parameters of the underlying RNN model. The motivation was to effectively implement the top layer-wise training seen in deep RBM training models. We do not allow the gradient to back-propagate to update W or other parameters, as it did not improve our overall results.

4.3 Improvement to CKY through Leaf Vector Nudging

Different parse decisions made at lower levels of the tree result in different parent vectors. Even when the structure of two trees differs only slightly, as T_1 and T_2 did in Section 4.1.3, the changes to the parent vectors are not local but propagate up all the way to the root node.

This affects the ability of the CKY algorithm to find the parse with the highest log likelihood score. For longer sentences, the scoring model may give the gold parse the highest parse score, but using the CKY algorithm with a small beam may fail to find this highest scoring parse.

In this situation, there must be a node where the CKY algorithm made an incorrect decision. For the parent of this node, the gold parse's local sub-tree score was lower than several incorrect parse's sub-tree scores, enough to incorrectly fill the CKY beam with the wrong parent node vector. In this case, there will be some group of higher nodes that would have scored higher had the gold node's children vectors been used, rather than the incorrect parse node's children vectors found in the CKY beam (and by a more significant margin), but the CKY algorithm would have never calculated them.

With this motivation, we examine how to temporarily nudge some parameters of the model to encourage the CKY algorithm to find the best overall parse and, in particular, how the gradient of higher node scores can be back-propagated at evaluation time to lower nodes to encourage the CKY algorithm to favour parse decisions at a lower level that will do better at higher levels.

4.3.1 Learning Leaf Vectors

Another motivation for this approach comes from considering the process of updating the word-vectors during learning. The back-propagation for the RNN passes through the tree and leaves into the lexicon L. One can imagine that words with multiple senses will be given a blended vector representation, sometimes being tugged towards one sense's vector, and sometimes tugged towards another. In this instance, even during testing, where a parse is mostly correct it is likely that the vector would have been tugged mostly in the correct direction.

With this in mind, we found it possible to back-propagate the parse score gradient down through the tree and into the leaf vectors temporarily, just for the current parse. The goal of this is to temporarily give the leaf vector a value closer to the sense that is currently being used, thereby giving a chance of fixing any errors in the parse.

During testing, however, we do not know the correct parse, so cannot use the correct gradient for back-propagation. Instead, we use the parse scores s as a measure of confidence on each parse node, and back-propagate the direction of the gradient $\frac{ds}{dp}$ from the most confident nodes.

4.3.2 Method of Gradient Improvement

First, a parse of the sentence is generated using the RNN model and scores s_i for each parse decision calculated. The average score $\hat{s}$ is computed for the entire tree, and a reliability variable $g_i = \max(0, s_i - \hat{s})$ calculated to determine the most confident parse decisions. These values are then used as the error model and the gradient $kg_i \frac{ds_i}{dp}$ is back-propagated to the leaf-node level. This gradient is temporarily added to the leaf vectors and the sentence is re-parsed and the highest scoring parse selected. Finally the new parse is re-scored using the original leaf vectors and the new parse is only selected if the total score exceeds the sentence's original total parse score. Since we expect half the nodes to contribute gradients, we set $k = 2/(10 + \text{len(sentence)})$ which empirically worked well with the development set.

5 Results

We used the WSJ corpus of the Penn Treebank (Marcus et al., 1993), using sections 2-21 for training, 22 as a development set and 23 as the test set.

We require binary branching parse trees so, before commencing we made the following adjustments to the WSJ data that was kept for both training, development and final evaluation: all traces were removed; all unary rules (nodes with a single child) were collapsed (the resulting label was the POS tag of the leaf word); all nodes with more than 2 children were right branched (except for the most right node if it was punctuation) and the resulting node labels of the newly generated parents was made the same as for the original multi-branched parent; for short sentence experiments only sentences containing 15 or fewer tokens (including punctuation) were used.

The networks were trained with 500 passes over the training corpus with an initial learning rate of 0.005 that decreased by 2% of its value after pass. The parameters were stored whenever evaluation against the development set revealed a higher result, which was tested every 1000 sentences. Final testing was performed against section 23.

The word-vector table was initialized from pre-computed values in the lexicon (Collobert and Weston, 2008) and the word vectors took the values $L_{u(S,i)}$ according to each word $u(S, i)$ of the

Method	F1	significance
Socher ⩽ 15 words	92.06%	
RNN ⩽ 15 words	92.41%	
RNN all sentences	83.94%	
Gradient	84.32%	+0.38% (p=0.0001)
RBM	84.77%	+0.83% (p=0.0008)

Table 1: F1 Test scores for section 23 of the WSJ

text. For tokens found in WSJ but not in the table, we added a fresh token with a value given by either the lowercase version of the token (if found) or with the value of the *UNKNOWN* token. All numeric characters were replaced with the digit '2'.

5.1 De-Binarization

The RNN and DBN model both require binarized trees. The standard WSJ corpus is not binarized, but (especially at the leaf layer) contains many nodes consisting of more than two children. For example, tokens of multi-word nouns like "`New York Stock Exchange`" will all be children of the same node, while binarized trees might nest these as `(New (York (Stock Exchange)))`. During evaluation, the computed F-scores for binarized trees will appear inflated compared to those of un-binarized trees, since it is easier to learn and reproduce the additional right bracketings.

For comparison with other systems, we consider how the binarization process could be reversed so that generated trees may be compared with the original unbinary WSJ corpus. This could be done through coded compounding rules based on predicted phrase categories labels, or by learning a de-binarize feature. The method we used was as follows.

There are 28 non-terminal labels that are applied to the gold-standard text (e.g. `S, VP, NP`). During the process of right binarizing the text, we label the newly created nodes with one of 28 newly created categories (e.g. `!S, !VP, !NP`) generated by adding '`!`' to the label of the new node's parent node. In this manner we identify which nodes to delete when debinarizing generated parses.

Unless otherwise noted, the results we show are for binarized trees, as binarized trees were used in the original experiments by Socher (2012).

5.2 Evaluation

Testing was performed using evalb. The F1 score (precision and accuracy are are identical for binarized trees) was taken as the overall reported performance.

A result of 92.41% was achieved by training on all the text, and testing on just the sentences up to 15 tokens long. A result of 83.94% used the same model, testing on the entire test set, both short and long sentences (refer to Table 1). For comparison with other parsers, we obtained a debinarized F-Score of 89.47% on short sentences.

Significance was calculated using Dan Bikel's Randomized Parsing Evaluation Comparator. [1]

[1] http://www.cis.upenn.edu/~dbikel/download/compare.pl

token	$error^2$	token	$error^2$
.	0.128	if	0.032
?	0.125	It	0.029
:	0.077	get	0.028
.	0.054	what	0.026
–	0.046	That	0.025

Table 2: Tokens and square errors of modifications during testing

5.3 RBM

The discriminative RBM model was used as a post-process to learn an improved model of the scoring function. It was trained in about 24 hours, holding the other parameters of the model fixed. It out-performed the original model by 0.83% (with a p-value of 0.0008). The training was entirely discriminatively performed with no back propagation through the structure.

5.4 Gradient Update

The gradient update method was used entirely during evaluation time. It improved the F-score by just 0.38% (with a p-value of 0.0001). The significance was higher because more of the mistakes it made were also made in the baseline model.

The tokens with the greatest average modification were mostly punctuation and function words that shaped the structure of the sentence (see Table 2). These words control the sentence structure but have little distributional information. Other highly modified words were those with low frequency counts in the corpus.

Conclusion

We have demonstrated the performance of a Recursive Neural Network parser to binarized forms of all sentences in the WSJ corpus, obtaining a baseline of 83.94%. We have presented a method to improve parser performance by using a discriminative Restricted Boltzmann Machine for scoring productions increasing F-score by 0.83%. We have also presented a gradient method that can be used during evaluation that augments the CKY algorithm and improves accuracy by a separate 0.38%. We have provided a framework to draw a connection between top-down generative conditional parsers with bottom-up conditional RNN parsers, and using this framework have analytically calculated estimates of the learned RNN parameters.

In future work, we hope to implement the generative model described in this paper using Contrastive Divergence starting with just the leaf layer and increasing one parse layer at a time.

The methods we presented are easily applied to other recursive tasks and network structures and provide a link between generative and conditional parsing. These improvements can be used in future down-stream tasks.

Acknowledgments

We would like to thank the reviewers and Schwa-Lab for their comments. This work was supported by Australian Research Council Discovery grant DP1097291, the Capital Markets CRC and an Australian Postgraduate Award. We are grateful to Richard Socher for his guidance reproducing his results in our system.

References

Bengio, Y., Simard, P., and Frasconi, P. (1994). Learning long-term dependencies with gradient descent is difficult. *Neural Networks, IEEE Transactions on*, 5(2):157–166.

Bottou, L. (2010). Large-scale machine learning with stochastic gradient descent. In *Compstat*, volume 2010, pages 177–186.

Charniak, E. (1997). Statistical techniques for natural language parsing. *AI magazine*, 18(4):33.

Charniak, E. (2000). A maximum-entropy-inspired parser. In *North American chapter of the Association for Computational Linguistics*, pages 132–139.

Charniak, E. and Johnson, M. (2005). Coarse-to-fine n-best parsing and maxent discriminative reranking. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 173–180.

Chen, S. F. and Goodman, J. (1996). An empirical study of smoothing techniques for language modeling. In *Proceedings of the 34th annual meeting on Association for Computational Linguistics*, pages 310–318.

Clark, S. and Curran, J. R. (2004). Parsing the wsj using ccg and log-linear models. In *Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics*, ACL '04.

Collobert, R. and Weston, J. (2008). A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th international conference on Machine learning*, pages 160–167.

Costa, F., Frasconi, P., Lombardo, V., and Soda, G. (2003). Towards incremental parsing of natural language using recursive neural networks. *Applied Intelligence*, 19(1):9–25.

Curran, J. R., Clark, S., and Bos, J. (2007). Linguistically motivated large-scale nlp with c&c and boxer. In *Proceedings of the 45th Annual Meeting of the ACL on Interactive Poster and Demonstration Sessions*, pages 33–36.

Dempster, A. P., Laird, N. M., and Rubin, D. B. (1977). Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 1–38.

Garg, N. and Henderson, J. (2011). Temporal restricted boltzmann machines for dependency parsing. In *Proceedings of the 11th International Conference on Parsing Technologies*, pages 11–17.

Hinton, G. E., Osindero, S., and Teh, Y. W. (2006). A fast learning algorithm for deep belief nets. *Neural computation*, 18(7):1527–1554.

Kasami, T. (1965). An efficient recognition and syntax analysis algorithm for context-free languages. Technical report.

Klein, D. and Manning, C. D. (2003). Accurate unlexicalized parsing. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics-Volume 1*, pages 423–430.

Le Roux, N. and Bengio, Y. (2008). Representational power of restricted boltzmann machines and deep belief networks. *Neural Computation*, 20(6):1631–1649.

Marcus, M. P., Marcinkiewicz, M. A., and Santorini, B. (1993). Building a large annotated corpus of english: The penn treebank. *Computational linguistics*, 19(2):313–330.

McClosky, D., Charniak, E., and Johnson, M. (2006). Effective self-training for parsing. In *Proceedings of the main conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics*, pages 152–159.

Mnih, A. and Hinton, G. (2007). Three new graphical models for statistical language modelling. *Proceedings of the 24th international conference on Machine learning - ICML '07*, pages 641–648.

Pearson, K. (1901). On lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 2(11):559–572.

Petrov, S. and Klein, D. (2007). Improved inference for unlexicalized parsing. In *Proceedings of North American Association for Computational Linguistics HLT 2007*, pages 404–411.

Rumelhart, D. E., Hintont, G. E., and Williams, R. J. (1986). Learning representations by back-propagating errors. *Nature*, 323(6088):533–536.

Salakhutdinov, R. and Hinton, G. E. (2009). Deep boltzmann machines. In *Proceedings of the international conference on artificial intelligence and statistics*, volume 5, pages 448–455.

Schmah, T., Hinton, G. E., Zemel, R., Small, S. L., and Strother, S. (2009). Generative versus discriminative training of rbms for classification of fmri images. In *Proceedings of Neural Information Processing Systems*, volume 2009.

Socher, R. (2012). Personal communication.

Socher, R., Manning, C. D., and Ng, A. Y. (2010). Learning continuous phrase representations and syntactic parsing with recursive neural networks. In *Deep Learning and Unsupervised Feature Learning Workshop*, pages 1–9.

Socher, R., Pennington, J., Huang, E. H., Ng, A. Y., and Manning, C. D. (2011). Semi-supervised recursive autoencoders for predicting sentiment distributions. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Titov, I. and Henderson, J. (2007). Constituent parsing with incremental sigmoid belief networks. In *Annual Meeting of the Association for Computational Linguistics*, volume 45, page 632.

Thread Specific Features Are Helpful For Identifying Subjectivity Orientation of Online Forum Threads

Prakhar Biyani[1] *Sumit Bhatia*[2] *Cornelia Caragea*[3] *Prasenjit Mitra*[1]
(1) College of Information Sciences and Technology, The Pennsylvania State University, USA
(2) Department of Computer Science, The Pennsylvania State University, USA
(3) Department of Computer Science and Engineering, University of North Texas, USA
prakharbiyani@gmail.com, sumit@cse.psu.edu, ccaragea@unt.edu,
pmitra@ist.psu.edu

ABSTRACT
Subjectivity analysis has been actively used in various applications such as opinion mining of customer reviews in online review sites, question-answering in CQA sites, multi-document summarization, etc. However, there has been very little focus on subjectivity analysis in the domain of online forums. Online forums contain huge amounts of user-generated data in the form of discussions between forum members on specific topics and are a valuable source of information. In this work, we perform subjectivity analysis of online forum threads. We model the task as a binary classification of threads in one of the two classes: subjective and non-subjective. Unlike previous works on subjectivity analysis, we use several non-lexical thread-specific features for identifying subjectivity orientation of threads. We evaluate our methods by comparing them with several state-of-the-art subjectivity analysis techniques. Experimental results on two popular online forums demonstrate that our methods outperform strong baselines in most of the cases.

KEYWORDS: Online Forums, subjecitivity, dialogue act.

Proceedings of COLING 2012: Technical Papers, pages 295–310,
COLING 2012, Mumbai, December 2012.

1 Introduction

A large number of online forums in various domains (e.g., health, sports, travel, camera, laptops, etc.) exists today, containing huge volumes of user-generated data in the form of discussions between members. The topics discussed in the threads of these forums are very unique in nature as they are often related to practical aspects of life (e.g., *How much to tip after bad service?*). Since such information is not available in other webpages, online forums are increasingly becoming very popular among internet users for discussing real life problems.

In this work, we *analyze subjectivity of online forum threads*. We identify two types of threads in an online forum: *subjective* and *non-subjective* and we model the subjectivity analysis task as a binary classification problem. Subjective threads discuss subjective topics that seek opinions, viewpoints, evaluations, and other private states of people, whereas non-subjective threads discuss non-subjective topics that seek factual information. Figure 1 shows a subjective thread from an online forum, Trip–Advisor New York. Figure 2 shows a non-subjective thread from the same forum. In the former, the topic of discussion is *whether to tip or not after bad service?*, which seeks opinions, whereas the latter seeks factual information about *bands/artists playing in December in Madison Square Gardens*. To the best of our knowledge, previous work on subjectivity analysis has not tackled the problem of identifying subjectivity orientation of online threads.

1.1 Why Subjectivity Analysis of Online Forum Threads?

- ***Improving forum search***: Internet users search online forums, generally, for two types of information. Some of them search the forums for subjective information such as different viewpoints, opinions, emotions, evaluations, etc., on specific problems instead of a single correct answer. Other users want short factual (objective) answers. Previous works on online forum search have focused on improving the lexical match between searcher's query keywords and thread content (Seo et al., 2009; Bhatia and Mitra, 2010; Duan and Zhai, 2011). However, these works do not take into account a searcher's intent, i.e., the *type of* information a searcher wants. Let us consider the following two example queries issued by a searcher to some camera forum: 1) How is the resolution of Canon 7D, 2) What is the resolution of Canon 7D. The two queries look similar, but they differ in their intents. In the first query, the searcher wants to know what other camera users think about the resolution of the Canon 7D, how are their experiences (good, bad, okay, excellent, etc.) with the camera as far as its resolution is concerned and other such types of *subjective* information. The second query, however, is *objective* in nature in which the searcher wants a factual answer, which, in this case, is the value of the resolution of the camera. Hence, queries having similar keywords may differ in their intents. Search algorithms based only on keyword search would perform badly for these types of queries. We believe that by knowing the type of information (subjective or objective) contained in a forum thread, these types of queries can be addressed in a better way. A forum search model can then match the searcher's intent with the type of information a thread contains in addition to the keyword match between the two and thus, handle the queries more intelligently.
- ***Abuse detection***: Online forums are informal in nature. Often, discussions in threads get heated with users getting engaged in abusive conversations. Forum administrators continuously monitor forums for such contents and remove them as they are against the community rules. These conversations are subjective in nature and hence can potentially

be detected by analyzing threads for subjectivity.

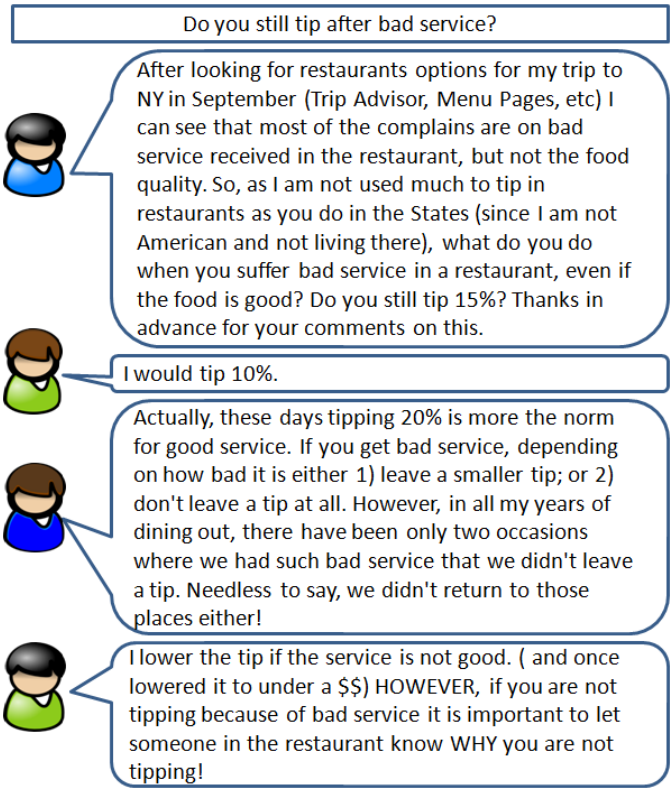


Figure 1: An example thread with subjective topic.

Previous works on subjectivity classification have extensively used lexical features such as bag-of-words, n-grams, combinations of n-grams and parts of speech tags, etc (Yu and Hatzivassiloglou, 2003; Li et al., 2008a; Aikawa et al., 2011). A major issue with these features is their high dimensionality feature space and hence there is a risk of model overfitting especially with small training data. In this work, we explore the possibility of using non-lexical and thread specific features for the subjectivity classification of threads. Specifically, we explore the following research question: *Can non-lexical thread specific features (e.g., number of users in a thread, number of posts in a thread, etc.) help in inferring the subjectivity of online forum threads?* To address the question, we propose and evaluate several thread specific features for subjectivity classification. We compare the performance of our classification model with various state-of-the-art techniques and show that our model outperforms the baselines in most of the cases.

Figure 2: An example thread with non-subjective topic.

1.2 Contributions

Our work has the following contributions:

1. We are the first to perform subjectivity analysis of online forum threads automatically.
2. We propose two new types of non-lexical features for subjectivity analysis of online forum threads: *structural* features and *dialogue act* features. Previous works on subjectivity analysis have mainly used lexical and syntactic features like n-grams, POS tags, subjectivity clues, etc. In this work, we empirically show that, for online forum threads, in addition to the traditionally used features, thread's structure and information about dialogue acts of its posts also help in analyzing thread's subjectivity.
3. We extensively evaluate our methods by comparing with various state-of-the-art baselines.
4. The dataset used in this paper for subjectivity analysis of online forum threads is being made publicly available for the research community [1].

The rest of the paper is organized as follows: The next section overviews the related work in the field of subjectivity analysis. Section 3 describes the problem and the features used for subjectivity classification. In Section 4, we describe our dataset, experimental settings and present and analyze the results of the classification. Section 5 concludes the paper and discusses the future work.

2 Related Work

Subjectivity analysis has been an active field of research due to its important applications in opinion mining, sentiment analysis, question-answering, summarization, etc. Here, we first provide a brief survey of works on subjectivity analysis in general and then we review the works that performed subjectivity analysis in different domains (online review sites, community answers, etc.) and used it in different applications (opinion mining, question-answering, etc.).

2.1 Subjectivity Analysis

Wiebe et al. (1999) did a seminal work on generating and using a gold standard dataset for subjectivity classification. They performed subjectivity classification of sentences using

[1] http://www.personal.psu.edu/pxb5080/dataSubj.html

basic features such as presence of a pronoun, an adjective, a modal, etc. in the sentence. Bruce and Wiebe (1999) performed a case study of manual subjectivity tagging. Wiebe and Riloff (2005) performed subjectivity classification of sentences in World Press articles using unannotated data. They used high precision rule-based classifiers for generating an initial training data and then used semi-supervised learning to iteratively learn subjectivity patterns and augment the training data. Su and Markert (2008) performed word sense subjectivity classification using the training data generated from the existing opinion mining resources and showed that the performance is comparable with that of the classifier trained on a dedicated training set. Other works have performed subjectivity classification across different languages (Mihalcea et al., 2007; Banea et al., 2008). They discussed and evaluated methods to develop subjectivity analysis tools for selected languages by applying machine translation on the available subjectivity analysis tools and resources for English language. Banea et al. (2010) performed subjectivity classification in six different languages and showed that including multilingual information improves the classification performance across all the languages. Mukund and Srihari (2010) proposed a vector-space classification algorithm boosted by co-training for subjectivity classification of sentences in Urdu Language.

2.2 Opinion Mining

An integral part of opinion mining and sentiment analysis is to separate subjective sentences from objective ones and then to identify the polarity (negative, neutral or positive) of the opinions expressed in the subjective sentences (Liu, 2010). Works in this area have mainly focused on online review sites for summarizing product reviews given by different users of those products (Hu and Liu, 2004; Ly et al., 2011). Our work, in contrast, deals with online forum threads. A review in a review site is a continuous piece of text written by a person with additional information such as ratings, date and time. On the other hand, a thread in an online forum has a distinctive structure due to the presence of messages posted by multiple users. Also, a review, usually, has a single role of providing user's feedback on a product whereas posts in a thread have multiple roles, e.g., a post can be a question, solution, feedback, junk, etc (Bhatia et al., 2012). These differences make subjectivity analysis of online forum threads different from that in review sites in both nature and the approaches that can be used for the analysis. For example, thread structure, role of posts and other thread-specific information can be used as features for subjectivity analysis (as will be described later in the paper).

2.3 Question-Answering

Subjectivity analysis has also been used to improve question-answering in online communities and social media (Li et al., 2008b; Gurevych et al., 2009; Stoyanov et al., 2005; Yu and Hatzivassiloglou, 2003; Somasundaran et al., 2007). Yu and Hatzivassiloglou (2003) classify documents and sentences from news data into facts and opinions with the aim of improving answering of complex opinion questions. Stoyanov et al. (2005) separate opinion (subjective) answers from factual (objective) answers and then filter out factual answers for opinion questions to improve answering of opinion questions in multi-perspective question answering. Somasundaran et al. (2007) identify different types of attitudes in questions and answers and then use it to improve opinion question answering on web-based discussions and news data by matching the attitude types of questions and answers. Li et al. (2008a) classify questions in Yahoo QA as subjective or objective using semi-supervised learning by utilizing the text of labeled questions and their unlabelled answers for learning subjectivity patterns.

Gurevych et al. (2009) used an unsupervised lexicon based approach to classify questions as subjective or factoid (non-subjective). They manually build a lexicon of subjective words and word patterns from annotated questions and classify test questions based on a score calculated using the number of patterns present in them. These works did subjectivity analysis of questions and answers given by single authors in community sites. In contrast, we analyze the subjectivity of online forum threads that contain replies from multiple authors. These differences have implications described in the previous paragraph.

2.4 Online Forums

In the domain of online forums, there have been two recent works that are close to our work. Hassan et al. (2010) performed sentence-level attitude classification in online discussions to model user interaction that may be helpful in facilitating collaborations. Zhai et al. (2011) classified sentences in online discussions as evaluative or non-evaluative for getting relevant opinion sentences. In contrast, our work does thread-level subjectivity classification as we are interested in knowing the subjectivity of the overall topic of discussion of a thread and plan to use it for improving online forum search in the future.

3 Problem Formulation and Feature Engineering

In this section, we state our problem and describe various features used in the subjectivity classification task.

3.1 Problem Formulation

An online forum thread discusses a topic specified by thread starter in the title and the initial post. The topics of discussion in the threads can either be subjective or non-subjective (See Figures 1 and 2 for examples of subjective and non-subjective threads, respectively). Based on the definitions of subjective and objective sentences given by Bruce and Wiebe (1999), we define a subjective topic of discussion as a topic that seeks people's opinions, viewpoints, evaluations, speculations, and other private states and a non-subjective topic as a topic that seeks factual information. We call a thread subjective if its topic of discussion is subjective and non-subjective if it discusses a non-subjective topic. We assume that in online forum threads subjective topics have discussions in subjective language (i.e., expressing different private states) and non-subjective topics have discussions in objective language (ie., expressing facts and verifiable information). We note that there may be some cases where the assumption does not hold good, however, analysis of such exceptional cases is not the focus of this paper and is left for future work.

Problem statement: Given an online forum thread T, our task is to classify it into one of the two classes: *Subjective* (denoted by *s*) or *Non-Subjective* (denoted by *ns*).

In this work, we assume that a thread has a single topic of discussion which is specified by the thread starter in the title and the initial post. Analyzing subjectivity of threads with multiple topics is a separate research problem that is out of scope of this work.

3.2 Feature Engineering

As discussed before, we wanted to explore the effect of using various thread specific features for subjectivity analysis of online forum threads and compare them with the state-of-the-art

subjectivity analysis techniques. In this section, we describe the features used and intuition behind using them. Table 1 lists the features used.

3.2.1 Structural Features

We posit that subjective threads have different structural properties than non-subjective threads. Since subjective topics have more scope of discussion, we expect the subjective threads to be longer and invoke more participation of users than non-subjective threads. We use the length of a thread and the participation of users in a thread as features. For the length, we use the length of the initial post, the length of the thread and the average of the length of all the reply posts in the thread as features. All the lengths are measured in terms of the number of words. For the participation, we use the number of users that participated in the given thread, the number of posts and the average number of posts by a user in a thread as features.

Feature Name	Description
	Structural Features
InitPostLength	Total number of words in the initial post.
ThreadLength	Total number of words in the thread.
NumPost	Total number of posts in the thread.
NumUser	Total number of users in the thread.
AvgPostAuthor	Average number of posts by a user in the thread.
AvgLengthPost	Average number of words in a post in the thread.
	Dialogue Act Features
numQues	No. of *question* posts in the thread.
numRepeat	No. of *repeat question* posts in the thread.
numClar	No. of *clarification* posts in the thread.
numDetails	No. of *further details* posts in the thread.
numSol	No. of *solution* posts in the thread.
numNegFB	No. of *negative feedback* posts in the thread.
numPosFB	No. of *positive feedback* posts in the thread.
numJunk	No. of *junk* posts in the thread
	Subjectivity Lexicon-based Features
NumSubjTitle	No. of subjectivity clues in the title of the thread.
NumSubjInit	No. of subjectivity clues in the initial post of the thread.
NumSubjReply	No. of subjectivity clues in all the reply posts of the thread.
	Sentiment Features
InitSentiAvgPos	Positive sentiment score of initial post based on all the indicative word patterns in it.
InitSentiAvgNeg	Negative sentiment score of initial post based on all the indicative word patterns in it.
InitSentiStrngPos	positive sentiment score of initial post based on the strongest indicative word patterns in it.
InitSentiStrngNeg	Negative sentiment score of initial post based on the strongest indicative word patterns in it.
ReplySentiAvgPos	Average of positive sentiment scores of all the reply posts based on all the indicative word patterns in them.
ReplySentiAvgNeg	Average of Negative sentiment scores of all the reply posts based on all the indicative word patterns in them.
ReplySentiStrngPos	Average of positive Sentiment scores of all the reply posts based on the strongest word patterns in them.
ReplySentiStrngNeg	Average of Negative Sentiment scores of all the reply posts based on the strongest word patterns in them.

Table 1: Description of various features used for subjectivity classification.

3.2.2 Dialogue Act Features

Online forum threads have conversational nature and hence there are different types of dialogue acts (question, solution, feedback, etc.) expressed in thread posts (Bhatia et al., 2012;

Jeong et al., 2009; Joty et al., 2011). For example, a thread starts with a *question* posted by the thread starter. Then, there are posts (by other users) that ask for some *clarifying* details about the question and the thread starter provides *further details* to make the question clearer. After getting the details, users suggest *solutions* and finally there are *feedbacks* (by the thread starter or other users) to the suggested solutions that can be *positive* or *negative*. Also, there may be posts that ask the *same question* (as asked in previous posts) and posts that are *junk* and not related to thread discussion. We posit that dialogue acts expressed in the posts maybe helpful in identifying thread's subjectivity. In a subjective thread, there could be multiple solutions suggested for a question (e.g. *Sony or Nikon which is better?*) as there is no single correct answer to subjective questions and hence multiple feedbacks would be given. In contrast, in non-subjective threads, since questions seek factual materials (e.g., *what do the numbers on camera lens mean?*), there is little scope of discussion or disagreement among solution providers and hence there would be less solutions suggested and less number of feedbacks. Also, in subjective threads, the discussions can get heated due to disagreements with users posting inappropriate content such as abuses which are *junk* as they are not related to the discusion whereas in non-subjective threads, these situations are unlikely to happen. To explore the impact of dialogue acts on a thread's subjectivity, we used eight dialogue acts in thread posts as proposed by Bhatia et al. (2012) and used their presence in a thread as features for the subjectivity classification. The dialogue acts are as follows: 1. Question, 2. Repeat Question, 3. Clarification, 4. Further Details, 5. Solution, 6. Negative Feedback, 7. Positive Feedback, 8. Junk. We implemented their classification model to identify the dialogue acts in thread posts. We designed 8 features corresponding to the 8 dialogue acts for a thread. Each feature represents the number of posts in a thread that belong to a given dialogue act class.

3.2.3 Subjectivity Lexicon Based Features

Subjective threads discuss subjective topics seeking private states such as opinions, emotions, evaluations, etc. whereas non-subjective threads seek factual information. This difference should result in differences in the vocabularies of these two types of threads. Subjective threads should contain words that are used to express subjectivity whereas non-subjective threads should either not have these words or have less number of these words. We call these words *subjectivity clues* in this paper. Hence, the frequency or term counts of subjectivity clues in a thread should be a good indicator of its subjectivity. We use a publicly available subjectivity lexicon compiled from MPQA corpus by Wiebe et al. (2005) to get the subjectivity clues. The lexicon contains 8221 subjectivity clues. Some of the examples of subjectivity clues from the lexicon are *abhor, abuse, bother, champion*. We count the number of subjectivity clues in the title, initial post and all reply posts of a thread, normalize the subjectivity clue counts with the number of words in the corresponding element (title, initial post, reply posts) and use them as features. For a thread, we computed three lexicon features: NumSubTitle, NumSubInit and NumSubReply. We calculated NumSubTitle and NumSubInit by normalizing the frequency counts of subjectivity clues in the title and the initial post, respectively, by their total number of words. For computing NumSubReply, we first calculated the normalized frequency counts of subjectivity clues for all the reply posts and then added all the normalized counts.

3.2.4 Sentiment Features

These features take into account the sentiment/emotion of a thread. We expect subjective threads to have posts with higher sentiments (as they expose private states) than the posts in

non-subjective threads. To calculate sentiment features for a thread, we compute sentiment strength of its individual posts using the SentiStrength algorithm (Thelwall et al., 2012). We use the implementation of the algorithm available at http://sentistrength.wlv.ac.uk/. The algorithm is developed specifically to compute sentiment strength scores for short informal pieces of text common in social media such as forum posts, blog comments, etc. SentiStrength calculates both positive as well as negative sentiment scores for a piece of text. This feature is desirable as the posts can express sentiments of multiple polarity and a single sentiment score (positive, negative or neutral) will not be able to capture the individual sentiments. For both polarities, the algorithm gives two types of scores for a piece of text (i) using the strongest sentiment-indicative word patterns and (ii) using all the sentiment-indicative word patterns and taking their average. Thus, we get four different sentiment strength scores for each post. We use the four sentiment strength scores for the initial post and averages of the four sentiment scores for all the reply posts as features, thus getting eight sentiment features for a thread (see Table 1).

4 Experiments

4.1 Data

To conduct our experiments, we used threads from two popular online forums: 1. **Trip Advisor–New York** that contains travel related discussions mainly for New York city [2] and 2. **Ubuntu Forums** that contains discussions related to the Ubuntu operating system [3]. We used a publicly available dataset [4] (Bhatia and Mitra, 2010). We randomly sampled 700 threads from both the datasets to conduct our experiments. Table 2 provides various statistics of the data. We selected these two forums because we wanted to evaluate our methods on two different genres of online forums. Ubuntu forums generally have technical discussions that tend to be non-subjective in nature whereas Trip Advisor is a travel related forum having discussions on topics like transport, hotels, restaurants, tourism, etc that are generally non-technical in nature and hence tend to be subjective.

Statistic	Trip–Advisor	Ubuntu
Total # threads	609	621
Total # posts	6591	3603
Total # users	1206	1786
Average thread length (in terms of # posts)	10.82	5.80
Average thread length (in terms of # words)	907	387.57
Average # users in a thread	1.98	3.41

Table 2: Statistics of the Dataset

We hired two human annotators for tagging the threads. The annotators were asked to tag a thread as subjective if its topic of discussion is subjective or non-subjective if the topic of discussion is non-subjective. The annotators were provided with a set of instructions for annotations. The set contained definitions of subjective and non-subjective topics with examples and guidelines for doing annotations. The annotations for each dataset were conducted in three stages. First, the annotators were asked to annotate a sample of 20 threads from the

[2] http://www.tripadvisor.com/ShowForum-g60763-i5-New_York_City_New_York.html
[3] http://ubuntuforums.org
[4] http://www.cse.psu.edu/ sub194/datasets/ForumData.tar.gz

dataset using the instruction set. Second, separate discussions were held between the authors and each annotator. Each annotator was asked to provide his arguments (for his annotations) and, in case of inconsistent arguments, he was educated through discussions. Next, he was given the full dataset for annotation.

The overall percentage agreement between the annotators and Kappa value for the Trip Advisor dataset were 87% and 0.713 respectively and for the Ubuntu dataset were 88.7% and 0.732 respectively, indicating substantial agreement between the taggers in both the cases. For our experiments, we used the data on which the annotators agreed. There were 412 subjective and 197 non-subjective threads in Trip Advisor dataset and 231 subjective and 390 non-subjective threads in Ubuntu dataset. The tagged dataset can be downloaded from the authors' website. [5].

4.2 Baseline

Lexical features such as n-grams and parts-of-speech tags have been shown to perform well for subjectivity analysis tasks. Many works have used these features for subjectivity classification (Li et al., 2008a; Yu and Hatzivassiloglou, 2003; Aikawa et al., 2011). In this work, we use classifiers built on these features as our baselines. We used the *Lingua-en-tagger* package from CPAN[6] for part-of-speech tagging. The extracted features and their description is given in Table 3. The table describes feature generation on a sentence containing three words W_i, W_{i+1} and W_{i+2}. POS_i, POS_{i+1} and POS_{i+2} are the parts-of-speech tags for the words W_i, W_{i+1} and W_{i+2}, respectively. For feature representation, we used term frequency as the weighting scheme (we empirically found it to be more effective than *tf-idf* and *binary* representations), and used minimum document frequency for a term to be included in the vocabulary as 3 (we experimented with minimum document frequency 3, 5 and 10 and 3 gave the best results).

Feature type	**Generated feature**
Uni	W_i, W_{i+1}, W_{i+2}
Uni+Bi	$W_i, W_{i+1}, W_{i+1}, W_iW_{i+1}, W_{i+1}W_{i+2}$
Uni+Bi+Tri	$W_i, W_{i+1}, W_{i+1}, W_iW_{i+1}, W_{i+1}W_{i+2}, W_iW_{i+1}W_{i+2}$
Uni+POS	$W_i, POS_i, W_{i+1}, POS_{i+1}, W_{i+2}, POS_{i+2}$
Uni+Bi+POS	$W_i, POS_i, W_{i+1}, POS_{i+1}, W_{i+2}, POS_{i+2}, W_iW_{i+1}, W_iPOS_{i+1}, POS_iW_{i+1}, W_{i+1}W_{i+2}, W_{i+1}POS_{i+2}, POS_{i+1}W_{i+2}$
Uni+Bi+Tri+POS	$W_i, POS_i, W_{i+1}, POS_{i+1}, W_{i+2}, POS_{i+2}, W_iW_{i+1}, W_iPOS_{i+1}, POS_iW_{i+1}, W_{i+1}W_{i+2}, W_{i+1}POS_{i+2}, POS_{i+1}W_{i+2}, W_iW_{i+1}W_{i+2}, W_iW_{i+1}POS_{i+2}, W_iPOS_{i+1}W_{i+2}, POS_iW_{i+1}W_{i+2}, W_i, POS_{i+1}POS_{i+2}, POS_iW_{i+1}POS_{i+2}, POS_i, POS_{i+1}W_{i+2}$

Table 3: Feature Generation for sentence $W_i\ W_{i+1}\ W_{i+2}$. Uni, Bi, Tri and POS denote unigrams, bigrams, trigrams and parts-of-speech tags respectively.

We extracted the above features (Table 3) from the textual content of different structural units (title, initial post, reply posts) of the threads. First, we built a basic model where we used only the text of the titles (denoted by t) for classification. Then, we used the text of initial posts and reply posts. We experimented with the following four settings: title (t), initial post (I), title and initial post (t+I), entire thread (t+I+R).

[5] http://www.personal.psu.edu/pxb5080/dataSubj.html

[6] http://search.cpan.org/dist/Lingua-EN-Tagger/Tagger.pm

4.3 Experimental Setting

We used various supervised learning algorithms to perform our classification experiments. We experimented with Multinomial NaiveBayes, Support Vector Machines, Logistic regression, Bagging, Boosting and Decision Trees. Logistic regression gave the best results with our features whereas in case of the baseline lexical features, Multinomial NaiveBayes outperformed all the other classifiers. We used Weka data mining toolkit with default settings to conduct our experiments (Hall et al., 2009). To evaluate the performance of our classifiers, we used macro-averaged precision, recall and F-1 measure. For a metric M, macro-average M_{mav} is calculated by taking weighted average of M for both subjective and non-subjective classes for each fold and then taking mean of the weighted averages across all the folds. For n-fold cross validation, M_{mav} is mathematically defined as follows:

$$M_{mav} = \frac{1}{n}\sum_{i=1}^{n}\frac{n_{s_i}M_{s_i} + n_{ns_i}M_{ns_i}}{n_{s_i} + n_{ns_i}} \quad (1)$$

where n_{s_i} and n_{ns_i} are the number of subjective and non-subjective threads in the test set in the i^{th} fold . M_{s_i} and M_{ns_i} are the values of metric M for the subjective and the non-subjective classes, respectively, in the i^{th} fold . We used $n = 10$ in our experiments. We use F-1 measure to compare performances of two classifiers. A naive baseline that classifies all the threads in the majority class will have a macro-averaged precision, recall and F-1 measure of 0.457, 0.676 and 0.545 respectively for Trip–Advisor and 0.394, 0.628 and 0.485 respectively for Ubuntu. We consider these values to be the lower bounds for any of our methods.

4.4 Classification Results

4.4.1 Baseline Results

Table 4 reports the results of the subjectivity classification obtained from different baselines. A total of 24 experiments (using the six types of features for the four settings (t, I, t+I, t+I+R)) were conducted for both the datasets. From the table, we note that titles give fair estimate of thread's subjectivity and initial posts (I) provide a better estimate. Incorporating text from initial post and title (t+I) improves the performance slightly over the initial post (I) setting. Further, adding the text of reply posts (t+I+R) gives the best performance. This is expected as titles only contain some keywords related to the discussion topic whereas initial posts contain the entire problem of discussion and reply posts constitute a major portion of the discussion in the thread. We also note that unigrams+bigrams+POS and unigrams+bigrams consistently perform better than the other features for all the settings except for the title (t) setting where unigrams and unigrams+POS performed the best.

4.4.2 Performance of the Proposed Classification Model

Table 5 reports the results of our classification model. We achieve an overall accuracy of 77.01%, a precision of 0.763 and an F-1 measure of 0.764 on the Trip–Advisor dataset and an overall accuracy of 70.05%, a precision of 0.692 and an F-1 measure of 0.684 on the Ubuntu dataset. We further analyze the classification performance of our classifier by analyzing its performance for the two classes. Table 6 reports precision, recall and F-1 measure for subjective and non-subjective classes for both the datasets. We observe that the classification performance for the subjective class is better than the non-subjective class for the Trip–Advisor dataset. This

Trip–Advisor												
	t			I			t+I			t+I+R		
	Pr.	Re.	F-1	Pr.	Re.	F-1	Pr.	Re.	F-1	Pr.	Re.	F-1
U	0.618	0.644	0.625	0.662	0.665	0.664	0.671	0.673	0.672	0.703	0.716	0.706
U+B	0.56	0.586	0.565	0.713	0.718	0.715	0.700	0.704	0.702	0.738	0.747	**0.723**
U+B+T	0.627	0.55	0.564	0.703	0.658	0.669	0.697	0.655	0.666	0.721	0.732	**0.723**
U+POS	0.56	0.586	0.565	0.669	0.673	0.671	0.686	0.69	0.688	0.701	0.713	0.704
U+B+POS	0.606	0.616	0.610	0.704	0.711	0.704	0.701	0.709	0.704	0.733	0.741	0.71
U+B+T+POS	0.614	0.522	0.566	0.709	0.67	0.68	0.706	0.675	0.684	0.722	0.736	0.716
Ubuntu												
	t			I			t+I			t+I+R		
	Pr.	Re.	F-1	Pr.	Re.	F-1	Pr.	Re.	F-1	Pr.	Re.	F-1
U	0.546	0.578	0.553	0.652	0.646	0.648	0.649	0.643	0.645	0.694	0.689	**0.691**
U+B	0.551	0.58	0.557	0.662	0.655	0.658	0.659	0.654	0.656	0.688	0.67	0.675
U+B+T	0.548	0.576	0.554	0.656	0.646	0.649	0.657	0.647	0.651	0.696	0.663	0.669
U+POS	0.626	0.647	0.633	0.644	0.638	0.64	0.649	0.641	0.644	0.694	0.688	**0.69**
U+B+POS	0.552	0.564	0.556	0.659	0.652	0.655	0.659	0.652	0.655	0.72	0.696	**0.701**
U+B+T+POS	0.551	0.557	0.554	0.646	0.631	0.636	0.64	0.63	0.633	0.705	0.657	0.662

Table 4: Classification performance of different baseline features (Table 3) extracted from different structural components of the forum threads. t, I and R are title, initial post and set of all reply posts of a thread respectively. U, B, T and POS are unigrams, bigrams, trigrams and parts-of-speech tags respectively.

can be attributed to the significantly more number of subjective threads than non-subjective threads (refer to Section 4.1) in the Trip–Advisor dataset and hence more patterns for the classifier to learn for the majority (subjective) class leading to the better performance for that class. Similarly, for the Ubuntu dataset, we see a better performance for the non-subjective class whose number of threads are significantly more than that of the subjective class.

Next, we compare the performance of our classification model with the baselines. As can be seen from Table 6, our classification model outperforms the best performing baseline (U+B for the t+I+R setting, refer to Table 4), thus outperforming all the 24 baselines, for the Trip–Advisor dataset. For the Ubuntu dataset, our model achieves an F-1 measure of 0.684 whereas the best performing baseline (U+B+POS for the t+I+R setting, refer to Table 4) achieves an F-1 measure of 0.701. In this case, our model outperforms 21 out of the 24 baselines. The other two baselines that achieved a better performance than our model are unigrams (U) for the t+I+R setting and unigrams+POS (U+POS) for the t+I+R setting with an F-1 measure of 0.691 and 0.69 respectively. Thus, we see that we achieve classification performance which is similar to, and, in most cases, better than that obtained from the baseline features by using thread specific features which are much less in number (no. of baseline features is of the order of the size of the vocabulary whereas no. of features in our model = 25.)

Metric	**Trip–Advisor**	**Ubuntu**
Classification Accuracy	77.01%	70.05%
Precision	0.763	0.692
F1-Measure	0.764	0.684

Table 5: Classification results.

	Trip–Advisor			Ubuntu		
	Precision	**Recall**	**F-1**	**Precision**	**Recall**	**F-1**
Subjective class	0.805	0.871	0.837	0.647	0.429	0.516
Non-subjective class	0.675	0.558	0.611	0.718	0.862	0.783
Overall	0.763	0.77	0.764	0.692	0.7	0.684
Best performing baseline	0.738	0.747	0.723	0.72	0.696	0.701

Table 6: Classification performance of the proposed model for subjective and non-subjective classes on the two datasets.

4.4.3 Relative Performance of Different Types of Features

In this subsection, we investigate the effect of different types of features used for the subjectivity classification task. We perform the classification experiment using only one type of feature at a time. Table 7 shows the relative performance of different types of features. We see that, for both the datasets, structural features gave the best performance which confirms our hypothesis that thread structure is a strong indicator of its subjectivity orientation. Lexicon-based and Sentiment features are the second best performing features, outperforming the dialogue act features, for the Trip–Advisor forum whereas for the Ubuntu forum, dialogue act features outperform the two types of features with sentiment features being the worst performing and Lexicon-based features being the third best performing features. This difference in the relative performance of Sentiment and Lexicon-based features across the two forums may be attributed to the difference in the nature of the two forums. Trip–Advisor is a non-technical forum where majority of discussions are subjective in nature and hence there are more number of subjectivity clues and sentiment indication patterns for the classifier to learn, whereas discussions in Ubuntu forum are technical and hence, usually, non-subjective in nature. Further, the combined performance of all the features is better than the performances of all the individual types of features.

	Trip–Advisor			Ubuntu		
Class	**Precision**	**Recall**	**F-1**	**Precision**	**Recall**	**F-1**
Structural	0.741	0.75	0.742	0.692	0.697	0.67
Dialogue Act	0.683	0.703	0.683	0.639	0.654	0.598
Subjectivity Lexicon Based	0.713	0.727	0.699	0.622	0.643	0.569
Sentiment	0.71	0.726	0.699	0.534	0.602	0.525
All	0.762	0.768	0.763	0.692	0.7	0.684

Table 7: Classification results for NYC and Ubuntu datasets obtained using different types of features.

4.4.4 Most Informative Features

We study the importance of individual features by measuring the chi-squared statistic with respect to the class variable. Table 8 shows top 10 features, ranked by their chi-square values. From the table, we note that, for both the datasets, five out of six structural features (ThreadLength, NumPost, AvgPostLength, NumAuthor, InitPostLength) are among the top 10

most informative features which again confirms that a thread's structure is a strong indicator of its subjectivity. We note that the lexicon-based features and the sentiment features have relatively higher ranks in Trip Advisor dataset as compared to the Ubuntu dataset. We also note that, for Trip–Advisor, two of the three lexicon-based features (NumSubReply, NumSubInit) are among the top 10 features whereas for Ubuntu, only one lexicon-based feature (NumSubReply) is ranked among the top 10 features. This observation is consistent with our previous observation where we noted that sentiment and lexicon-based features performed relatively better in Trip–Advisor as compared to Ubuntu and can be attributed to the difference in the nature of the two forums as explained in the previous subsection. Among the lexicon-based features, NumSubReply is the most informative feature which suggests that, for a thread, reply posts are more helpful than initial post and title of the thread in identifying the thread's subjectivity. This is also manifested in case of sentiment features where features corresponding to reply posts (ReplySentiStrngPos, ReplySentiAvgNeg, etc.) are ranked higher than the corresponding features for the initial post (which are not in the top 10 list). These observations are consistent with the results we got from our baselines where we found that incorporating text from reply posts gave the best performance across all the features. We note that, for Ubuntu, there is one dialogue act feature (NumSol) in the top 10 list, whereas for Trip–Advisor, none of the dialogue act features are in the list.

Trip–Advisor	**Ubuntu**
ThreadLength	ThreadLength
NumSubReply	NumPost
AvgPostLength	NumSubReply
NumPost	NumUser
NumUser	AvgPostLength
ReplySentiStrngPos	InitPostLength
ReplySentiAvgNeg	NumSol
InitPostLength	ReplySentiAvgNeg
ReplySentiAvgPos	ReplySentiStrngPos
NumSubInit	ReplySentiStrngNeg

Table 8: Top 10 features ranked by chi-square values for the two datasets.

5 Conclusions and Future Work

In this work, we proposed a supervised machine learning model for subjectivity classification of online forum threads. We used various novel thread-specific features in addition to lexicon-based and sentiment features for the classification task. We evaluated our model by comparing it with various state-of-the-art techniques used for subjectivity classification and showed that our model outperformed them in most of the cases. A major contribution of this work is the introduction of thread-specific features for subjectivity classification of online forum threads which significantly reduces the complexity of the learning model compared to that of the models built on lexical features without compromising the performance of the model. In future, we plan to investigate semi-supervised and unsupervised learning for subjectivity classification of online forum threads. We also plan to use the subjectivity analysis to improve the search in online forums.

References

Aikawa, N., Sakai, T., and Yamana, H. (2011). Community qa question classification: Is the asker looking for subjective answers or not? *IPSJ Online Transactions*, 4:160–168.

Banea, C., Mihalcea, R., and Wiebe, J. (2010). Multilingual subjectivity: are more languages better? In *Proceedings of the 23rd International Conference on Computational Linguistics*, COLING '10, pages 28–36, Stroudsburg, PA, USA. Association for Computational Linguistics.

Banea, C., Mihalcea, R., Wiebe, J., and Hassan, S. (2008). Multilingual subjectivity analysis using machine translation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '08, pages 127–135, Stroudsburg, PA, USA. Association for Computational Linguistics.

Bhatia, S., Biyani, P., and Mitra, P. (2012). Classifying user messages for managing web forum data. In *15th International Workshop on the Web and Databases(WebDB)*.

Bhatia, S. and Mitra, P. (2010). Adopting inference networks for online thread retrieval. In *AAAI 2010, Atlanta, Georgia, USA, July 11-15*, pages 1300–1305.

Bruce, R. and Wiebe, J. (1999). Recognizing subjectivity: a case study in manual tagging. *Natural Language Engineering*, 5(2):187–205.

Duan, H. and Zhai, C. (2011). Exploiting thread structures to improve smoothing of language models for forum post retrieval. *Advances in Information Retrieval*, pages 350–361.

Gurevych, I., Bernhard, D., Ignatova, K., and Toprak, C. (2009). Educational question answering based on social media content. In *Proc. of the 14th International Conf. on Artificial Intelligence in Education*, pages 133–140.

Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., and Witten, I. (2009). The weka data mining software: an update. *ACM SIGKDD Explorations Newsletter*, 11(1):10–18.

Hassan, A., Qazvinian, V., and Radev, D. R. (2010). What's with the attitude? identifying sentences with attitude in online discussions. In *EMNLP 2010*, pages 1245–1255. ACL.

Hu, M. and Liu, B. (2004). Mining and summarizing customer reviews. In *SIGKDD 2004*, pages 168–177. ACM.

Jeong, M., Lin, C.-Y., and Lee, G. G. (2009). Semi-supervised speech act recognition in emails and forums. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 3 - Volume 3*, EMNLP '09, pages 1250–1259, Stroudsburg, PA, USA. Association for Computational Linguistics.

Joty, S., Carenini, G., and Lin, C.-Y. (2011). Unsupervised modeling of dialog acts in asynchronous conversations. In *Proceedings of the Twenty-Second international joint conference on Artificial Intelligence - Volume Volume Three*, IJCAI'11, pages 1807–1813. AAAI Press.

Li, B., Liu, Y., and Agichtein, E. (2008a). Cocqa: co-training over questions and answers with an application to predicting question subjectivity orientation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '08, pages 937–946, Stroudsburg, PA, USA. Association for Computational Linguistics.

Li, B., Liu, Y., Ram, A., Garcia, E., and Agichtein, E. (2008b). Exploring question subjectivity prediction in community qa. In *SIGIR 2008*, pages 735–736. ACM.

Liu, B. (2010). Sentiment analysis and subjectivity. *Handbook of Natural Language Processing,*, pages 978–1420085921.

Ly, D., Sugiyama, K., Lin, Z., and Kan, M. (2011). Product review summarization from a deeper perspective. In *JCDL 2011*, pages 311–314.

Mihalcea, R., Banea, C., and Wiebe, J. (2007). Learning multilingual subjective language via cross-lingual projections. In *ACL*.

Mukund, S. and Srihari, R. K. (2010). A vector space model for subjectivity classification in urdu aided by co-training. In *Proceedings of the 23rd International Conference on Computational Linguistics: Posters*, COLING '10, pages 860–868, Stroudsburg, PA, USA. Association for Computational Linguistics.

Seo, J., Croft, W. B., and Smith, D. A. (2009). Online community search using thread structure. In *CIKM 2009*, pages 1907–1910, New York, NY, USA. ACM.

Somasundaran, S., Wilson, T., Wiebe, J., and Stoyanov, V. (2007). Qa with attitude: Exploiting opinion type analysis for improving question answering in on-line discussions and the news. In *ICWSM 2007*.

Stoyanov, V., Cardie, C., and Wiebe, J. (2005). Multi-perspective question answering using the opqa corpus. In *EMNLP 2005*, pages 923–930. ACL.

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

Thelwall, M., Buckley, K., and Paltoglou, G. (2012). Sentiment strength detection for the social web. *J. Am. Soc. Inf. Sci. Technol.*, 63(1):163–173.

Wiebe, J., Bruce, R., and O'Hara, T. (1999). Development and use of a gold-standard data set for subjectivity classifications. In *ACL 1999*, pages 246–253. ACL.

Wiebe, J. and Riloff, E. (2005). Creating subjective and objective sentence classifiers from unannotated texts. *Computational Linguistics and Intelligent Text Processing*, pages 486–497.

Wiebe, J., Wilson, T., and Cardie, C. (2005). Annotating expressions of opinions and emotions in language. *Language Resources and Evaluation*, 39:165–210. 10.1007/s10579-005-7880-9.

Yu, H. and Hatzivassiloglou, V. (2003). Towards answering opinion questions: separating facts from opinions and identifying the polarity of opinion sentences. In *Proceedings of the 2003 conference on Empirical methods in natural language processing*, EMNLP '03, pages 129–136, Stroudsburg, PA, USA. Association for Computational Linguistics.

Zhai, Z., Liu, B., Zhang, L., Xu, H., and Jia, P. (2011). Identifying evaluative sentences in online discussions. In *Twenty-Fifth AAAI Conference on Artificial Intelligence*.

Natural Language Generation for Nature Conservation: Automating Feedback to help Volunteers identify Bumblebee Species

Steven Blake[1] *Advaith Siddharthan*[1] *Hien Nguyen*[1]
Nirwan Sharma[1] *Anne-Marie Robinson*[2] *Elaine O'Mahony*[3]
Ben Darvill[3] *Chris Mellish*[1] *René van der Wal*[2]

(1) Department of Computing Science, University of Aberdeen, U.K.
(2) Aberdeen Centre for Environmental Sustainability (ACES), University of Aberdeen, U.K.
(3) Bumblebee Conservation Trust, University of Stirling, U.K.

s.blake.08@aberdeen.ac.uk, advaith@abdn.ac.uk, h.nguyen@abdn.ac.uk,
n.sharma@abdn.ac.uk, annierobinson@abdn.ac.uk,
elaine.omahony@bumblebeeconservation.org,
ben.darvill@bumblebeeconservation.org, c.mellish@abdn.ac.uk,
r.vanderwal@abdn.ac.uk

ABSTRACT

This paper explores the use of Natural Language Generation (NLG) for facilitating the provision of feedback to citizen scientists in the context of a nature conservation programme, BEEWATCH. BEEWATCH aims to capture the distribution of bumblebees, an ecologically and economically important species group in decline, across the UK and beyond. The NLG module described here uses comparisons of visual features of bumblebee species as well as contextual information to improve the citizen scientists' identification skills and to keep them motivated. We report studies that show a positive effect of NLG feedback on accuracy of bumblebee identification and on volunteer retention, along with a positive appraisal of the generated feedback.

KEYWORDS: NLG, Natural Language Generation, Educational Application, Bumblebee Conservation, Citizen Science, Generating Feedback.

Proceedings of COLING 2012: Technical Papers, pages 311–324,
COLING 2012, Mumbai, December 2012.

1 Introduction

There is a growing realisation of the potential of digital approaches, including the use of websites and social media, to increase participation in "citizen science", which includes observing and monitoring the natural world. For instance, in the UK, the Open Air Laboratories (OPAL) network (www.opalexplorenature.org) is a large current initiative led by Imperial College, which aims to create and inspire a new generation of nature-lovers by getting people to explore, enjoy and protect their local environment (Silvertown, 2009). Within OPAL, iSpot (www.ispot.org.uk) is an online nature community that connects beginners with experts and fellow enthusiasts. Other groups have explored the use of standard social networking sites to generate public interest and collect data about the distribution of species (Stafford et al., 2010). Publicly available resources include www.thewildlab.org, which provides software for a number of mobile platforms, and www.scienceforcitizens.net, which acts as a forum for citizen scientists to find out about projects they can participate in and for researchers to publicise their projects.

Although digital tools can enthuse the public and be used to enlist (for a short time at least) willing volunteers for nature conservation projects, initiatives such as the above have to contend with at least the following issues:

1. Data quality: participants are generally untrained, and not necessarily motivated to produce high-quality data (Stafford et al., 2010). This is not a problem if a project is primarily an "outreach" activity, but many projects also have specific scientific goals.

2. Retention of volunteers: in order to secure continuing participation, systems need to constantly renew to keep users interested and engaged.

Existing digital support for nature conservation volunteers can only give them feedback and encouragement by allowing them to interact with human experts or by showing them pre-prepared material. Human experts are in short supply, and pre-prepared material is inherently limited in scope. We believe that this latter is a problem, and that the level of interest and motivation for people to participate in ecological monitoring activities is in part a function of the richness of information they are provided with on an ongoing basis. Through the use of a Natural Language Generation (NLG) component, we automate the provision of rich information to address the two key issues listed above: improving the accuracy of volunteer contributed records, and volunteer retention over time.

The Bumblebee Conservation Trust[1] is seeking to map the current distribution of bumblebee species across the UK through a collaborative project with the University of Aberdeen called BeeWatch[2]. BeeWatch allows volunteers from the general public to submit photos of bumblebees they have seen in the wild, along with the location and date of sighting. The submission interface includes an online identification guide (see Fig. 1) to help classify the bumblebee in the photo as one of 22 bumblebee species[3]. Through this interface, the volunteer can select visual features of the bumblebee (types of thorax, abdomen, etc.) to narrow down the possible species. Once the image and the user's identification have been submitted, an expert identifies

[1] http://www.bumblebeeconservation.org

[2] http://bumblebeeconservation.org/get-involved/surveys/

[3] There are actually 24 species of bumblebee in the United Kingdom, but three of these (*Bombus lucorum*, *Bombus cryptarum* and *Bombus magnus*) cannot be reliably distinguished from each other based on visual charecteristics alone. These form a species complex, and for the purposes of BeeWatch they are treated as one species— *Bombus lucorum*, the White-tailed bumblebee.

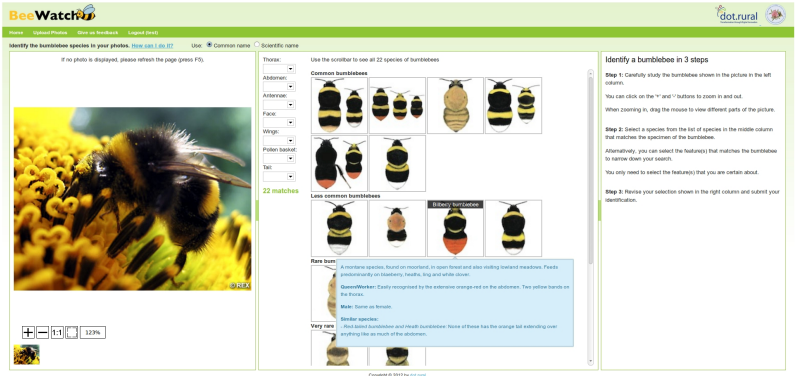


Figure 1: Screenshot of online identification tool

the bumblebee in the photo. Before the current collaboration, this expert communicated the correct identification to the volunteer by email. In the case of incorrect identification, the conservation charity would have liked to provide explanations as to why this was the case, but their experts only rarely found the time to do this. The goal of the NLG component described in this paper is to automate the provision of feedback to the volunteer based on the expert's identification.

Fig. 2 shows an extract of the computer generated feedback for a user who has incorrectly identified a photo as a broken-belted bumblebee. The key element in this automatically generated extract is the comparison between the user-identified species and the correct species as identified by the expert. Although there are only 22 species of bumblebees in the UK, this means that there are almost 500 bee comparisons to be generated, and the overall number of possible texts would be orders of magnitude greater when contextual information (e.g., based on location and time of sighting) is included in the feedback. On the other hand, the NLG

> Thank you for submitting this photo. Our expert identified the bee as a Heath bumblebee rather than a Broken-belted bumblebee. You correctly identified the face, the wings and the pollen basket; however, the abdomen (rear body) and the thorax (central body) are different. Although some of these features may not be visible in your photograph, the following advice might be helpful for next time you are in the field.
>
> The Heath bumblebee's thorax is black with two yellow to golden bands whereas the Broken-belted bumblebee's thorax is black with one yellow to golden band. The Heath bumblebee's abdomen is black with one yellow band near the top of it and a white tip whereas the Broken-belted bumblebee's abdomen is black with one yellow band around the middle of it and a white to buff tip.

Figure 2: Example of NLG feedback explaining why a user identification was incorrect

Thank you for submitting this photo. You have correctly identified the bumblebee as a White-tailed bumblebee.

As you are already aware all individuals of this species have two yellow bands and a white tail. Although they can be difficult to separate from Buff-tailed workers, the tail of the White-tailed bumblebee is pure white, with a complete absence of any buff-coloured hairs. The colour of the two yellow bands is brighter and more lemon than that seen in the Buff-tailed bumblebee.

Figure 3: Example of NLG feedback when a user has made a correct identification

system relies on just a model of each of the 22 species found in the UK. So, whereas it would be infeasible to produce a human-authored text for each possible situation, this use of NLG can even scale up to, for example, bumblebees in other countries (there are over 250 known species worldwide), or likewise, other genera.

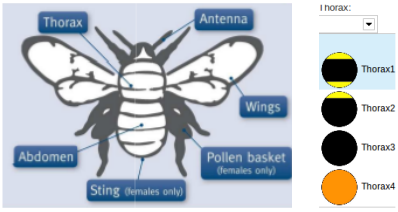
We also use feedback to reinforce relevant information when a user has made a correct identification. Fig. 3 provides an extract from computer generated feedback that illustrates this.

A key contribution of this paper is an evaluation of the effect of the automatically generated text on volunteers. It is unusual for an NLG application (that typically aid decision making in the workplace) to have as diverse a set of users as we do. Our results demonstrate the potential for NLG in real world applications targeting members of the general public. The automation of feedback provision, as described in the rest of this paper, has removed a major bottleneck for the charity we are working with, and is allowing them to scale up what was initially just a public engagement exercise generating around 200 records a year into an initiative that has produced 650 records a month.

2 Related work

Much of the recent focus within NLG applications research has been on data-to-text systems, which typically generate summaries of technical data for professionals such as engineers or nurses (Goldberg et al., 1994; Theune et al., 2001; Portet et al., 2009). These are capable of generating high quality texts; e.g., offshore oil rig workers preferred weather forecasts generated by the SumTime system to texts written by professional human forecasters (Sripada et al., 2003). There is some previous work on the use of data-to-text for lay audiences; e.g., generating narratives from sensor data for automotive (Reddington et al., 2011) and environmental (Molina et al., 2011) applications, generating personal narratives to help children with complex communication needs (Black et al., 2010), and summarising neonatal intensive care data for parents of premature babies (Mahamood et al., 2008).

There are some notable examples of NLG systems that make use of structured textual records rather than numeric data. Peba-II (Milosavljevic, 1997) was an online animal encyclopedia that provided descriptions and comparison of animals using HTML pages. The Power (Daley et al., 1998) and ILEX (O'Donnell et al., 2001) systems in the virtual museum domain dynamically generated descriptions of museum objects based on the user's discourse history and user model. Dial Your Disc (Van Deemter and Odijk, 1997) generated spoken monologues about classical music, with the aim of generating engaging texts, attempting to keep its users amused by



(a) (b)

Figure 4: Bumblebee Model: (a) Visual bumblebee features, (b) List of Thorax types

focusing on the expression of *unusual* content. Our work shares commonalities with these systems, in that it targets non-expert audiences and has educational goals.

The main mechanism employed by us in the current work is the use of comparisons for generating feedback. This builds on a body of previous work. Milosavljevic (1997) described comparisons as a useful tool for augmenting the user's existing knowledge with new knowledge. Karasimos and Isard (2004) showed that texts which contained comparisons and aggregations helped readers retain more information and perform better on factual recall while also finding these texts more interesting and pleasant to read. Later, Marge et al. (2008) performed a similar experiment to isolate the effects of comparison from those of aggregation. These two experiments provide evidence that comparisons can help to improve the knowledge of users on a given domain, and are a basis for our work.

3 Implementation of the NLG module

Our NLG system uses the architecture proposed by Reiter and Dale (2000) and is compatible with a wide range of work within the field. There are three main components in this architecture; a document planner, a micro planner and a surface realiser. Additionally, the document planning makes use of a domain model. We describe our implementation in this section and the evaluation of the system in Sections 4 and 5.

3.1 Domain model

Our domain model is primarily the representation of each bumblebee species as a set of visual features, as implemented in the online identification guide (see Fig. 1). The tool contains information about the thorax and abdomen colour patterns, face length, wing type and the presence of a pollen basket (shown in Fig. 4 (a)). The model includes a textual description of each visual feature. For example, the thorax with visual pattern `Thorax2` in Fig. 4 (b) has the associated text: "black with one yellow to golden band".

Note that the domain model used in the identification tool only covers the main bumblebee features. This is important for usability, and also makes it easier to port to other insect species. However, this means that some bumblebee species are indistinguishable in terms of the modelled features. To distinguish such species, each model of a bumblebee species has associated with it a list of similar bumblebee species and a description of how it can be distinguished from them. An example of this similarity can be seen between the Moss carder bee and the Common carder bee (shown in Fig. 5(a)). They share the same values for all their domain features (e.g., both have thorax type 4, etc.), and their only distinguishing feature is the black hairs on the

(a) (b)

Figure 5: Difficult cases: (a) Similarity of Moss Carder and Common Carder Bumblebee, (b) Differences between castes for the White-tailed bumblebee

abdomen of the Common carder bee. The model also contains contextual information; for instance, where the species is usually found, what time of year they are usually seen and how rare they are in the UK.

A further point worth noting is that bumblebees have a caste system of queens, males (drones) and females (workers). These castes can sometimes have different visual features (e.g., see Fig. 5(b)). We do not model castes explicitly because it is difficult for novice recorders to identify caste, and there is thus limited gain in using caste information in the feedback. As an alternative, we allow features for the abdomen and thorax to sometimes have multiple values for the same species.

3.2 Document planner

The NLG system described here primarily uses comparisons of bumblebee species to improve the user's identification skills. Milosavljevic (1997) introduced three types of comparison that are useful for educational purposes: direct, clarifactory and illustrative. Briefly, direct comparison is used to describe to the user attributes that two objects share and the attributes that distinguish them. This comparison is bi-focal; neither object is more important than the other. Clarifactory comparison is used to describe an object by distinguishing it from a similar object that it may be confused with. This confusion usually arises from the two objects sharing similar attributes or features. Illustrative comparison is used to describe an attribute (or attributes) of an object by referring to the same attribute(s) of another object that the user is familiar with.

The main methods used in this work are direct comparisons, which allows the user to understand the differences between species, and clarifactory comparisons to distinguish similar species (those with identical features in our visual model). Illustrative comparison is currently not used but could be employed when the system has access to a user's history of interaction with the system. For example if the user had identified a Garden bumblebee previously and has just identified a Heath bumblebee, illustrative comparison could be used to explain that the colour pattern of the Garden bumblebee is similar to that of the Heath bumblebee. However, users typically submit records only once or twice a week, so past submissions might not be salient.

Our document planner first determines content, and then decides document structure.

3.2.1 Content determination

Before the system can decide what to include it needs a data representation to store this content. This representation will, later on, map to a sentence (or a group of related sentences) that will be generated in the final output. We represent content using *messages*, i.e. Java objects that contain domain specific entities and information.

The creation of message objects is handled by the message generator at run-time. The message generator looks at the input the system has been given and then analyses it to determine which of a set of content determination rules match. These rules determine which messages are created and what they contain.

A `thankYou` message is always generated with canned text content thanking the user for the submission. Then a message is generated that holds information about whether the user has identified the bee correctly or not. For example, if the identification by the user is incorrect then a `result` message is generated containing the value "incorrect" along with the bumblebee species as identified by the expert and by the user. Further along the line, this message will be realised as a sentence such as "*Our expert identified the bee as a Heath bumblebee rather than a Broken-belted bumblebee*" (see Fig. 2). If the identification is correct, a similar message is generated with the value "correct", which could later be realised as a sentences such as "*You have correctly identified the bee as a Heath bumblebee*".

If the user has incorrectly identified the bumblebee species, the system needs to explain to the user why the identification was incorrect. This is handled through the `features` message. This message compares the visual features of the two species and stores similar features and dissimilar features as two separate sets. Due to the fact that some features (e.g., thorax and abdomen) can have multiple values in a bumblebee model due to variation between castes, the similarity of features is determined through set intersection. A feature goes into the "similar" list, if there is any value that is common to both bumblebee models. This message is later realised as a sentence such as "*You correctly identified the face, the wings and the pollen basket; however, the abdomen (rear body) and the thorax (central body) are different.*"

One of our communicative goals is to help the user improve their identification skills. This is pursued by explaining to the user why they were incorrect (if that were indeed the case). For each feature that is dissimilar, a `featureIdentification` message is created that contains the values of this feature in both bumblebee models. These messages will be used to explain the differences between the two species using direct comparison.

The message generator also searches the "similar species" list in the model of the bumblebee as identified by the expert. If the bumblebee as identified by the user is found, then a `similarSpecies` message is created, containing the canned text from the model describing the distinguishing features using clarifactory comparison.

If there is any contextual information available about the identified species, this is realised through a `contextualInformation` message. In our current implementation, we only use a summary of the known habitat and behaviour of the species in a non-adaptive manner. We plan to use such information more intelligently in the future, notably to help contextualise a user's record.

When all the messages have been constructed, they are passed on to the Document Structuring component.

3.2.2 Document structuring

The messages are structured in a specific way by the document planner using schemas, resulting in a document plan which takes the form of a tree. Groups of related messages are represented by the internal nodes while the messages themselves are the terminal nodes. In this case, the internal nodes will represent actual textual structures such as paragraphs. These structures are

represented in Java as objects that are instantiated from classes. The design of these structures is heavily influenced by the structures proposed by Reiter and Dale (2000).

The exact structure is determined by how many messages have been generated. There are four possible groups that the messages can be placed in: the intro group, the features group, the similar species group and the contextual information group. The intro group will always be present but the other three may or may not be present depending on the outcome of the identification and the two species that are being compared. For example, if two species are not classed as similar species then the similar species group will not be present at all. The schema loops through the entire message list and decides what groups are needed based on the type of messages present.

3.3 Microplanner

The microplanner is the second stage of the generation architecture. Here, the document plan produced by the first stage, the document planner, is refined to produce a text specification. This includes phrase specifications and their aggregation into sentences. The phrase specification is structured in a way that it can be realised by the surface realiser. We use SIMPLENLG (Gatt and Reiter, 2009), and the design of these structures is therefore influenced by the functionalities of the SIMPLENLG library.

To allow for the generation of more complex sentence structures and sentences that are easier to understand, the microplanner also carries out aggregation. The aggregation performed in this system focuses on the formation of sentences. For example, in the paragraph that deals with comparisons (see Fig. 2), the system knows that it will be comparing different features. This means that the phrase specifications can be aggregated through subordination, using conjunctions such as “whereas” or “although”.

The descriptions of the visual features, specifically the abdomen and thorax, are also analysed to check if they are a candidate for aggregation. These two features can have multiple values due to morphological differences among castes within a species. Our system does not explicitly model castes and therefore when describing a feature that has multiple values the system has to use disjunctions. For example, the White-tailed bumblebee has two possible thorax values: `thorax1` (“black with two yellow to golden bands”) and `thorax2` (“black with one yellow to golden band”; see Fig. 5). Rather than generating:

> “the thorax is either black with two yellow to golden bands or black with one yellow to golden band,”

we use aggregation to generate the more natural sounding succinct phrase:

> “the thorax is black with either one or two yellow to golden bands.”

Once the microplanner has built all the sentence specifications (by the processes mentioned previously) and organised them into paragraphs, the resultant text specification document is passed on to the surface realiser.

3.4 Surface realiser

The role of the surface realiser is to convert the text specification received from the microplanner into text that the user can read and understand. This includes linguistic realisation (converting

the sentence specifications into sentences) and structural realisation (structuring the sentences inside the document). Both the linguistic and structural realisations are performed by using functionalities provided by the SIMPLENLG realiser library (Gatt and Reiter, 2009).

4 Experiment 1

This section details the evaluation of how volunteer recorders perceive the NLG, and the effect of feedback on their bumblebee identification accuracy. Later, in Experiment 2 (Section 5), we study the effect of NLG feedback on volunteer retention. For Experiment 1, we distinguish between two types of feedback, which differ with respect to the richness of information they communicate:

Type 1: Acknowledgement of submission + Correct Answer

Type 2: Type 1 + Feedback based on comparisons of visual features

4.1 Method

We designed an evaluation interface that steps each participant through twenty distinct images of bumblebees for which we have an expert identification. There were seven species of bumblebee represented in this data set, each occurring between one and four times. At each step, the participant identified the bumblebee species in the photo using an identification guide, and then received feedback on their identification. All participants viewed the photos in the same order. Upon completing the identification task for all 20 photos, each user was asked:

- To rate how helpful they found the feedback (on a scale of 1–5)
- What they thought of the exercise in general (free text)
- What extra types of information they would like to see in the feedback (free text)?
- What information they would like to see removed (free text)

To test the effect of feedback type on recording accuracy, we randomly divided our participants into two groups. Group A always received Type 1 feedback, while Group B always received the richer Type 2 feedback[4]. 48 participants completed Experiment 1, 21 in Group A and 27 in Group B. All participants were undergraduate students studying biology at the University of Aberdeen and none had prior experience with bumblebee identification. The use of naive participants was deliberate. Most of the volunteers to the BEEWATCH program start out as naive participants, and the role of training through feedback about identification features is likely to be most useful at this stage.

4.2 Results

4.2.1 Effect of feedback on accuracy

The accuracy of bumblebee identification for both groups was analysed. We expected that initially there would be no difference between the two groups, but that over time, Group B would improve their identification accuracy faster than Group A due to the more informative

[4]There is a further Type 3 feedback that our system generates, which includes contextual information about the bumblebee species. However this is mainly useful for identification in the real world, and we did not consider it for Experiment 1. We use Type 3 feedback later in Experiment 2, which was conducted with volunteers submitting photos to the live BEEWATCH website

Identifications	Acc(Type2) - Acc(Type1)	Significance of Difference
images 1-10	-2.7%	p=.55
images 11-20	7.4%	p=.10

Table 1: Accuracy for identifications 1-10 and 11-20

feedback received. Table 1 shows that the difference in accuracy between the two groups for the first ten identifications is minimal. For the last ten identifications, however, there is a bigger difference between the means of the two groups, and the group receiving NLG feedback is performing better by 7.4% points.

To understand the effect of Type 2 feedback in more detail, we performed a generalised linear mixed model fit by the Laplace approximation. The dependent variable was Accuracy (whether a photo was identified correctly). The independent variables were Time (the order of presentation of photos) and Condition (Type 1 or Type 2 feedback). We expected that there would be differences between participants and that some bee species would be easier to identify than others. To accommodate these expectations, Participant and BeeSpecies were included as random factors in the model.

We found a significant main effect of Time (z=-1.768; p=0.0423) and Condition (z=2.03l; p=0.017); i.e, both groups improved over time, and Group B was overall more accurate. More importantly, we also found a strong interaction between Time and Condition (z=-3.260; p=0.001); i.e., the accuracy of Group B increased faster over time than Group A. Thus, the richer Type 2 feedback proved useful for improving recorder skills over time.

The graph of the difference in accuracy between the two groups over time in Fig. 6 illustrates this finding. Positive values indicate that the mean accuracy of Group B is greater while negative values indicate that the mean accuracy of Group A is greater. The mix of positive and negative values for the first fifteen identifications show that neither of the groups are consistently more accurate than the other. However, the last five identifications show continually positive values, indicating that the richer feedback received by Group B was beginning to take effect.

These results are consistent with those reported in Karasimos and Isard (2004) and Marge et al. (2008), which collectively suggest that the use of comparison helps users to retain information.

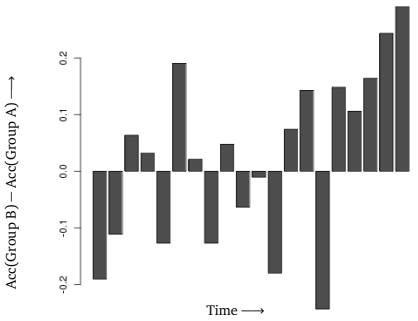


Figure 6: Graph of difference in accuracy between groups over time

Type 1 (no NLG)	Type 2 (NLG)
3.09	3.85

Table 2: Helpfulness of feedback: Mean score for each condition

4.2.2 Feedback helpfulness

One of the questions that participants were asked was to rate how helpful they found the feedback on a scale of 1–5. The NLG feedback was perceived to be significantly more helpful (t-test; t=2.78; p <0.01), as shown in Table 2.

The 21 participants in Group A who received Type 1 feedback overwhelmingly requested more feedback in their free text answers:

- Would like to have been given information on why their identification was incorrect (14 participants)
- Would like none of the information to be removed (12 participants)
- Would like to see more information (3 participants)
- Thought there was a lack of information (3 participants)
- Would like to have been told some facts about the bumblebee species that had been identified (2 participants)

From the feedback listed above, it is clear that participants would like to see more information than just a "correct/incorrect" response; specifically, explanation as to why their identification was incorrect was requested my most participants. There was also a small proportion who would like to be presented with contextual information about the bumblebee identified.

The 27 participants in Group B who received Type 2 feedback were more satisfied with the richness of information in the feedback:

- Would like none of the information to be removed (11 participants)
- Thought that no more information should be added and that the level of detail was sufficient (8 participants)
- Found the information to be useful (5 participants)
- Would like to see comparative pictures between the two bumblebee species (2 participants)
- Some comparisons didn't go into a lot of detail, specifically between similar species (2 participants)
- Face shape was rarely visible so did not help in identification (2 participants)
- Would like to see some contextual information about the bumblebee species (rarity, life history etc) (1 participant)

It is clear from the user feedback that users were more or less satisfied with Type 2 feedback, and that Type 1 users would have preferred Type 2 feedback. Further, a small proportion from both groups wanted more contextual information. As we report next, the inclusion of such information in Type 3 feedback has a positive effect on volunteers.

5 Experiment 2

The experiment described above showed that the richer feedback provided by our NLG component helped improve recorder accuracy. The other key issue we are interested in is volunteer motivation. We present an initial evaluation of this using our live system, by comparing the number of submissions made by users who receive three different types of feedback (volunteer return rates).

Feedback	No. of Submissions	No. of Users	Submissions/User	χ^2
Type 1	356	110	3.23	
Type 2	412	123	3.35	$p = 0.04$
Type 3	542	104	5.21	$p < 0.0001$

Table 3: Number of submissions by feedback type

5.1 Method

The NLG component described in this paper is being used live by the Bumblebee Conservation Trust since June 26 2012. When a photograph and volunteer identification of the bumblebee therein are submitted, the information is logged in a database. Periodically an expert reviews the database using a special administrator tool. When an expert calls up a particular submission and enters their own identification of the bumblebee in the photo, the NLG text is automatically provided. The expert can then edit this text if necessary (to address a question asked by the recorder, for instance) and then clicks on a button to send the feedback by email. This interface has dramatically increased the throughput of the experts, enabled them to spend more time on difficult cases, and giving the charity the confidence to publicise the project in the media and scale it up in size. The interface also means that the expert feedback reaches volunteers quicker, which presumably helps to motivate users and encourage further submissions.

When users register for the live system, they are randomly allocated to one of the three groups, by dividing their unique user IDs by 3 and using the remainder. We thus expect similar numbers of users in each group. As not everyone who registers submits records, group sizes are not identical (see Table 3). Each group always received feedback in only one of the following three types:

Type 1: Acknowledgement of submission + Correct Answer

Type 2: Type 1 + Feedback based on comparisons of visual features

Type 3: Type 2 + Contextual information

For the experiment, the administration interface made the experts aware of the three feedback conditions and which users were in which group. The automatic NLG feedback reflected the feedback condition and the experts were instructed to only make edits compatible with the prevailing condition.

5.2 Results

We report preliminary results from the first 2 months of the system going live. We received 1310 submissions from 337 participants during this period. Table 3 shows the number of submissions by experimental condition. There was a significant difference in submission numbers ($\chi^2 = 41.338$; 2 degrees of freedom; $p < 0.0001$) for the three treatment groups. The table shows the pair-wise significance for differences of Type 2 and 3 from Type 1 feedback.

As users who received the richer Type 2 or Type 3 feedback submitted more records on average than users who received Type 1 feedback, it appears that increasing the richness of information provision through NLG feedback has a positive effect on return rates of participants to the website. However, this analysis is preliminary and these figures are potentially skewed by the presence of a small number of dedicated volunteers in each feedback group.

On average, users submitted 3.7 photos during this period. This is insufficient to test improvement in accuracy from the live system at this point in time.

Conclusion and perspectives

We have described an NLG system that is being used by a nature conservation charity for a citizen science initiative. The automation of feedback provision has removed a major bottleneck for the charity, and has allowed them to scale up what was initially just a public engagement exercise generating around 200 records a year into an initiative that has produced 650 records a month. We are also investigating crowd sourcing models to reduce the time commitments on experts even further and allow a further scaling up of the initiative. These models work better with more accurate identifications by individuals; thus improving recorder accuracy is vital.

Our results show that the feedback generated by the NLG system described in this paper helps users to improve their identification skills faster that those who only receive the correct answer as feedback. Users also found the feedback produced through NLG more helpful than the simple feedback with no NLG elements, as evident from both qualitative and quantitative data reported for Experiment 1, and more motivating, as evident from the return rates reported for Experiment 2.

Acknowledgments

This research was supported by an award made by the RCUK Digital Economy programme to the dot.rural Digital Economy Hub; award reference: EP/G066051/1. We would also like to thank the Bumblebee Conservation Trust for making available their resources, including the use of their organisational infrastructure to recruit volunteers, the provision of experts to identify submitted records, and their contributions towards the testing of the NLG component. Further, this research has depended on records submitted by volunteer recorders participating in the BeeWatch programme.

References

Black, R., Reddington, J., Reiter, E., Tintarev, N., and Waller, A. (2010). Using nlg and sensors to support personal narrative for children with complex communication needs. In *Proceedings of the NAACL HLT 2010 Workshop on Speech and Language Processing for Assistive Technologies*, pages 1–9. Association for Computational Linguistics.

Daley, R., Greeny, S. J., Milosavljevicz, M., Parisz, C., Verspoory, C., and Williamsy, S. (1998). The realities of generating natural language from databases. In *Proceedings of the 11th Australian Joint Conference on Artificial Intelligence*.

Gatt, A. and Reiter, E. (2009). Simplenlg: A realisation engine for practical applications. In *Proceedings of ENLG-2009*.

Goldberg, E., Driedger, N., and Kittredge, R. (1994). Using natural-language processing to produce weather forecasts. *IEEE Expert*, 9(2):45–53.

Karasimos, A. and Isard, A. (2004). Multi-lingual evaluation of a natural language generation system. In *Proceedings of the Fourth International Conference on Language Resources and Evaluation*.

Mahamood, S., Reiter, E., and Mellish, C. (2008). Neonatal intensive care information for parents an affective approach. In *Computer-Based Medical Systems, 2008. CBMS'08. 21st IEEE International Symposium on*, pages 461–463. IEEE.

Marge, M., Isard, A., and Moore, J. (2008). Creation of a new domain and evaluation of comparison generation in a natural language generation system. In *Proceedings of the Fifth International Language Generation Conference*.

Milosavljevic, M. (1997). Augmenting the user's knowledge via comparison. In *In Proceedings of the 6th International Conference on User Modelling*.

Molina, M., Stent, A., and Parodi, E. (2011). Generating automated news to explain the meaning of sensor data. *Advances in Intelligent Data Analysis X*, pages 282–293.

O'Donnell, M., Mellish, C., Oberlander, J., and Knott, A. (2001). Ilex: An architecture for a dynamic hypertext generation system. *Natural Language Engineering 7*.

Portet, F., Reiter, E., Gatt, A., Hunter, J., Sripada, S., Freer, Y., and Sykes, C. (2009). Automatic generation of textual summaries from neonatal intensive care data. *Artificial Intelligence*, 173(7-8):789–816.

Reddington, J., Reiter, E., Tintarev, N., Black, R., and Waller, A. (2011). "Hands Busy, Eyes Busy": Generating Stories from Sensor Data for Automotive applications. In *Proceedings of IUI Workshop on Multimodal Interfaces for Automotive Applications*.

Reiter, E. and Dale, R. (2000). *Building Natural Language Generation Systems*. Cambridge University Press, Cambridge, UK.

Silvertown, J. (2009). A new dawn for citizen science. *Trends in Ecology & Evolution*, 24(9):467–471.

Sripada, S., Reiter, E., and Davy, I. (2003). SumTime-Mousam: Configurable marine weather forecast generator. *Expert Update*, 6(3):4–10.

Stafford, R., Hart, A., Collins, L., Kirkhope, C., Williams, R., Rees, S., Lloyd, J., and Goodenough, A. (2010). Eu-Social Science: The Role of Internet Social Networks in the Collection of Bee Biodiversity Data. *PloS one*, 5(12):e14381.

Theune, M., Klabbers, E., de Pijper, J., Krahmer, E., and Odijk, J. (2001). From data to speech: a general approach. *Natural Language Engineering*, 7(01):47–86.

Van Deemter, K. and Odijk, J. (1997). Context modeling and the generation of spoken discourse. *Speech Communication*, 21(1-2):101–121.

Studying the effect of input size for Bayesian Word Segmentation on the Providence Corpus

Benjamin Börschinger[1,3] *Katherine Demuth*[2] *Mark Johnson*[1]
(1) Department of Computer Science, Macquarie University
(2) Department of Linguistics, Macquarie University
(3) Department of Computational Linguistics, Heidelberg University
benjamin.borschinger@mq.edu.au, katherine.demuth@mq.edu.au,
mark.johnson@mq.edu.au

ABSTRACT
Studies of computational models of language acquisition depend to a large part on the input available for experiments. In this paper, we study the effect that input size has on the performance of word segmentation models embodying different kinds of linguistic assumptions. Because currently available corpora for word segmentation are not suited for addressing this question, we perform our study on a novel corpus based on the Providence Corpus (Demuth et al., 2006). We find that input size can have dramatic effects on segmentation performance and that, somewhat surprisingly, models performing well on smaller amounts of data can show a marked decrease in performance when exposed to larger amounts of data. We also present the data-set on which we perform our experiments comprising longitudinal data for six children. This corpus makes it possible to ask more specific questions about computational models of word segmentation, in particular about intra-language variability and about how the performance of different models can change over time.[1]

KEYWORDS: word segmentation, language acquisition, resources, Bayesian modelling, unsupervised learning.

[1]The corpus and the code to run the experiments is available at http://web.science.mq.edu.au/~bborschi/.

Proceedings of COLING 2012: Technical Papers, pages 325–340,
COLING 2012, Mumbai, December 2012.

1 Introduction

Segmenting a stream of sounds into discrete words is one of the first tasks that children acquiring their native language have to tackle. Computational models of word segmentation enable us to study this problem in a controlled and detailed manner, allowing for example for an examination of the usefulness of different kinds of cues or different learning strategies. Just as important as the actual models, however, is the adequacy of the input used to evaluate them — if we are interested in answering questions about human language acquisition, the data we evaluate our models on needs to be comparable to what children are likely to have access to. To this end, several datasets of phonemically transcribed child directed speech (CDS) have been constructed in several languages, ranging from English to Italian, Polish, Sesotho and Chinese (Brent and Cartwright, 1996; Gervain and Erra, 2012; Boruta and Jastrzebska, 2012; Johnson, 2008a; Johnson and Demuth, 2010). In addition to cross-linguistic variation, however, adequate computational models also need to handle language-internal variation along several dimensions, a topic that has so far received little interest. In this paper, we look at a basic point of variation, namely the actual size of the input to the learner. The longer children are exposed to language, the more data they are exposed to and the better at their language they become, something one would expect from adequate models of language acquisition as well.

We run our experiments on a novel dataset that contains longitudinal data for six children from the Providence Corpus (Demuth et al., 2006). It has two advantages over the current defacto standard for word segmentation studies for English, the Bernstein-Ratner-Brent corpus (Brent, 1999, in the following, BRB Corpus). First of all, it cleanly separates the CDS that is directed at different children whereas the BRB Corpus contains data from 9 different children with no clear indication of the different portions. In addition, recording for some of the children in the BRB corpus began as late as month 21 and for others as early as month 13, making the data available for some of the children hard to compare. In contrast, the Providence Corpus provides data for all of the children starting from month 16 at the latest and starting from month 11 at the earliest and thus constitutes a much more homogeneous data set. Finally, the overall size of the BRB Corpus with a little less than 10,000 utterances altogether is relatively small, in particular if one considers the individual sub-parts which, on average, have only around 1,100 utterances; in contrast, our dataset contains more than 90,000 CDS utterances in total and spans a period of several months for all of the children. This makes it possible to both compare inter-child variability in word segmentation across comparable situations and to study developmental changes in individual children over a period of several months. As such, the resource will allow researchers to ask a wider range of questions than is currently the norm, in particular with respect to the study of incremental models that have recently received a lot of interest (Pearl et al., 2011; Börschinger and Johnson, 2012; Phillips and Pearl, 2012). While for our own experiments on the effects of input size we focus on one of the six sub-corpora of the dataset, we describe and make available the full data so as to enable other researchers to take advantage of this new resource as well.

The contribution of this paper is two-fold. We present a new CDS corpus for computational word segmentation that is derived from the Providence Corpus (Demuth et al., 2006), comprising data from six different children that have been collected in comparable situations over several months. The second and major contribution of our paper is the identification of a so far unreported "overlearning" effect of certain word segmentation models that runs counter to the plausible intuition that more data should lead to better learning outcomes. We also discuss these findings and propose an explanation for the behaviour exhibited by different models, highlighting the

importance of linguistic structure for computational models of language acquisition. The outline of the paper is as follows. First, we provide background about the original Providence Corpus, our way of phonemically transcribing it and the properties of the new data-set we created. In section 3, we introduce the models of word segmentation which we examine with respect to the effect of input size in section 4. Section 5 discusses our findings and the final section concludes.

2 The Providence Corpus

The Providence Corpus was collected during 2002-2005 from participants in southern New England. It contains longitudinal audio/video recordings of 6 monolingual English-speaking mothers and their children from approximately 1-3 years during spontaneous interactions at home. The children included 3 boys (Alex, Ethan, William) and 3 girls (Lily, Naima, Violet). Each was recorded for approximately 1 hour every 2 weeks beginning at the onset of first words. Two of the girls have denser corpora, with weekly recordings from 1;3-2;10 (Naima) and 2;0-3;0 (Lily), and Naima's recordings tended to be 1.5 hours long. There is therefore more data for this mother and child. Lily's mother also talked quickly; there is therefore much data from Lily's mother as well. Recording began around one year or once the parent reported that the child was producing approximately four words.

Digital audio/video recordings took place in each child's home. Although parent and child could move freely about, the video information was useful in determining the context of what was being discussed, including possible target words. The availability of video would allow future work along the lines of Frank et al. (2009) and Jones et al. (2010) although so far, we haven't made direct use of the video recordings.

The digital audio/video recordings were downloaded onto a computer, and both adult and child speech were orthographically transcribed using CHAT conventions (cf. MacWhinney (2000)). The child data — but unfortunately not the caregivers' — were then also transcribed in phonemic transcription. All mother and child transcriptions, as well as audio/video files, can be found on the CHILDES database `http://childes.psy.cmu.edu/`. We used the XML version of the data for our transcription process.

2.1 Producing a phonemically transcribed version

To find CDS utterances, for all six children we extract the orthographic transcriptions for all utterances made by caregivers from the XML transcripts of the Providence corpus, starting from 11 months up to and including 22 months.[2] This makes our data qualitatively comparable to the BRB Corpus that includes CDS from between 13 and 21 months of the children's age. In total, we extract 101,451 utterances with a 9395 distinct (orthographic) types but the number of utterances we use is smaller than this because we do not keep all of the utterances in the process of transcription (see below).

To turn the orthographic representations into a phonemic format that is suitable for studying language acquisition, we perform a four step process of filtering, dictionary look-up, heuristic construction of pronunciations for unknown types and manual transcription of unknown types not covered by our heuristic as well as correction of mistakes made during earlier steps.

[2] Available at `http://childes.psy.cmu.edu/data-xml/Eng-USA/Providence.zip`

2.1.1 Filtering words

We manually remove types that we consider to be obvious non-words, in particular interjections such as *hmmmhmmm* or *mmmmhmmm*, obvious onomatopoeic wordplay such as *nananana* and unintelligible words which are transcribed in the Providence Corpus as *xxx* and *yyy*. This is consistent with the procedure followed by Brent (1999) and, more recently, Boruta and Jastrzebska (2012), making our corpus comparable in this respect to theirs. We do not, however, remove these items in cases where the resulting utterance would have been rendered fully unintelligible or where a word that should have been excluded according to the above criteria was used as an actual word in a large number of cases.[3]

In total, we identify 785 such non-words and we remove all occurrences of these types from the transcript, leaving the remaining words in the utterance:[4] utterances including any of the stop-words are still transcribed as long as there is at least one word left after removing all stop words. A total of 7,362 utterances are thus completely ignored, with 6,123 utterances consisting of exactly one of these filtered elements, in particular *xxx* (unintelligible, 2215), *oh* (525) and *hmmm* (521).

2.1.2 Dictionary lookup

After filtering, we perform a simple dictionary-lookup transcription using a phonemic dictionary. We use the current version of the VoxForge dictionary which uses a standard phone set for American English, corresponding to the current DARPABET coding.[5] We also provide a script that maps this representational scheme into one-character-per-phoneme representations that are required by some of the currently common word segmentation tools.[6]If there are multiple pronunciations available for a type, we always pick the first one. While this constitutes an idealization we believe that a well-understood and explicit idealization is to be preferred over an overly simplistic method of artificially introducing variability such as randomly choosing a pronunciation.

In total, the VoxForgeDictionary covers 7035 of the 8610 remaining types in the data, leaving 1575 of the types untranscribed. We transcribe these words manually, using a simple pre-processing heuristic to aid the process.

2.1.3 Heuristically constructing pronunciations for unknown words

Many of the unknown words are either forms of types that already are in the lexicon, e.g. possessives (*Elmo's*) or plurals (*Legos*), or compounds of two types that are both in the lexicon individually (*frenchtoast*, *teddybear*). We handle the former case by simple rules operating on the orthographic forms directly, looking for possible plural or possessive endings and then checking whether the orthographic form can be decomposed into a known base-form with the

[3]The former applies mostly to cases where an item is mentioned rather than used, e.g. "Does the baby say 'Wah wah'?"; the latter, for example, applies to 'bonk' which, in addition to its onomatopoeic use, also occurs as a verb in the corpus, including its preterite and participle. Our data includes the full list of filtered items as well as the scripts that perform the automatic steps of transcription from the original xml-data so that researchers can easily make their own decisions about which items to exclude.

[4]While this may seem like a lot of items to exclude, most of these are hapaxes like *bumpoopadoompadadooboom* or *doodleuhdoo*.

[5]The dictionary is available at `http://www.repository.voxforge1.org/downloads/SpeechCorpus/Trunk/Lexicon/VoxForge.tgz`

[6]E.g. dpseg (Goldwater et al. 2009).

Name	#Utt	#Tok	#Type	Avg. Utt. Len.	Avg. Tok. Len	Avg. Type Len.
Alex	8330	29423	1877	11.00	3.12	4.58
Violet	9024	39135	2343	13.43	3.10	4.68
William	10697	45689	2061	13.01	3.05	4.59
Ethan	18020	75564	2999	13.11	3.13	4.69
Lily	20641	94696	3946	14.77	3.22	4.96
Naima	27377	141990	4579	16.51	3.18	5.05

Table 1: Statistics about the different sub-corpora.

suffix added. In this case, we add the corresponding morph according to the rules of English phonology. To identify potential compounds, we try to decompose words into a prefix *p* and a maximal suffix *s* such that both *p* and *s* are known forms.

Taken together, our heuristic detects 924 cases which we then manually correct for mistakes. An alternative way of constructing pronunciations is to use letter-to-sound rules, something we plan to experiment with in future work.

2.1.4 Manual transcription

The remaining 651 word types are labelled manually, using where available the form-annotation in the XML files as guide-line.[7] Finally, we also manually inspect the forms generated by the automatic steps and correct mistakes.

2.2 Statistics

The final corpus comprises a total of 94,089 phonemically transcribed utterances and consists of six distinct sub-corpora, each corresponding to the CDS directed at one of the six children. Each sub-corpus is, in turn, subdivided into individual files corresponding to the age of the child at which recording took place, ranging from 11 up to 22 months. Both within these individual files and within the overall corpus, the order of the CDS utterances corresponds to the order in which these utterances were actually made, making them suitable for studies that look at changes over time.

Table 1 gives summary statistics over the full amount of data for each individual child. Looking at total number of utterances, we can broadly identify two groups: for Alex, Violet and William there are considerably less CDS utterances than for Ethan, Lily and Naima. This is presumably mostly due to the factors mentioned above. Yet, there also seem to be noticeable differences in terms of utterance-, token- and type-length. Although we will not do so in this paper, performing comparative evaluation of models across the children may lead to the discovery of interesting predictors of model performance and perhaps even actual language ability on behalf of the children.

For the rest of the paper, we will focus on the Naima part of the corpus and take a closer look at how the segmentation performance of different segmentation models changes as a function of the size of the input.

[7] While not always provided for caregiver utterances, some of them include phonetic markup for individual words, in particular if the words were names.

3 Bayesian Word Segmentation

The word segmentation models we study in this paper are Goldwater's Unigram and Bigram model (Goldwater, 2007; Goldwater et al., 2009) and Johnson (2008b)'s collocation-syllable Adaptor Grammar models. For the mathematical details of these models, we refer the reader to these original works.

All the models are Bayesian probabilistic models that define a generative process for the target of learning, in this case segmented utterances or sequences of words. This generative process is defined with the help of the Dirichlet Process (DP):[8] at an intuitive level, the DP can be used for word segmentation because it can bias models to identify compact ways to represent the observed unsegmented utterances, trading off the number of both tokens and types used in an analysis of the data. This trade-off is a consequence of the way that probabilities are assigned to tokens under a DP model: the probability of hypothesizing a word token[9] depends on the number of times that its type has previously been hypothesized, and the probability of a full hypothesis or segmentation of the data is the product of the probabilities for all the tokens used in the segmentation. This tends to make solutions in which a small number of words is used relatively frequently yet not over-excessively (as would be the case if every individual segment were a type) the most probable which, in most cases, also leads to linguistically reasonable results. What differentiates the different models from one-another are the specific assumptions about *the nature of possible word types* and *the relationships between word tokens*. We will quickly elaborate on these points, thus introducing all models used in our experiments.

3.1 Assumptions about possible words

A naive assumption about possible words is that they can be any arbitrary sequence of segments. Thus, both *dog* and *qfx* would be equally good candidates for possible words a priori. While obviously not true of human languages (*bnik* isn't a possible English word), such a unigram phoneme model is embodied in the original Unigram and Bigram models (Goldwater et al., 2009) and has been shown to work reasonably well on the BRB Corpus.

An alternative and linguistically more adequate assumption is that a word must conform to (arguably universal) constraints of syllable structure: words aren't just sequences of segments but consist of one or more syllables, themselves structured entities that consist of an optional initial consonantal onset, an obligatory vocalic nucleus and an optional consonantal coda (Smolensky and Legendre, 2005). In fact, Norris et al. (1997) provided evidence for a constraint on possible words along these lines in human segmentation of unsegmented speech and Johnson (2008b) showed previously how incorporating this kind of 'prior knowledge' into word-segmentation models can lead to a considerable improvement in segmentation accuracy.

From a language acquisition point of view, there are also reasons to assume that syllables and sub-syllabic units are learned by infants in addition to entire words: there is strong evidence that even very young infants track probabilities defined over entire syllables (Saffran et al., 1996) and that infants are sensitive to the phonotactics of their language from early on (Jusczyk et al., 1993), leading us to look at models that learn both entire syllables and sub-syllabic units. In addition, many languages including English are more or less restrictive about the material allowed to occur word-initially and word-finally than word-internally, having led previous work

[8] Adaptor Grammars actually use the Pitman-Yor Process, a strict generalization of the DP. Here, we gloss over this detail.

[9] Or a token of another entity, e.g. a syllable or a multi-word expression, if the model incorporates these notions.

such as Johnson (2008b) to distinguish between word-internal and word-peripheral onsets and codas.[10] The models we discuss, then, incorporate all of these ideas, that is they learn sub-syllabic units that are distinguished as to whether they occur word-peripherally or internally as well as entire syllables. The Adaptor Grammar framework makes it very easy to implement these different models, and we refer the reader to Johnson (2008b) for a discussion of how this is done.[11]

3.2 Assumptions about relations between word tokens

The simplest assumption about the relation between words within an utterance is probably that there is none. For a probabilistic model, this leads to a Unigram assumption about words, i.e. that the probability of a sequence of words is simply the product of the probability of each individual word, irrespective of its specific context. This is the assumption embodied by Goldwater's original Unigram model and a lot of previous work on word segmentation (Brent, 1999; Venkataraman, 2001).

A more plausible assumption is that words in a sequence are predictive of each other, a simple version of which can be spelt out as a Bigram assumption about words that is employed by Goldwater's Bigram model.

Yet another way of modelling the relation between words has been proposed by Johnson (2008b), employing a hierarchical instead of a sequential notion of context. The *collocation* models assume that sentences are sequences of *phrases* that, themselves, are made up of individual words. Consequently, the model not only learns words but an additional kind of entity, entire chunks of words. Importantly and in contrast to the Unigram and Bigram models, these chunks are stored in addition to and not at the expense of the words that make them up — a collocation model can infer that *thedoggie* is a high-frequent unit that is made up of the distinct words *the* and *doggie*. Johnson's Adaptor Grammar framework makes it easy to assume multiple levels of such collocations: a collocation2-grammar, for example, learns, in addition to words and collocations composed of words, collocations that are themselves composed of collocations of words. In this paper, following Johnson and Goldwater (2009) we examine models that use up to three levels of collocations.

3.3 The different models

As the different kinds of assumptions about the internal structure of words and about the relationship between words are independent, we can freely combine them, yielding a total of 9 models.[12] We refer to the different models by names that indicate which assumptions they embody. Models that do not incorporate the syllable structure assumption are simply referred to by the relationship assumed between word tokens, i.e. *unigram*, *bigram*, *colloc*, *colloc2* and *colloc3*, the last two names referring to models assuming a total of 2 and 3 levels of collocations, respectively.

Except for the bigram model, each of these models can either use a 'naive' word-assumption

[10] For example, in English *str* is only valid as an onset word-initially, as in *string* or *strong*, and the consonant cluster *dths* in *widths* is restricted to the end of words.

[11] In fact, the sole difference between Johnson's models and our own is that his do not learn entire syllables.

[12] With the exception of the bigram model for which we know of no existing implementation that makes it possible to incorporate syllable structure. In principle, however, this model is a possibility and we hope to be able to study it in the future.

allowing arbitrary sequences of segments or the syllable-structure word-assumption which, in addition to constraining the space of possible words, enables the model to explicitly learn properties of the language's phonology. To distinguish these two cases, we suffix models that embody the latter assumption with *Syll*.

4 Experiments

We are interested in how the segmentation performance of the different models varies with the amount of input available to the models during learning. As human language learners tend to get better at their native language with longer exposure, one would expect adequate computational models to exhibit something similar, initially improving as more data is observed and, at some point (probably beyond the size of samples we usually can look at in practice), asymptotically approaching some upper bound.

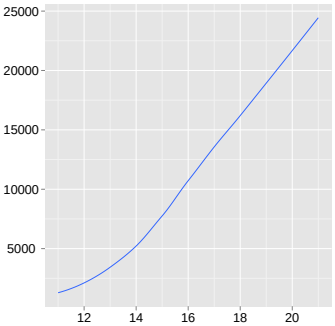


Figure 1: The months on the x-axis correspond to the different inputs, the y-axis gives the number of utterances in an input. The smallest input is month 11 with 973, the largest input month 21 with 24,327 utterances, approx 2.5 times the size of the BRB corpus.

The longitudinal data available in the Providence Corpus suggests a natural setup for studying this question by constructing inputs that consist of all CDS utterances directed at an individual child up to a certain point of time. For our experiments, we use the Naima section of the Providence Corpus and collect CDS utterances from when Naima was 11 months old through to when she was 21 months old to construct 11 differently sized inputs, each input consisting of all CDS utterances in the corpus up to and including a given month. We will refer to the different input sets by the last month from which it includes data and will use "language exposure" and "input size" exchangeably, a simplifying yet justified choice as is evident from figure 1 that shows how the size of the input grows over time and thus with language exposure.

We choose to evaluate on held-out data and construct the test-set by sampling 200 CDS utterances from the 22nd month of each of the six children's sub-corpus in the Providence Corpus. We propose this as a standard test corpus that can be used as a standard for any or all of the 6 children's data in the Providence Corpus. Our evaluation metric is token f-score, the harmonic mean between token precision (number of correct tokens identified by the model over the number of total tokens predicted by the model) and token recall (number of correct tokens identified by the model over the true number of tokens in the input) as a measure of segmentation accuracy. The segmentation on the held-out data is calculated after probabilistic inference has been performed on the input, thus implicitly defining a (probabilistic) lexicon according to which we sample a segmentation for each utterance in the test-set. Note that during this process, no novel words are added to the model's lexicon; in this sense, we evaluate the knowledge the learner has acquired after having had access to the input.

Our experimental procedure follows closely the one outlined in Johnson and Goldwater (2009). We use the current version of Mark Johnson's Adaptor Grammar implementation[13] to run

[13]Available at `http://web.science.mq.edu.au/~mjohnson/code/py-cfg-2012-08-16.tgz`.

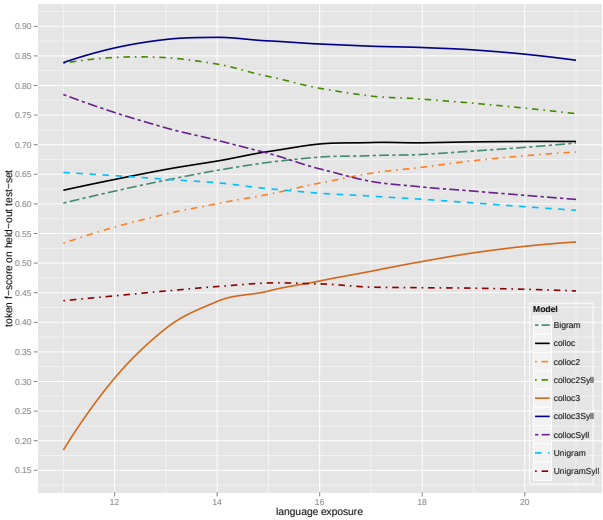


Figure 2: Word segmentation performance on the held-out test-set for the difference models, as a function of language exposure and consequently, size of the training set (see figure 1). The models incorporating syllable structure all exhibit some degree of performance decrease for larger inputs although the colloc3Syll consistently remains well above 80%. In contrast, the models without syllable structure exhibit an increase in performance over time although their segmentation accuracy is considerably worse than that of the best performing models with syllable-structures.

two Markov Chain Monte Carlo chains for each of the models for 1000 iterations, collecting sample analyses for the held-out test-set with a lag of 5 after a burn-in of 800 iterations and performing Minimum Bayes Risk decoding to get a single score for each condition at the end. For the Bigram model, the only model not expressible as an Adaptor Grammar, we use Sharon Goldwater's implementation and the experimental paradigm outlined in Goldwater (2007) and Goldwater et al. (2009), using simulated annealing for 20,000 iterations and evaluating on a single sample taken at the end. Figure 2 plots segmentation accuracy as measured by token f-score for different amounts of input size, with the size of the input growing from left to right.

Overall, we can see two broad patterns of behaviour across the different models. One group of models exhibits a degradation in performance for larger amounts of inputs, in particular the Unigram, the collocSyll and the colloc2Syll model. Neither the UnigramSyll nor the colloc3Syll model show a lot of variation although a small drop in performance for the last month as compared to their peak performance is noticeable for them as well. Another group consisting of the colloc, colloc2, colloc3 and Bigram models exhibits an increase in performance with amount of training data.

The colloc3Syll model clearly emerges as the most accurate with an accuracy of 87% at peak performance at around 14 months (4794 utterances) that drops to around 83% at month 21 (24,327 utterances). The second best model, the colloc2Syll model, peaks at around 85% at

month 13 and drops by 10% to about 75% at month 21, showing a clear decrease in accuracy. A third place is harder to assign. The collocSyll model shows the most dramatic drop in performance, from around 75% at month 11 (973 utterances) to just above 60% for month 21. In contrast, both the colloc and the Bigram model start around 60% at month 11 but, for month 21, reach around 70% so that there is no single model that comes in third for all amounts of input size. Incidentally, note how the behaviour of the Bigram and the colloc model is very similar, supporting Johnson (2008b)'s hypothesis that "the collocation word adaptor grammar can capture inter-word dependencies similar to those that improve the performance of Goldwater's bigram segmentation model."[14] The most dramatic performance improvement is seen for the colloc3 model which jumps from around 18% for month 11 to about 53% for month 21. Despite this, it is the second-worst model, loosing only to the UnigramSyll model which stays consistently below 50% accuracy with little variation.

To sum up, we observe two types of behaviour. The Unigram model and the models incorporating syllable structure and the notion of a multi-word phrase exhibit what we call "overlearning": they reach their peak performance for relatively small amounts of input and gradually get worse as the size of the input grows larger. This is much more pronounced for the collocSyll and the colloc2Syll than for the colloc3Syll model, with the latter remaining well above 80% accuracy even for the largest amount of input. Despite overlearning, the colloc3Syll and the colloc2Syll perform word segmentation the most accurate for all sizes of input. On the other hand, the models lacking the assumption of syllable structure do not exhibit overlearning, at least not on the amount of data we were able to test them.

5 Discussion

Why do the models we examine exhibit these two kinds of behaviour? We begin with a detailed explanation of the "overlearning", starting from an original observation going back to Goldwater (2007) who noticed that the Unigram model tends to identify undersegmented solutions where the predicted (incorrect) words often consist of several of the (correct) words. Her explanation for this is that "groups of words that frequently co-occur violate the unigram assumption in the model since they exhibit strong word-to-word dependencies", and that "[t]he only way the model can capture these dependencies is by assuming that these collocations are in fact words themselves." (Goldwater, 2007, p.72) Why is it, however, that these "misleading" co-occurrences occur in the data in the first place, and can this explanation be extended to account for the input-size effect we detected?

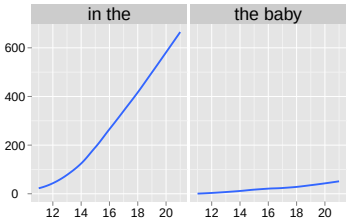


Figure 3: Frequency of a preposition+determiner pattern and determiner+noun pattern on the y-axis as a function of input size on the x-axis. Note the steep increase of frequency of *in the* and the much less dramatic increase for *the baby*.

We suspect that many of the "collocations" a model such as the Unigram model is susceptible to arise from principled regularities governing language: English syntax, for example, requires (most) prepositional phrases to begin with a preposition-determiner sequence and both preposi-

[14] Unlike Johnson (2008b) and Goldwater (2007), we did not hand-tune any of the parameters of the models which may partly explain why our scores for the Bigram model are slightly lower for the earlier months, in addition to the fact that we didn't use Minimum Bayes Risk decoding for the Bigram model.

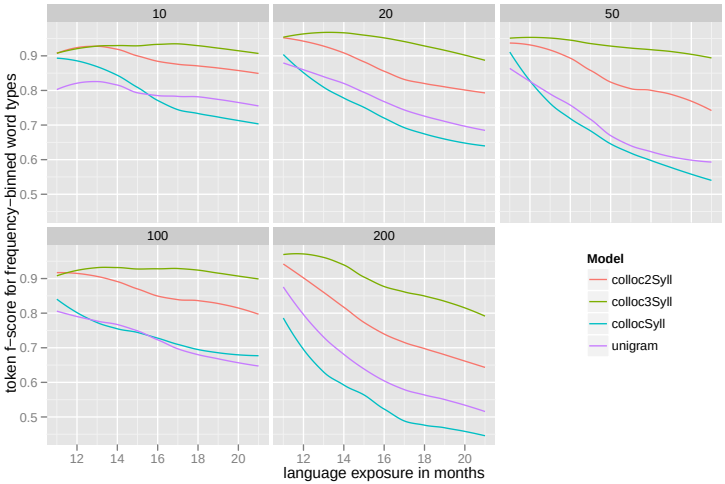


Figure 4: Token f-scores for word types of different frequencies in the test-set as a function of the size of the input. Note that the Unigram and the collocSyll model already show clear decreases in accuracy over time for types of frequency larger than 10 whereas the colloc3Syll model shows a relatively robust performance up to the highest-frequency bin of types with frequency larger than 200. The presence of the drop in performance shows that even the colloc3Syll model suffers from "overlearning" although this behaviour is much less pronounced than for the other models and only occurs dramatically for the highest-frequency types in the data.

tions and determiners are small closed classes, leading to a huge number of sequences such as *in the* in virtually any English text. Crucially, the occurrence of a sequence such as *in the* is largely independent of what is actually being talked about and the number of such occurrences can therefore be expected to grow with the amount of data considered. This is supported by figure 3 which plots the change in frequency of the preposition-determiner sequence *in the*, showing that the number of "misleading" co-occurrences of that kind does indeed increase for larger amounts of data. This leads to the prediction that the Unigram model will perform worse when trained on larger amounts of data, simply because the evidence for these kinds of spurious words that lead to undersegmentation errors grows with the input size. To our knowledge, we are the first to formulate this hypothesis and our experimental results strongly suggest that it is true. The drop in performance for the Unigram model is clear from figure 1: it reaches peak performance of around 65% when its input consists of a mere 973 utterances, and its segmentation accuracy steadily drops as it processes larger inputs down to around 58% for an input of 24,327 utterances.

More direct support for explaining the drop by the negative impact of the increasing frequency of patterns like the one in figure 3 comes from figure 4 that plots how well the model is able to identify word types of different frequencies in the test set as a function of input size. Note how for higher-frequency types, the Unigram model's performance decreases more dramatically than for low frequency types for larger amounts of input. To investigate whether this difference in

pattern	#test	month 11 (973 utterances)					month 21 (24,327 utterances)				
		#input	Unigram		colloc3Syll		#input	Unigram		colloc3Syll	
			% cor	% col	% cor	% col		% cor	% col	% cor	% col
in the	18	14	50%	50%	94%	0%	671	0%	89%	0%	100%
are you	19	14	100%	0%	100%	0%	536	0%	71%	0%	100%
on the	21	11	100%	0%	81%	0%	347	0%	86%	14%	86%
this is	9	5	100%	0%	100%	0%	196	0%	67%	56%	44%
with the	4	2	100%	0%	100%	0%	153	0%	75%	75%	0%
the baby	4	0	100%	0%	100%	0%	80	0%	100%	100%	0%

Table 2: A qualitative look at how bigram-patterns of different frequency are handled by the Unigram and the colloc3Syll model at month 11 and month 21, respectively. The %cor and %col columns give the percentage of occurrences of a pattern in the test-set that were handled correctly or were misanalysed as constituting a two-word collocation. Note that %cor and %col need not add up to 100% as the models can also make other errors than simply undersegmeting. It is clear that at month 11, both the Unigram and the colloc3Syll model handle all the cases nearly perfectly; but unlike the Unigram model, the colloc3Syll model handles lower-frequency patterns at month 21 with increasing ease, making no mistake for a unit that occurs 80 times in its input such as *the baby* but still mis-analysing a pattern such as *in the*.

performance could actually be due to high-frequency items getting "absorbed" into larger units as we suggest following Goldwater (2007), in table 2 we perform a qualitative evaluation of a number of actual examples of patterns involving high-frequency that are themselves of different frequencies. As is clear, cases that are analysed correctly at month 11 are almost consistently misanalysed as a single word at month 21, showing that the "loss" of high-frequency items is a major reason for the overlearning.

Surprisingly, perhaps, the same kind of explanation seems to apply to the collocation-syllable models. While originally proposed by Johnson (2008b) to specifically address the problem of undersegmentation, looking at figure 4 indicates that collocation models do not solve the problem of high-frequency words completely, although it seems to get less problematic with the number of collocational levels the model has at its disposal. Looking at the kind of units learned by the collocation-models helps understand why that is: among the high-frequency collocations learned by both the collocSyll and the colloc2Syll model from the largest input is, for example, the two 'word' sequence *doyou remember*; this is a better solution than the Unigram model's *doyouremember* but it still involves the undersegmented collocation *do you*. While the colloc3Syll model analyses this specific case correctly as a three-word collocation *do you remember*, it fails to acquire the collocation *do you* on its own and prefers to use the "word" *doyou* in most other cases, showing that collocations do not solve the undersegmentation problem but only push it back a level, in line with figure 4: note that while the colloc3Syll model behaves relatively stable for word-types with frequencies smaller than 200, it also shows a marked drop for the high-frequency word types from around 95% at month 11 to just below 80% for month 21. Some concrete examples of patterns which the colloc3Syll model is and isn't able to handle correctly are given in table 2, alongside the performance of the Unigram model for these cases, showing how its ability to handle word-dependencies breaks for high-frequency patterns but handles patterns the Unigram model is unable to analyse correctly. After this discussion, the fact that even for collocation-models the undersegmentation problem gets worse for larger inputs shouldn't be too surprising. As pointed out above, many of the patterns leading to undersegmentation errors are due to syntactic regularities that, for example, require prepositions to be followed (in almost all cases) by articles. Figure 3 indicates that these kinds

of patterns grow continuously with the size of the input, suggesting that models that "merely" model co-occurrence statistics are bound to fail at some point. This may almost seem like a general problem for Bayesian probabilistic models of the kind discussed here that, in a sense, simply try to identify high-frequency patterns in the input. Yet this is not so. For one thing, the lack of detailed linguistic structure is not inherent to the Bayesian framework that is fully unrestricted as to what kind of structures a model is defined over. This much is clear from the ease with which syllable structure can be incorporated into the models. Secondly, even without additional linguistic structure the relative robustness of the colloc3Syll model shows that while not fully solving the problem of misleading co-occurrences, a sufficiently rich collocational structure goes a long way in alleviating the problem for input sizes that go well beyond 20,000 utterances. It suffices to handle most cases involving content words such as nouns, correctly learning for example that despite its (relatively) high frequency, *the baby* consists of two individual words. Figure 3 shows that for patterns like this, frequency grows much slower over time (although still too fast for a model lacking any ability to model larger-than-word-units such as the Unigram model), not too surprising considering that the occurrence of content words — unlike function words — is mainly dependent on what is actually being talked about and that conversation topics tend to change over time. This then suggests that a model like colloc3Syll will handle correctly these important cases for even considerably larger amounts of data, a clear desideratum for models of early language acquisition phenomena.

This leaves us to explain why the collocation models without syllable structure lead to an overall worse performance but seem to exhibit a positive relation between amount of input and segmentation accuracy. The key to this, we believe, lies again in considering the kinds of regularities the model is sensitive to. With no restriction on what an actual word may look like, high-frequency patterns of any kind — including individual segments and short n-gram like sequences of segments — can be employed by the models to explain the input they get. In particular for little input with overall few word tokens and, consequently, relatively few repetitions for each of the actual word types, the evidence for high-frequency non-words is extremely high, leading to over- rather than undersegmentation. To understand why this isn't happening for the Unigram model, recall that under the Unigram model, all tokens in a hypothesis are fully independent whereas the collocation and the Bigram model assume dependencies between tokens. Consequently, under the Unigram model a solution involving a large number of items is automatically discouraged because it can only work with marginal probabilities, whereas the other models without syllable structure can "abuse" their respective notions of context to capture dependencies not only between words but also between sub-word units. Ironically then, the oversegmentation behaviour is worst for models with a lot of additional structure such as the colloc3-model that, when combined with a syllable structure constraint, leads to the best performance. A quick glance at the kinds of units identified by it suggests why that is — at month 11, the most frequent "word" that is learned is *t* which is used in "collocations" like *t o*, *ge t* or, illustrating the problem very nicely, *j us t*. Crucially, the more input it has access to the stronger the evidence for larger (more word-like) units gets, leading to less undersegmentation at month 21 with a top-5 word list of *ing*, *you*, *z*, *the* and *to*. Yet, even for well over 20,000 utterances the colloc3 model prefers extremely short 'words' as it has another three levels of collocations that it can use to capture words; and it actually does learn collocations such as *ma mi*. The colloc2, colloc and the Bigram model are less extreme in their oversegmentation behaviour as they have fewer "levels" at their disposal, discouraging the excessive use of one-segment 'words' as in the colloc3 model more severely. In fact, as is suggested from both figure 2 and from inspecting their highest-frequency words at

month 21 that include "overlearned" units like *areyou* and *doyou*, the Bigram and the colloc model are starting to overlearn just like the syllable-structure models, as we would expect from basically every model ignorant of the linguistic regularities that can explain high-frequency patterns for sufficiently large amounts of data. In conclusion, we think this shows that the intuitively attractive behaviour of these models is an artefact of their strong preference for short units that, for small amounts of data, masks the overlearning at the expense of segmentation accuracy; in particular, the fact that models lacking syllable structure do not exhibit overlearning in our experiments should not be taken as evidence that they are more adequate than their overlearning relatives.

It is possible that the overlearning behavior we describe in this paper can be addressed by manually choosing appropriate values for the models' hyper-parameters, which control the models' Pitman-Yor Processes (PYPs). Goldwater (2007) observed that the segmentations proposed by the unigram and bigram models depend on the choice of hyper-parameters, and Johnson (2008b) observed a similar sensitivity to hyper-parameters in Adaptor Grammar models. However, the number of hyper-parameters is twice the number of PYPs in a model, and searching for hyper-parameter values that result in the most accurate segmentation is computationally very challenging. For this reason Johnson and Goldwater (2009) placed Beta and Gamma priors on the PYP a and b hyper-parameters respectively in their Adaptor Grammar models, and reported that sampling the hyper-parameters actually improved segmentation f-score as compared to the manually-specified settings they investigated. It is reasonable to expect that more realistic models of human language will be more complex, and therefore will have even more hyper-parameters, than the models investigated here. It is certainly conceivable that the hyper-parameters are innately fixed by universal grammar to values that result in good segmentations across different languages. But all else being equal, models which do not require hyper-parameters to be fixed to specific values should be preferred on general simplicity grounds over models that do require such prespecification. For this reason we chose to use Johnson and Goldwater's hyper-parameter sampler in the Adaptor Grammar models we studied here.

Conclusion and perspectives

We have presented a novel corpus of English CDS derived from the Providence Corpus for studying models of word segmentation. This corpus makes it possible to address a wider range of questions than is currently common, for example with respect to the study of developmental effects in incremental algorithms. We identified an interesting "overlearning" effect for state-of-the-art word segmentation models on large amounts of data that has so far gone unnoticed and proposed an explanation of this behaviour that highlights the importance of linguistic structure for Bayesian models; we have argued that the apparent lack of overlearning for linguistically less-sophisticated models is due to an undesirable preference for oversegmentation and shouldn't be mistaken for an advantage.

In future work, we want to further explore the impact of linguistic knowledge for Bayesian models of the kind discussed here; we suspect that giving the model the ability to model more of the regularities languages exhibit more appropriately, the overlearning behaviour we observed will become less severe. Also, we want to test the cross-linguistic usefulness of different kinds of constraints such as assuming (a certain kind of) syllable-structure. This is complicated by the fact that most current data-sets do not separate CDS directed at different children and children of different ages. We hope that our presentation of a data-set that fulfills these desiderata will lead to the creation of corpora like this for other languages as well.

References

Börschinger, B. and Johnson, M. (2012). Using rejuvenation to improve particle filtering for bayesian word segmentation. In *The Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics*, Jeju, South Korea. Association for Computational Linguistics.

Boruta, L. and Jastrzebska, J. (2012). A phonemic corpus of polish child-directed speech. In *Proceedings of the Eight International Conference on Language Resources and Evaluation (LREC'12)*, pages 1017–1020, Istanbul, Turkey. European Language Resources Association (ELRA).

Brent, M. (1999). An efficient, probabilistically sound algorithm for segmentation and word discovery. *Machine Learning*, 34:71–105.

Brent, M. and Cartwright, T. (1996). Distributional regularity and phonotactic constraints are useful for segmentation. *Cognition*, 61:93–125.

Demuth, K., Culbertson, J., and Alter, J. (2006). Word-minimality, epenthesis, and coda licensing in the acquisition of english. *Language and Speech*, 49:137–174.

Frank, M. C., Goodman, N., and Tenenbaum, J. (2009). Using speakers' referential intentions to model early cross-situational word learning. *Psychological Science*, 20:579–585.

Gervain, J. and Erra, R. G. (2012). The statistical signature of morphosyntax: A study of hungarian and italian infant-directed speech. *Cognition*, (0):–.

Goldwater, S. (2007). *Nonparametric Bayesian Models of Lexical Acquisition*. PhD thesis, Brown University.

Goldwater, S., Griffiths, T. L., and Johnson, M. (2009). A Bayesian framework for word segmentation: Exploring the effects of context. *Cognition*, 112(1):21–54.

Johnson, M. (2008a). Unsupervised word segmentation for Sesotho using adaptor grammars. In *Proceedings of the Tenth Meeting of ACL Special Interest Group on Computational Morphology and Phonology*, pages 20–27, Columbus, Ohio. Association for Computational Linguistics.

Johnson, M. (2008b). Using adaptor grammars to identifying synergies in the unsupervised acquisition of linguistic structure. In *Proceedings of the 46th Annual Meeting of the Association of Computational Linguistics*, pages 398–406, Columbus, Ohio. Association for Computational Linguistics.

Johnson, M. and Demuth, K. (2010). Unsupervised phonemic Chinese word segmentation using adaptor grammars. In *Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010)*, pages 528–536, Beijing, China. Coling 2010 Organizing Committee.

Johnson, M. and Goldwater, S. (2009). Improving nonparameteric Bayesian inference: experiments on unsupervised word segmentation with adaptor grammars. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 317–325, Boulder, Colorado. Association for Computational Linguistics.

Jones, B. K., Johnson, M., and Frank, M. C. (2010). Learning words and their meanings from unsegmented child-directed speech. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 501–509, Los Angeles, California. Association for Computational Linguistics.

Jusczyk, P., Friederici, A., Wessels, J., Svenkerud, V., and marie Jusczyk, A. (1993). Infants' sensitivity to the sound patterns of native language words. *Jounral of Memory and Language*, 32:402–420.

MacWhinney, B. (2000). *The CHILDES Project: Tools for analyzing talk, 3rd Edition. Vol 2: The Database*. Lawrence Erlbaum Associates.

Norris, D., McQueen, J. M., and Cutler, A. (1997). The possible-word constraint in the segmentation of continuous speech. *Cognitive Psychology*, 34:191–243.

Pearl, L., Goldwater, S., and Steyvers, M. (2011). Online learning mechanisms for Bayesian models of word segmentation. *Research on Language and Computation*, 8(2):107–132.

Phillips, L. and Pearl, L. (2012). "Less is more" in Bayesian word segmentation: when cognitively plausible learners outperform the ideal. In Miyake, N., Peebles, D., and Cooper, R. P., editors, *Proceedings of the 34th Annual Conference of the Cognitive Science Society*, Austin, Texas. Cognitive Science Society.

Saffran, J., Aslin, R., and Newport, E. (1996). Statistical learning by 8-month-old infants. *Science*, 274:1926–1928.

Smolensky, P. and Legendre, G. (2005). *The Harmonic Mind: From Neural Computation To Optimality-Theoretic Grammar*. The MIT Press.

Venkataraman, A. (2001). A statistical model for word discovery in transcribed speech. *Computational Linguistics*, 27(3):351–372.

Bayesian Language Modelling of German Compounds

Jan A. Botha[1] *Chris Dyer*[2] *Phil Blunsom*[1]
(1) University of Oxford
(2) Carnegie Mellon University
{jan.botha,phil.blunsom}@cs.ox.ac.uk, cdyer@cs.cmu.edu

ABSTRACT
In this work we address the challenge of augmenting *n*-gram language models according to prior linguistic intuitions. We argue that the family of hierarchical Pitman-Yor language models is an attractive vehicle through which to address the problem, and demonstrate the approach by proposing a model for German compounds. In our empirical evaluation the model outperforms a modified Kneser-Ney *n*-gram model in test set perplexity. When used as part of a translation system, the proposed language model matches the baseline BLEU score for English→German while improving the precision with which compounds are output. We find that an approximate inference technique inspired by the Bayesian interpretation of Kneser-Ney smoothing (Teh, 2006) offers a way to drastically reduce model training time with negligible impact on translation quality.

TITLE AND ABSTRACT IN AFRIKAANS

Bayes-modellering van saamgestelde woorde in Duits

Hierdie werk neem uitdagings rondom die uitbreiding van *n*-gramtaalmodelle volgens voorafgaande linguistieke intuïsie onder die loep. Ons voer aan dat die familie van hiërargiese Pitman-Yor taalmodelle 'n wenslike stuk gereedskap is om hierdie probleem mee aan te pak en formuleer 'n model van Duitse saamgestelde woorde om die benadering te demonstreer. Met behulp van 'n empiriese evaluering bevind ons dat die model in terme van toetsdataperpleksiteit beter vaar as die aangepaste Kneser-Ney *n*-grammodel. As onderdeel van 'n Engels→Duits-vertalingstelsel behaal die model in terme van die BLEU-metriek dieselfde vertaalafvoerkwaliteit as die kontrole stelsel en genereer saamgestelde woorde teen 'n hoër presisie. Verder stel ons vas dat 'n benaderde inferensietegniek, geïnspireer deur die Bayes-interpretasie van Kneser-Ney-gladstryking (Teh, 2006), gebruik kan word om die modelberamingtyd drasties te verminder sonder wesenlike impak op die vertaalafvoerkwaliteit.

KEYWORDS: language model; Bayesian methods; machine translation; compounding; ngram model; approximate inference.

KEYWORDS IN L_2: taalmodel; Bayes-metodes; masjienvertaling; samestellings; ngrammodel; benaderde inferensie.

Proceedings of COLING 2012: Technical Papers, pages 341–356,
COLING 2012, Mumbai, December 2012.

1 Introduction

Statistical language modelling addresses the problem of assigning probabilities to sentences in natural languages. In an effective model, these probabilities function as statistical proxies for sentences' syntactic well-formedness and semantic plausibility. As such, language models (LMs) play a crucial role in machine translation (MT) and automatic speech recognition (ASR) systems, which need to distinguish well-formed output sentences from ill-formed ones.

To tackle the problem of assigning reasonable probabilities to an infinite space of possible sentences, two assumptions are commonly made: first, the closed vocabulary assumption, which states that sentences are sequences of words from a *finite* vocabulary $\mathscr{W}$, and second, a Markov assumption is made, which states that the probability of each word in a sentence is conditionally independent of all others, given the previous $n-1$ words of context. Relying on these assumptions, language modelling becomes the problem of estimating the conditional probabilities of $|\mathscr{W}|$ words in $|\mathscr{W}|^{n-1}$ contexts.

There are two problems with this approach that we address in this paper.[1] First, the closed vocabulary assumption is often unreasonable, in particular for languages that use productive compounding to create novel word types. We therefore focus on modelling German since it makes extensive use of productive compounding and gives us an opportunity to explore this problem in depth. Second, in a naïve n-gram parametrisation, words are modelled independently of each other. This is problematic since the number of parameters is far too large to estimate reliably from even the largest corpora, and it ignores our intuition that related word forms have related behaviour. We solve these problems with an n-gram language model based on the hierarchical Pitman-Yor process (HPYP): Our model relaxes the closed vocabulary assumption by incorporating productive compound formation in its generative story, while the hierarchical structure enables us to relax the naïve independence assumptions about the statistical behavior of related word forms.

In the next section, we address the German compound problem in further detail and use this to motivate the structure of our model (§3). We then discuss the inference problem (§4), and evaluate the model's performance in terms of held-out perplexity and on translation quality when used inside an English→German translation system (§5). We conclude by placing this work in the context of related approaches (§6) and addressing avenues for future work.

2 Compound Words

Our aim in this work is to develop a language model that accounts for the structure of compound words. Compounding is a process whereby words are formed by combining other words. In some languages (including German, Swedish, Dutch and Afrikaans), compounds are written as single orthographic units. NLP systems that rely on whitespace to demarcate their elementary modelling units, e.g. the "grams" in n-gram models, are thus prone to suffer from sparse data effects that can be attributed to compounds specifically. An account of compounds in terms of their components therefore holds the potential of improving the performance of such systems.

Examples of compounds

- A basic noun-noun compound:
  Auto + Unfall = Autounfall *(car crash)*

[1] Preliminary work on the approach we follow in this paper was previously reported on by Botha (2012). Here, we expand on the scale and depth of the empirical evaluation and investigate an additional inference technique.

- *Linking elements* can appear between components
  Küche + Tisch = Küchentisch *(kitchen table)*
- Components can undergo stemming
  Schule + Hof = Schulhof *(schoolyard)*
- Compounding is recursive
  (Geburt + Tag) + Kind = Geburtstag + Kind = Geburtstagskind *(birthday boy/girl)*
- Compounding extends beyond noun components
  Zwei-Euro-Münze *(two Euro coin)* Fahrzeug *(vehicle)*

A compound is said to consist of a *head* component and one or more *modifier components*, with optional *linking elements* between consecutive components (Goldsmith and Reutter, 1998). The linguistic intuition that we propose to exploit in our language model is that German compounds are overwhelmingly right-headed (Toman, 1992), i.e. the right-most component fully determines the word's morphosyntactic properties. For example, the "Bahn" in "Eisenbahn" (railway) identifies the word as singular feminine, which determines the requirements for its agreement with verbs, articles and adjectives.

A language model could therefore give a reasonable assessment of the syntactic fluency of a sequence of German words by ignoring the non-head components of compounds. For example, the sentence, "I'm going by train" can be rendered in German as either of the following:

- Ich fahre mit der Eisenbahn.
- Ich fahre mit der Bahn.

Collapsing all compounds to their heads and ignoring modifiers would decrease sparsity and allow more robust n-gram probabilities to be estimated from data. But such a strategy would not be probabilistically sound as a generative model of a corpus. Moreover, a model that ignores modifiers would assign the same probability value to "Eisenbahn" and the empirically much rarer "Bobbahn" (bobsled), which would be unsatisfactory in a task where the language model plays a disciminative role. The model needs to account for the non-head components in some way. We expect the identity and number of modifier components to be strongly correlated with the identity of the head. In particular, the conditional distributions of modifier given head will be sharply peaked. A simple approximation is thus to assume that, conditioned on the head, modifiers are generated by a reverse n-gram model:[2]

$$p(\text{eisenbahn} \mid \text{mit der}) \equiv p(\text{bahn} \mid \text{mit der}) \times p(\text{eisen} \mid \text{bahn}) \times p(\$ \mid \text{eisen})$$

The sentinel \$ indicates the word boundary and doubles as a control on the number of modifiers. In general, we will use this process as a *back-off* strategy, i.e., when the trigram "mit der Eisenbahn" is unobserved. Note that this is markedly different from linguistically naïve back-off models that would score the unobserved trigram "mit der Eisenbahn" by falling back on bigram or unigram estimates. In our model, we instead permit the model to back off to this decomposition before dropping valuable context information.

3 An n-gram Model with Compounding

In this section we aim to marry an n-gram model with the intuition of compound formation that we proposed before. We present an extension of the hierarchical Pitman-Yor language model

[2]The majority of compounds have two components and thus match this assumption well enough. Multipart compounds where the modifiers themselves are compounds may violate it.

Figure 1: Intuition for the proposed generative process of a compound word: The context generates the head component, which generates a modifier component, which in turn generates another modifier. (Literally, "with the cable car"; idiomatically, "by cable car")

(HPYLM) (Teh, 2006) that fulfils this aim. The particular properties of the Pitman-Yor process (PYP) (Pitman and Yor, 1997) that we exploit are its flexibility to specify arbitrary back-off distributions (making it easy to incorporate an additional model) and the fact that it generates distributions that adapt well to power-law behaviour, as is often observed in language.

We employ this HPYLM framework with its accompanying inference machinery rather than a seemingly obvious alternative of using two distinct word-level and compound-level n-gram models. The reasons are that our unified model can learn a subtle interpolation between those levels, obviating the need to introduce and tune an extraneous interpolation scheme between sub-models, while opening the door for future extensions, e.g. analysing compounds occuring in the n-gram history.

3.1 Hierarchical Pitman-Yor Language Model (HPYLM)

An n-gram model is an $(n-1)$-th order Markov model that approximates the joint probability of a sequence of words $\mathbf{w}$ as

$$p(\mathbf{w}) \approx \prod_{i=1}^{|\mathbf{w}|} p(w_i \mid w_{i-n+1}, \ldots, w_{i-1}), \tag{1}$$

where we occasionally abbreviate a context $[w_i, \ldots, w_j]$ as $\mathbf{u}$. In the HPYLM, the conditional distributions $p(w|\mathbf{u})$ are smoothed by placing PYP priors over them. The PYP is defined through its base distribution, and a *strength* (θ) and *discount* (d) hyperparameter that control its deviation away from its mean (which equals the base distribution).

The generative process for a word w in context $\mathbf{u}$ is:

$$\begin{aligned}
G_0 &= \text{Uniform}(|\mathscr{W}|) \\
G_\emptyset &\sim PY(d_0, \theta_0, G_0) \\
&\vdots \\
G_{\pi(\mathbf{u})} &\sim PY(d_{|\mathbf{u}|-1}, \theta_{|\mathbf{u}|-1}, G_{\pi\circ\pi(\mathbf{u})}) \\
G_{\mathbf{u}} &\sim PY(d_{|\mathbf{u}|}, \theta_{|\mathbf{u}|}, G_{\pi(\mathbf{u})}) \\
w &\sim G_{\mathbf{u}},
\end{aligned}$$

where $\pi(\mathbf{u})$ truncates the context $\mathbf{u}$ by dropping the left-most word in it. The hyperparameters are tied across all priors with the same context length $|\mathbf{u}|$. To explain this process in terms of the familiar trigram case, consider how the probability $p(w \mid u, v)$ comes to be. Let $\mathbf{u} = [u, v]$. $G_{[u,v]}$ is then the PYP-distributed distribution over w. The hierarchy arises by using as the base distribution for the prior of $G_{[u,v]}$ another PYP-distributed $G_{[v]}$, i.e. the distribution $p(w \mid v)$. The recursion bottoms out at the unigram distribution $G_\emptyset$, which is drawn from a PYP with base distribution equal to the uniform distribution over the vocabulary $\mathscr{W}$.

3.2 Hierarchical Pitman-Yor Language Model + Compounds (HPYLM+c)

We define a compound word $\tilde{w}$ as a sequence of components $[c_1, \ldots, c_z]$, plus a sentinel symbol \$ marking either the left or the right boundary of the word, depending on the direction of the model. To maintain generality over this choice of direction, let Λ be an index set over the positions, such that c_{Λ_1} always designates the head component.

Following the motivation in §2, we set up the model to generate the head component c_{Λ_1} conditioned on the word context $\mathbf{u}$, while the remaining components $\tilde{w} \setminus c_{\Lambda_1}$ are generated by some model F, independently of $\mathbf{u}$.

To encode this, we modify the HPYLM thus:

1. Replace the support with the reduced vocabulary $\mathscr{M}$, the set of unique elementary components c obtained when segmenting the items in $\mathscr{W}$. ($\mathscr{M}$ also includes items consisting of a single component to begin with.)
2. Add an additional level of conditional distributions $H_{\mathbf{u}}$ (with $|\mathbf{u}| = n-1$) where items from $\mathscr{M}$ combine to form the observed surface words.

The generative process changes as follows (see also Figure 2):

$$
\begin{aligned}
G_0 &= \text{Uniform}(|\mathscr{M}|) \\
G_\emptyset \ \ldots \ G_{\mathbf{u}} &\quad \text{(as before)} \\
H_{\mathbf{u}} &\sim PY(d_{|\mathbf{u}|}, \theta_{|\mathbf{u}|}, G_{\mathbf{u}} \times F) \\
\tilde{w} &\sim H_{\mathbf{u}}
\end{aligned}
$$

So the base distribution for the prior of the word n-gram distribution $H_{\mathbf{u}}$ is the product of a distribution $G_{\mathbf{u}}$ over compound heads, given the same context $\mathbf{u}$, and another (n'-gram) language model F over compound modifiers, conditioned on the head component.

Choosing F to be a bigram model ($n' = 2$) yields the following procedure for generating a word:

$$
\begin{aligned}
&c_{\Lambda_1} \sim G_{\mathbf{u}} \\
&\text{for } i = 2 \text{ to } z \\
&\qquad c_{\Lambda_i} \sim F(\cdot | c_{\Lambda_{i-1}})
\end{aligned}
$$

The linguistically motivated choice for conditioning in F is $\Lambda^{\text{ling}} = [z, z-1, \ldots, 1]$ such that c_{Λ_1} is the true head component; \$ is drawn from $F(\cdot|c_1)$ and marks the left word boundary.

In order to see if the correct linguistic intuition has any bearing on the model's extrinsic performance, we will also consider the reverse, supposing that the left-most component were actually more important in this task, and letting the remaining components be generated left-to-right. This is expressed by $\Lambda^{\text{inv}} = [1, \ldots, z]$, where \$ this time marks the right word boundary and is drawn from $F(\cdot|c_z)$.

Linking Elements In the preceding definition of compound segmentation, the linking elements do not constitute items in the vocabulary $\mathscr{M}$. Regarding linking elements as components in their own right would sacrifice important contextual information and disrupt the conditionals $F(\cdot|c_{\Lambda_{i-1}})$. That is, faced with the compound Küche·n·tisch, we want P(küche|tisch) in the model, but not P(küche|n).

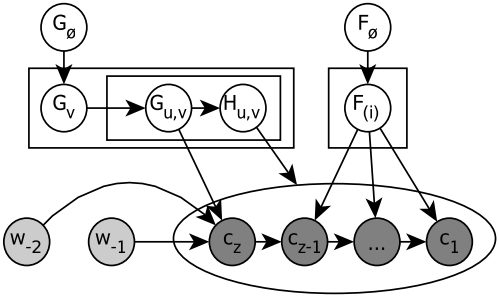


Figure 2: Plate diagram showing how a trigram version of HPYLM+c, using a bigram model F with condition scheme Λ^{ling} for modifiers, generates a word (the ellipse), consisting of head c_z and modifiers $c_1 \ldots c_{z-1}$. Here, w_{-2} and w_{-1} form the trigram context. We omit hyperparameters and their priors for clarity.

But linking elements must be accounted for to have a well-defined generative model. We follow the pragmatic option[3] of merging any linking elements onto the adjacent component – for Λ^{ling} merging happens onto the preceding component (e.g. P(küchen|tisch)), while for Λ^{inv} it is onto the succeeding one (e.g.P(ntisch|küche)). This keeps the 'head' component c_{Λ_1} intact.

4 Training

For ease of exposition we describe inference with reference to the trigram HPYLM+c model with a bigram HPYLM for F, but the general case should be clear.

The model is specified by the latent variables $\mathscr{L} = (G_{[\emptyset]}, G_{[v]}, G_{[u,v]}, H_{[u,v]}, F_{\emptyset}, F_c)$, where $u, v \in \mathscr{W}$, $c \in \mathscr{M}$, and hyperparameters $\Omega = (d_i, \theta_i, d'_j, \theta'_j, d''_2, \theta''_2)$, where $i = 0, 1, 2$, $j = 0, 1$, single primes designate the hyperparameters in F_{HPYLM} and double primes those of $H_{[u,v]}$. We can construct a collapsed Gibbs sampler by marginalising out the latent variables in $\mathscr{L}$, giving rise to a variant of the hierarchical Chinese Restaurant Process in which it is straightforward to do inference.

Chinese Restaurant Process A direct representation of a random variable G drawn from a PYP can be obtained from the stick-breaking construction (Pitman, 2002b). But the more indirect representation using the Chinese Restaurant Process (CRP) (Aldous, 1985; Pitman, 2002a) is more suitable here since it relates to distributions over items drawn from such a G. This fits the current setting, where words w are being drawn from a PYP-distributed G.

Imagine that a corpus is created in two phases: Firstly, a sequence of blank tokens x_i is instantiated, and in a second phase lexical identities w_i are assigned to these tokens, giving rise to the observed corpus. In the CRP metaphor , the sequence of tokens x_i are equated with a sequence of customers that enter a restaurant one-by-one to be seated at one of an infinite number of tables. When a customer sits at an unoccupied table k, they order a dish ϕ_k for the table, but customers joining an occupied table have to dine on the dish already served there. The dish ϕ_i that each customer eats is equated to the lexical identity (type) w_i of

[3]It is worth noting that for German the presence and identity of linking elements between c_i and c_{i+1} are in fact governed by the preceding component c_i (Goldsmith and Reutter, 1998).

the corresponding token, and the way in which tables and dishes are chosen gives rise to the characteristic properties of the CRP:

More formally, let $x_1, x_2, \ldots$ be draws from G, while T is the number of occupied tables, C the number of customers in the restaurant, and C_k the number of customers at the k-th table.

Conditioned on preceding customers $x_1, \ldots, x_{i-1}$ and their arrangement, the i-th customer sits at table $k = k'$ according to the following probabilities for the $T + 1$ choices:

$$\Pr(k = k' | \ldots) \propto \begin{cases} C_{k'} - d & \text{occupied table } k' \in [1, T] \\ \theta + dT & \text{unoccupied table } k' = T + 1 \end{cases} \tag{2}$$

Ordering a dish for a new table corresponds to drawing a value ϕ_k from the base distribution G_0, and it is admissible to serve the same kind of dish at multiple tables.

Some characteristic behaviour of the CRP can be observed easily from this description: 1) As more customers join a table, that table becomes a more likely choice for future customers too. 2) Regardless of how many customers there are, there is always a non-zero probability of joining an unoccupied table, and this probability also depends on the number of total tables.

The dish draws can be seen as backing off to the underlying base distribution G_0, an important consideration in the context of the hierarchical variant of the process explained shortly. Note that the strength and discount parameters control the extent to which new dishes are drawn, and thus the extent of reliance on the base distribution.

The predictive probability of a word w given a seating arrangement is given by

$$\Pr(w | \ldots) \propto C_w - dT_w + (\theta + dT)G_0(w), \tag{3}$$

where C_w is the number of customers of type w and T_w the number of tables serving dish w in the restaurant. In smoothing terminology, the first term can be interpreted as applying a discount of dT_w to the observed count C_w of w; the amount of discount therefore depends on the prevalence of the word (via T_w).

Hierarchical CRP When the prior of $G_{\mathbf{u}}$ has a base distribution $G_{\pi(\mathbf{u})}$ that is itself PYP-distributed, as in the HPYLM, the restaurant metaphor changes slightly. In general, each node in the hierarchy has an associated restaurant. Whenever a new table is opened in some restaurant R, another customer is spawned and sent to join the parent restaurant $\text{pa}(R)$. This induces a consistency constraint over the hierarchy: the number of tables T_w in restaurant R must equal the number of customers C_w in its parent $\text{pa}(R)$.

We take care to satisfy this constraint in our model where some restaurants have as base distribution a product of models. Here, when a new table serves a dish $\phi = \hat{w}$ in trigram restaurant $H_{[u,v]}$, a customer c_{Λ_1} joins the corresponding bigram restaurant $G_{[u,v]}$, *and* customers $c_{\Lambda_2}, \ldots, c_{\Lambda_z}, \$$ are sent to the restaurants for $F(\cdot|c_{\Lambda_1}), \ldots, F(\cdot|c_{\Lambda_z})$, respectively.

Sampling Although the CRP allows us to replace the priors with seating arrangements S, those seating arrangements are simply latent variables that need to be marginalised to compute the true posterior predictive probability of a word:

$$p(w|\mathscr{D}) = \int_{S,\Omega} p(w|S,\Omega)p(S,\Omega|\mathscr{D})\, dS\, d\Omega, \tag{4}$$

where $\mathcal{D}$ is the training data and, as before, Ω are the parameters. This integral can be approximated by averaging over m posterior samples (S, Ω) generated using Markov chain Monte Carlo methods. The simple form of the conditionals in the CRP allows us to do a Gibbs update whereby the table index k of a customer is resampled conditioned on all the other variables. Sampling a new seating arrangement S for the trigram HPYLM+c thus corresponds to visiting each customer in the restaurants for $H_{[u,v]}$, removing them while cascading as necessary to observe the consistency across the hierarchy, and seating them anew at some table k'.

In the absence of any strong intuitions about appropriate values for the hyperparameters, we place vague priors over them and use slice sampling[4] (Neal, 2003) to update their values during generation of the posterior samples: $d \sim \text{Beta}(1,1)$; $\theta \sim \text{Gamma}(10, 0.1)$

Lastly, we make the pragmatic approximation that $m = 1$, i.e. predictive probabilities are informed by a single sample[5] (S, Ω), taken after $B > 1$ iterations of burn-in.

Approximate Inference A common criticism of models like ours is that MCMC sampling increases training time unreasonably for an MT pipeline, despite the simplicity of Gibbs sampling.

To address this concern, we will evaluate the viability of using approximate inference in our model, inspired by the interpretation of original interpolated Kneser-Ney smoothing as approximate inference in the HPYLM (Teh, 2006; Goldwater et al., 2006). In each CRP, we constrain all the customers of a type to be seated at the same table, $T_w = 1 \; \forall w$. This changes the predictive probability of a word to

$$\Pr(w|\ldots) \propto C_w - d + (\theta + dT')G_0(w), \tag{5}$$

where T' is now the number of unique types in the restaurant. In the hierarchical model, this implies absolute discounting of the n-gram counts by an amount d.

Under this scheme, the calculation of all C_w and T' across the hierarchy is deterministic. We can therefore obtain the full seating arrangement S from a single pass through the training data. We update the hyperparameters as described in the previous section, although an alternative would be to tune them against perplexity on development data.

5 Experiments

In this section we report on experiments performed to gain insight into the behaviour of the proposed model. The first task we evaluate on is the model's ability to predict a previously unseen text. Our aim is to establish whether the model's account of compounds benefits it without hampering its global performance. We also investigate how the performance depends on the amount of context used when predicting tokens, and on the amount of training data used to estimate the model.

Secondly, we are interested in how the model interacts with a large-scale statistical machine translation system when translating from English to German. Compound words are known to be a challenging aspect of this task, and the aim is to see if specifically accounting for them in the language model can bias a decoder towards better translations. We did not modify other aspects of the translation system, thus it cannot hypothesise "new" compounds and will not benefit from our model's ability to score unseen compounds consisting of observed components.

[4] We employ Mark Johnson's implementation, http://www.cog.brown.edu/~mj/Software.htm

[5] Our preliminary experiments indicated that the posterior over the latent model structure is quite sharply peaked, so that a single sample constitutes a low-variance estimator of the posterior predictive distribution.

5.1 Methods

Data and Tools All data we used are from the WMT11 shared-task.[6] Standard data preprocessing steps comprised normalising punctuation, tokenising and lowercasing all words.

For language model training, we used the union of the news commentary data, Europarl and the news article corpus for 2011. Preprocessing and deduplication yielded a corpus of 59m running tokens, roughly a fifth of all the German monolingual data supplied in WMT11 when using the same preprocessing. No pruning was done on the n-gram counts, but we mapped training tokens to the "unknown" token if they do not appear in the target-side of the bitext (see below). The motivation is that the hypotheses to be scored against the language model during decoding are by definition constrained to this vocabulary.

Our test corpus for the monolingual task is the union of all the WMT11 development data for German (`news-test2008,9,10`, 7065 sentences).

For translation experiments, the preprocessed English-German bitext was filtered to exclude sentences longer than 50 tokens, resulting in 1.7 million parallel sentences; word alignments were inferred from this using the Berkeley Aligner (Liang et al., 2006) and used as a basis from which to extract a Hiero-style synchronous CFG (Chiang, 2007).

The weights of the linear translation models were tuned towards BLEU using `cdec`'s (Dyer et al., 2010) implementation of MERT (Och, 2003). For this, the development set `news-test2008` (2051 sentences) was used, while final BLEU scores are measured on the official test set `newstest2011` (3003 sentences, 171460 tokens), without detokenising or recasing hypotheses.

Compound segmentation For this evaluation, we used an *a priori* segmentation of compounds into parts to build our models. This means we assume a single, fixed analysis of a compound regardless of the context it occurs in, which is necessitated by the fact that our probabilistic model does not specify a step for choosing an analysis. To construct a segmentation dictionary, we ran a supervised[7] compound splitter (Dyer, 2009) on all the words[8] in the training vocabulary, retaining the one-best segmentation. In addition, word-internal hyphens were also taken as segmentation points. Finally, linking elements were merged onto components as discussed in §3.2. Any token that is split into more than one part by this procedure is regarded as a compound, and we find that the majority of compounds thus identified consist of one or two parts (Table 1b).

5.2 Compounds as n-grams

Our model is premised on the idea that better probability estimates can be obtained by analysing compounds into their components. To investigate this claim empirically, we trained a variety of 4-gram language models and compare them by how well they predict an unseen text consisting of N tokens. For each model q, we report measurements in terms of perplexity, $\text{PPL}=\exp\left(-1/N \sum_\tau \ln q(\tau)\right)$, calculated over all tokens τ in the text.

It should be noted that the domain of our model is a countably infinite set. According to the generative process of HPYLM+c (§3.2), there is no theoretical limit on the number of parts in a

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

[7] We chose a supervised splitter as the focus of our evaluation is on the language model's subsequent use of the segmentation, not on the quality of the segmentation itself. Unsupervised methods could also be used with our model.

[8] We also included tokens having numerals and at least two letters, e.g. "CO2-handel" (carbon trade)

	En	De	De LM
Sentences	1.7m	1.7m	2.4m
Tokens	49m	38m	59m
Token Types	112k	351k	596k

(a) Statistics of training corpora.

Parts per Compound	Compound Types
2	197233
3	25128
4	1194
≥5	59

(b) Compound types by length.

Table 1: Summary of training data and compound segmentation.

compound; there is always a non-zero probability of adding another modifier c from $\mathscr{M}$ to a partially formed compound. In this evaluation, we used the probabilities supplied by HPYLM+c without normalising over the finite vocabulary $\mathscr{W}$. Consequently, a comparison to baseline models that have a finite domain is somewhat biased in their favour.

Our main model of interest is HPYLM+c using the Λ^{ling} segmentation and a bigram model F_{HPYLM} over modifiers. To measure the importance of adhering to linguistic intuition, we also evaluate the variant using Λ^{inv}, other things equal. As baselines we used an interpolated, modified Kneser-Ney model (mKN) and an HPYLM. For the sampling-based models, we took one sample from the posterior after $B = 300$ iterations of burn-in.

We find that the main model achieves a slightly lower perplexity than HPYLM, which in turn beats the mKN baseline by 1.9% (Table 2a). The use of the linguistically implausible scheme Λ^{inv} has a noticeably detrimental effect on performance.

	Perplexity
mKN	299.9
HPYLM	294.1
F_{HPYLM} Λ^{ling}	293.6
F_{HPYLM} Λ^{inv}	305.5

(a) Performance of 4-gram models against baselines. Lower is better.

	n=2	**n=3**	**n=4**
mKN	394.5	307.2	299.9
HPYLM	396.6	303.3	294.1
F_{HPYLM} Λ^{ling}	390.0	299.3	293.6

(b) Test-set perplexity for different n-gram orders.

Table 2: Comparison of language models and effect of n-gram order.

For a more qualitative insight into the model performance, we did a further direct comparison of our main model and the mKN baseline by ranking test set compounds by the difference in probability value that each model assigns to the n-gram. The test compounds where the compound model does best (Table 3 top) are all words for which an analysis into a context-dependent head and modifiers should clearly be beneficial. For example, in scoring the phrases "wochen vor den präsidentschaftswahlen" (weeks before the presidential elections) and "tage vor den parlamentswahlen" (days before the parliamentary elections), the head "wahlen" is having a mutually reinforcing effect. In contrast, we find that the cases where the mKN baseline model does best (Table 3 bottom) feature various words that are not strictly speaking compounds, but largely artefacts of our segmentation method: e.g. mistakes such as "ging+rich" or "wissen+schaften", or greediness from splitting on hyphens, e.g. "ki-+moon". These are words where our compound model's smoothing is hurting performance, since it allocates some

HPYLM+c better	Δ
gegen die umstrittene **wieder+wahl**	0.058
aufbau der afghanischen **sicherheits+kräfte**	0.036
dessen zentralen **gesichts+punkten**	0.035
in annapolis , **mary+land**	0.035
wochen vor den **präsidentschafts+wahlen**	0.032
dieses vertrauen nicht **miss+brauchen**	0.030
für psychiatrie und **psycho+therapie**	0.028
tage vor den **parlaments+wahlen**	0.028
reduktion der **treibhausgas+emissionen**	0.025
in einem unblutigen **militär+putsch**	0.021
Baseline (mKN) better	**Δ**
, newt **ging+rich**	0.511
nächtlichem **flug+lärm**	0.449
generalsekretär ban **ki-+moon**	0.423
in st. **peters+burg**	0.420
im 17. **jahr+hundert**	0.419
saalpublikums in st. **peters+burg**	0.359
militanten klerikers moqtada **al-+sadr**	0.352
un-hochkommissarin für **menschen+rechte**	0.286
schwebt in **lebens+gefahr**	0.231
der akademie der **wissen+schaften**	0.212

Table 3: Compounds from the monolingual test set for which HPYLM+c outperforms mKN by the largest margin (top) and vice-versa (bottom). We define the margin Δ as the difference in probability that the models assign to the given test n-gram.

probability mass toward observing other modifiers with the head, which in the case of these proper nouns will not happen. This is evidence of success on the part of our model's underlying mechanism, but demonstrates that more care should be taken with the particular segmentation method used.

5.3 Scaling

Here we consider the behaviour of our model under scaling along two dimensions: n-gram order and training data size.

Our model reduces data sparsity by generalising over different compounds that have the same head. But this happens at the maximal n-gram order, meaning the full surface form is not available in the lower-order conditional distributions. There may be cases where this amounts to "premature back-off" when the lower-order distributions are very informative for a particular surface form.

To see if this has an observable effect, we performed an additional experiment using orders $n = 2$ and $n = 3$. The results in Table 2b indicate that we maintain a lower perplexity than the baselines.

For $n = 2$ and $n = 3$, the sampler had not fully converged after 300 iterations. We suspect this is due to the higher entropy in the distributions governing the seating assignments: If $n = 2$, there should be more customers (and therefore more seating configurations) in the

average restaurant for context-length 2 than in the same restaurant if n is larger. This did not affect perplexity, which was stable when evaluating with different individual samples from the posterior around 300 iterations.

The other dimension of scaling is training data size. We drew random subsamples of different sizes from our training corpus for training further language models.

For small data sizes, the baseline models achieve a noticeably lower perplexity than our compound model. This is contrary to the effect we expected in light of the sparsity reduction our model brings. We suspect that this is primarily due to the lack of normalising the model over a finite vocabulary. For larger sizes, it is competitive against the baselines once more.

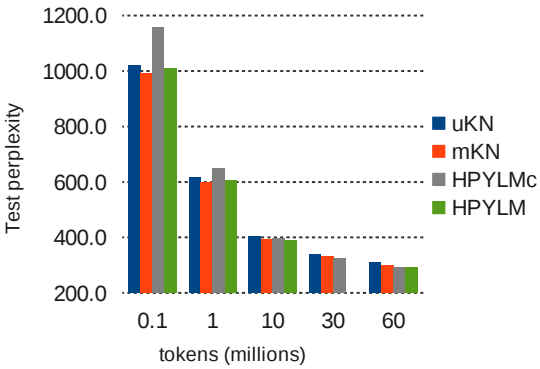


Figure 3: Test perplexity for different sizes of training data, keeping $n = 4$ fixed. uKN and mKN are original and modified Kneser-Ney, respectively, both using interpolation.

5.4 Effect on Translation

The performance of an n-gram language model in an intrinsic evaluation does not necessarily correlate with the effect it has when used as part of a translation system. We thus conducted a separate translation experiment, comparing the quality of the output produced by the translation system described in §5.1 when using different language models.

In terms of BLEU score, we do not find a meaningful difference between the various systems (Table 4a). The system using our main model matches the two baselines, a result that indicates our more expressive modelling is not sacrificing any performance in this task. This is an important outcome, as it means we avoid a common pitfall whereby a new model is proposed to target some specific phenomenon, does so successfully but then sacrifices performance globally. The linguistically implausible segmentation scheme again performs slightly worse.

When using the approximate inference scheme, denoted by `1tbl` in these results, we firstly find that language model training reduces to a trivial amount of time compared to the proper samplers; for `1tbl`, the posterior likelihood converged fully within 5 iterations, where an iteration comprises merely resampling the hyperparameters. By contrast, the posterior likelihood

	PPL	BLEU
mKN	299.9	13.9
HPYLM	294.1	13.9
F_{HPYLM}, Λ^{ling}	293.6	13.9
F_{HPYLM}, Λ^{inv}	305.5	13.7
F_{HPYLM}, Λ^{ling} `1tbl`	355.4	13.6

(a) BLEU over 3003 test sentences with single references. The standard deviation in BLEU score across the three independent runs varied between 0.1 and 0.3. For reference, we also show the perplexity of each model on the monolingual test set.

	P	R	F
mKN	25.4	17.1	20.5
HPYLM	24.3	**17.5**	20.4
F_{HPYLM}, Λ^{ling}	**27.5**	17.3	**21.3**
F_{HPYLM}, Λ^{inv}	23.7	17.2	19.9

(b) Precision, Recall and F-score for compounds in the translation output, relative to the reference set containing 2652 compounds. Each value is calculated across the union of hypotheses produced by decoding the test set with the weights obtained from the three independent runs.

Table 4: Translation results over three MERT runs, using 4-gram language models.

under proper sampling was still improving marginally after 300 iterations for the other models, where one iteration comprises hyperparameter resampling *and* a pass through all training tokens to resample their seating assignments in the CRP.

The `1tbl` model achieved a worse perplexity in the monolingual evaluation task, but with only a small negative effect on BLEU score compared to the baseline (Table 4a). This result suggests there is some leeway in the development of models in the HPYLM framework to explore in future work: model complexity can be pushed up by trading off predictive accuracy against training time.

Next, we turn to a more fine-grained look at the translation output. The BLEU metric is likely to miss small improvements in translation quality. Moreover, in our test corpus only 2652 of the 72661 reference tokens are compounds; a moderate improvement in generating them is unlikely to have a big impact on the BLEU.

To establish whether the model aids in the translation of compounds in particular, we measured the accuracy of hypotheses produced by the different translation systems against the reference translations. We use the standard metrics of precision (correct compounds as a fraction of all compounds output) and recall (correct compounds as a fraction of the compounds in the references).

The results in Table 4b show that using our model increases compound precision by 12% against the HPYLM baseline and 8% against the Kneser-Ney baseline (relative increases). The fact that recall remains stable proves that the gain in precision is not achieved simply by the system being more conservative about outputting compounds in the first place.

6 Related Work

Bilmes and Kirchhoff (2003) proposed a more general framework for n-gram language modelling, which can also be used for implementing sparsity reduction measures. Their Factored Language Model (FLM) views a word as a vector of features, such that a particular feature value is generated conditioned on some history of preceding feature values. This allows one to construct n-gram models with dependencies among sequences of PoS tags or semantic classes in addition to standard word-based dependencies. It should be possible to encode a model with structure comparable to ours in the FLM framework, but it does not lend itself naturally to

having a variable number of features depending on the predicted token in the way our model allows a variable number of parts in a compound.

Another common approach for addressing the sparsity effects of compounding (Koehn and Knight, 2003; Koehn et al., 2008; Stymne, 2009; Berton et al., 1996), and rich morphology (Habash and Sadat, 2006; Geutner, 1995), has been to use pre/post-processing with an otherwise unmodified translation system or speech recognition system. This approach views the existing machinery as adequate and shifts the focus to finding a more appropriate segmentation of words into tokens, i.e. compounds into parts or words into morphemes, thus achieving a vocabulary reduction. The downside of such a method is that training a standard n-gram language model on pre-segmented data introduces unwanted effects: in the case of German compounds, the split-off modifiers would take precedence in a split-off head's n-gram context, and during back-off the actual word-context information is discarded first. The problem is similar when modelling sequences of morphemes as n-grams, and earlier work in speech recognition has shown that taking steps against this effect can improve recognition accuracy (Ircing et al., 2001). Pre-processing also often requires heuristics to guard against over/under-segmentation, which do not generalise well to different settings or languages.

Our work is also subject to the whims of our compound segmentation method, but the model is more robust since it does retain the original surface form of the word – recall that the decomposition step amounts to interpolated back-off.

Baroni and Matiasek (2002) proposed basic models of German compounds for use in predictive text input, exploiting the same link between right-headedness and context as we have, although their focus was restricted to compounds with two components.

In terms of Bayesian modelling, the PYP has been found to be very useful in a variety of tasks, including word segmentation, speech recognition, domain adaption and unsupervised PoS tagging (Goldwater et al., 2006; Mochihashi et al., 2009; Huang and Renals, 2007; Neubig et al., 2010; Wood and Teh, 2009; Blunsom and Cohn, 2011). In all cases its power-law scaling and ease of extensibility via the base distribution allowed the formulation of interesting models that achieved competitive results.

7 Conclusion

We have demonstrated how an existing hierarchical Bayesian model can be used to build an n-gram language model that is informed by intuitions about the specific linguistic phenomenon of closed-form compounds. While our focus was on compounds, we argue that this approach can be useful for other phenomena, such as rich morphology more generally, where data sparsity creates smoothing problems for n-gram language models.

Our empirical results support the conclusions that the increased model expressiveness has a positive impact on the monolingual task of predicting unseen German text, outperforming a competitive Kneser-Ney baseline. When used as part of an English→German translation system, there was little effect on the BLEU metric, but the model was associated with an increase in the F-score for generating correct compounds during translation.

Future work will entail extending the translation system to hypothesise novel compounds, a situation where a productive language model should be vital for generating fluent translations. Further modelling work is therefore needed to handle novel compounds that occur in the n-gram history.

Acknowledgements

We thank the anonymous reviewers for their feedback, and acknowledge the support of The Rhodes Trust (Botha), EPSRC grant number EP/I010858/1 (Blunsom) and the U. S. Army Research Office under contract/grant number W911NF-10-1-0533 (Dyer).

References

Aldous, D. (1985). Exchangeability and related topics. In *École d'Été de Probabilités de Saint-Flour XIII — 1983*, volume 1117 of *Lecture Notes in Mathematics*, pages 1–198. Springer.

Baroni, M. and Matiasek, J. (2002). Predicting the components of German nominal compounds. In *Proc. of the European Conference on Artificial Intelligence (ECAI)*, pages 470–474.

Berton, A., Fetter, P., and Regel-Brietzmann, P. (1996). Compound Words in Large-Vocabulary German Speech Recognition Systems. In *Proc. of the International Conference on Spoken Language Processing (ICSLP)*, volume 2, pages 1165–1168. IEEE.

Bilmes, J. A. and Kirchhoff, K. (2003). Factored language models and generalized parallel backoff. In *Proc. of NAACL-HLT (short papers)*, pages 4–6. Association for Computational Linguistics.

Blunsom, P. and Cohn, T. (2011). A Hierarchical Pitman-Yor Process HMM for Unsupervised Part of Speech Induction. In *Proc. of ACL*. Association for Computational Linguistics.

Botha, J. A. (2012). Hierarchical Bayesian Language Modelling for the Linguistically Informed. In *Proc. of the EACL Student Research Workshop*, pages 64–73.

Chiang, D. (2007). Hierarchical Phrase-Based Translation. *Computational Linguistics*, 33(2):201–228.

Dyer, C. (2009). Using a maximum entropy model to build segmentation lattices for MT. In *Proc. of NAACL*, pages 406–414. Association for Computational Linguistics.

Dyer, C., Lopez, A., Ganitkevitch, J., Weese, J., Ture, F., Blunsom, P., Setiawan, H., Eidelman, V., and Resnik, P. (2010). cdec: A Decoder, Alignment, and Learning framework for finite-state and context-free translation models. In *Proc. of ACL (Demonstration session)*, pages 7–12. Association for Computational Linguistics.

Geutner, P. (1995). Using morphology towards better large-vocabulary speech recognition systems. In *Proc. of International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 445–448.

Goldsmith, J. and Reutter, T. (1998). Automatic Collection and Analysis of German Compounds. In F. Busa F. et al., editor, *The Computational Treatment of Nominals*, pages 61–69. Universite de Montreal, Canada.

Goldwater, S., Griffiths, T. L., and Johnson, M. (2006). Interpolating Between Types and Tokens by Estimating Power-Law Generators. In *Advances in Neural Information Processing Systems, Volume 18*.

Habash, N. and Sadat, F. (2006). Arabic preprocessing schemes for statistical machine translation. In *Proc. of NAACL-HLT*, pages 49–52. Association for Computational Linguistics.

Huang, S. and Renals, S. (2007). Hierarchical Pitman-Yor Language Models For ASR in Meetings. In *Proc. of Workshop on Automatic Speech Recognition and Understanding*, pages 124–129. IEEE.

Ircing, P., Krbec, P., Hajič, J., Psutka, J., Khudanpur, S., Jelinek, F., and Byrne, W. (2001). On large vocabulary continuous speech recognition of highly inflectional language - Czech. In *Proc. of Interspeech*, pages 487–490.

Koehn, P., Arun, A., and Hoang, H. (2008). Towards better Machine Translation Quality for the German – English Language Pairs. In *Proc. of Workshop on Statistical Machine Translation*, pages 139–142. Association for Computational Linguistics.

Koehn, P. and Knight, K. (2003). Empirical Methods for Compound Splitting. In *Proc. of EACL*, pages 187–193. Association for Computational Linguistics.

Liang, P., Taskar, B., and Klein, D. (2006). Alignment by Agreement. In *Proc. of NAACL-HLT*, pages 104–111. Association for Computational Linguistics.

Mochihashi, D., Yamada, T., and Ueda, N. (2009). Bayesian unsupervised word segmentation with nested Pitman-Yor language modeling. In *Proc. of ACL-IJCNLP*, pages 100–108, Suntec, Singapore. Association for Computational Linguistics.

Neal, R. M. (2003). Slice Sampling. *The Annals of Statistics*, 31(3):705–741.

Neubig, G., Mimura, M., Mori, S., and Kawahara, T. (2010). Learning a Language Model from Continuous Speech. In *Proc. of Interspeech*, pages 1053–1056, Chiba, Japan.

Och, F. J. (2003). Minimum Error Rate Training in Statistical Machine Translation. In *Proc. of ACL*, pages 160–167.

Pitman, J. (2002a). Combinatorial stochastic processes. Technical report, Department of Statistics, University of California at Berkeley.

Pitman, J. (2002b). Poisson-Dirichlet and GEM Invariant Distributions for Split-and-Merge Transformations of an Interval Partition. *Combinatorics, Probability and Computing*, 11(05):501–514.

Pitman, J. and Yor, M. (1997). The two-parameter Poisson-Dirichlet distribution derived from a stable subordinator. *The Annals of Probability*, 25:855–900.

Stymne, S. (2009). A comparison of merging strategies for translation of German compounds. In *Proc. of EACL Student Research Workshop*, pages 61–69. Association for Computational Linguistics.

Teh, Y. W. (2006). A hierarchical Bayesian language model based on Pitman-Yor processes. In *Proc. of ACL*, pages 985–992. Association for Computational Linguistics.

Toman, J. (1992). Compound. In Bright, W., editor, *International Encyclopedia of Linguistics*, volume 1, pages 286–288. Oxford University Press.

Wood, F. and Teh, Y. W. (2009). A Hierarchical Nonparametric Bayesian Approach to Statistical Language Model Domain Adaptation. In *Proc. of International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 607–614, Clearwater Beach, Florida, USA.

Can Spanish Be Simpler?
LexSiS: Lexical Simplification for Spanish

Stefan BOTT Luz RELLO Biljana DRNDAREVIC Horacio SAGGION
TALN / DTIC
Universitat Pompeu Fabra
Barcelona, Spain
{stefan.bott,luz.rello,biljana.drndarevic,horacio.saggion}@upf.edu

ABSTRACT
Lexical simplification is the task of replacing a word in a given context by an easier-to-understand synonym. Although a number of lexical simplification approaches have been developed in recent years, most of them have been applied to English, with recent work taking advantage of parallel monolingual datasets for training. Here we present LexSiS, a lexical simplification system for Spanish that does not require a parallel corpus, but instead relies on freely available resources, such as an on-line dictionary and the Web as a corpus. LexSiS uses three techniques for finding a suitable word substitute: a word vector model, word frequency, and word length. In experiments with human informants, we have verified that LexSiS performs better than a hard-to-beat baseline based on synonym frequency.

TITLE AND ABSTRACT IN SPANISH

¿Puede ser el Español más simple?
LexSiS: Simplificación Léxica en Español

La tarea de simplificación léxica consiste en sustituir una palabra en un contexto determinado por un sinónimo que sea más sencillo de comprender. Aunque en los últimos años han aparecido algunos sistemas para desempeñar esta tarea, la mayoría de ellos se han desarrollado para el inglés y hacen uso de corpus paralelos. En este artículo presentamos LexSiS, un sistema de simplificación léxica en español que utiliza recursos libremente disponibles tales como un diccionario en línea o la Web como corpus, sin la necesidad de acudir a la creación de corpus paralelos. LexSiS utiliza tres técnicas para encontrar un sustituto léxico más simple: un modelo vectorial basado en palabras, la frecuencia de las palabras y la longitud de la palabras. Una evaluación realizada con tres anotadores demuestra que para algunos conjuntos de datos LexSiS propone sinónimos más simples que el sinónimo más frecuente.

KEYWORDS: Lexical Simplification, Text Simplification, Textual Accessibility, Word Sense Disambiguation, Spanish.

KEYWORDS IN SPANISH: Simplificación Léxica, Simplificación Textual, Accesibilidad Textual, Desambiguación, Español.

Proceedings of COLING 2012: Technical Papers, pages 357–374,
COLING 2012, Mumbai, December 2012.

1 Introduction

Automatic text simplification is an NLP task that has received growing attention in recent years (Chandrasekar et al., 1996; Carroll et al., 1998; Siddharthan, 2002; Aluísio et al., 2008; Zhu et al., 2010). Text simplification is the process of transforming a text into an equivalent which is easier to read and to understand than the original, preserving, in essence, the original content. This process may include the manipulation of several linguistic layers, and consists of sub-tasks such as syntactic simplification, lexical simplification, content reduction and the introduction of clarifications and definitions. Historically, text simplification started as a task mainly intended as a preprocessing stage in order to make other NLP tasks easier (Chandrasekar et al., 1996; Siddharthan, 2002). However, the task of simplifying a text also has a high potential to help people with various types of reading comprehension problems (Carroll et al., 1998; Aluísio and Gasperin, 2010). For example, lexical simplification by itself, without syntactic simplification, can be helpful for users with some cognitive conditions, such as aphasic readers or people with dyslexia (Hyönä and Olson, 1995). This second context in which text simplification is carried out is closely related to social initiatives which promote easy-to-read material, such as the Simple English section of the Wikipedia.[1] There are also various national and international organizations dedicated to the (mostly human) production of simple and simplified text.

Lexical simplification, an indispensable component of a text simplification system, aims at the substitution of words by simpler synonyms, where the evident question is: "What is a simpler synonym?". The lion's share of the work on lexical simplification has been carried out for English. In this paper, we present LexSiS, the first system for the lexical simplification of Spanish text, which proposes and evaluates a solution to the previous question. LexSiS is being developed in the context of the Simplext project (Saggion et al., 2011), which aims at improving text accessibility for people with cognitive impairments. Until now text simplification in Spanish has concentrated mainly on syntactic simplification (Bott and Saggion, 2012). Lexical and syntactic simplification are tasks which are very different in nature. Working with Spanish presents particular challenges, most notably dealing with the lack of large-scale resources which could be used for our purposes.

LexSiS uses (i) a word vector model to find possible substitutes for a target word and (ii) a simplicity computation procedure grounded on a corpus study and implemented as a function of word length and word frequency. LexSiS uses available resources such as the free thesaurus OpenThesaurus and a corpus of Spanish documents from the Web. The approach we take here serves to test how well relatively simple open domain resources can be used for lexical simplification. Since comparable resources can be found for many other languages, our approach is, in principle, language independent. As will be shown in this paper, by using contextual information and a well-grounded simplicity criterion, LexSiS is able to outperform a hard-to-beat frequency-based lexical replacement procedure.

Next section discusses related work on text simplification with particular emphasis on lexical simplification. Section 3 presents the analysis of a sample of original and simplified texts to design a word simplicity criteria. In Section 4 we present the resources we use for the development of LexSiS, while in Section 5 we describe our lexical simplification approach. We present the evaluation design in Section 6 and discuss the obtained results in Section 7. Finally, in Section 8 we summarize our findings and indicate possible ways to improve our results.

[1] `http://simple.wikipedia.org/`

2 Related Work

Text simplification has by now become a well-established paradigm in NLP, combining a number of rather heterogeneous sub-tasks, such as syntactic simplification, content reduction, lexical simplification and the insertion of clarification material. In this paper, we are only interested in lexical simplification as one of the various aspects of text simplification. Lexical simplification requires, at least, two things: a way of finding synonyms (or, in some cases, hyperonyms), and a way of measuring lexical *complexity* (or simplicity, see Section 3). Note that applying word sense disambiguation can improve the accuracy of the simplification. Consider trying to simplify the word *hogar* in the following sentence: *La madera ardía en el hogar ('The wood was burning in the fireplace')*. The most frequent synonym of *hogar* is *casa ('house')*; however, choosing this word for simplification would produce the sentence *La madera ardía en la casa ('The wood was burning in the house')*, which does not preserve the meaning of the original sentence. Choosing the correct meaning of *hogar*, in this case *'fireplace'*, is important for lexical simplification.

Early approaches to lexical simplification (Carroll et al., 1998; Lal and Ruger, 2002; Burstein et al., 2007) often used WordNet in order to find appropriate word substitutions, in combination with word frequency as a measure of lexical simplicity. Bautista et al. (2011) use a dictionary of synonyms in combination with a simplicity criterion based on word length. De Belder et al. (2010) apply explicit word sense disambiguation, with a Latent Words Language Model, in order to tackle the problem that many of the target words to be substituted are polysemic.

More recently, the availability of the Simple English Wikipedia (SEW) (Coster and Kauchak, 2011b), in combination with the "ordinary" English Wikipedia (EW), made a new generation of text simplification approaches possible, which use primarily machine learning techniques (Zhu et al., 2010; Woodsend et al., 2010; Woodsend and Lapata, 2011b; Coster and Kauchak, 2011a; Wubben et al., 2012). This includes some new approaches to lexical simplification. Yatskar et al. (2010) use edit histories for the SEW and the combination of SEW and EW in order to create a set of lexical substitution rules. Biran et al. (2011) also use the SEW/EW combination (without the edit history of the SEW), in addition to the explicit sentence alignment between SEW and EW. They use WordNet as a filter for possible lexical substitution rules. Although they do not apply explicit word sense disambiguation, their approach is *context-aware*, since they use a cosine-measure of similarity between a lexical item and a given context, in order to filter out possibly harmful rule applications which would select word substitutes with the wrong word sense. Their work is also interesting because they use a Vector Space Model to capture the lexical semantics and, with that, their context preferences.

Finally, there is a recent tendency to use statistical machine translation techniques for text simplification (defined as a monolingual machine translation task). Coster and Kauchak (2011a) and Specia (2010), drawing on work by Caseli et al. (2009), use standard statistical machine translation machinery for text simplification. The former uses a dataset extracted from the SEW/EW combination, while the latter is noteworthy for two reasons: first, it is one of the few statistical approaches that targets a language different from English (namely Brazilian Portuguese); and second, it is able to achieve good results with a surprisingly small bi-data-set of only 4,483 sentences. Specia's work is closely related to the PorSimples project, described in Aluísio and Gasperin (2010). In this project a dedicated lexical simplification module was developed, and it uses a thesaurus and a lexical ontology for Portuguese. They use word frequency as a measure for simplicity, but apply no word sense disambiguation.

3 Corpus Analysis

As the basis for the development of LexSiS, we have conducted an empirical analysis of a small corpus of news articles in Spanish, the Simplext Corpus (Bott and Saggion, 2011). It consists of 200 news articles, 40 of which have been manually simplified. Original texts and their corresponding simplifications have been aligned at the sentence level, thus producing a parallel corpus of a total of 590 sentences (246 and 324 in the original and simplified sets respectively). All texts have been annotated using Freeling, including part-of-speech tagging, named entity recognition and parsing (Padró et al., 2010).

Our methodology, explained more in depth in Drndarevic and Saggion (2012), consists in observing lexical changes applied by trained human editors and preparing their computational implementation accordingly. In addition to that, we conduct quantitative analysis on the word level in order to compare frequency and length distributions in the sets of original and simplified texts. Earlier work on lexical substitution has largely concentrated on word frequency, with occasional interest for word length as well (Bautista et al., 2009). It has also been shown that lexical complexity correlates with word frequency: more frequent words present less cognitive effort for the reader (Rayner and Duffy, 1986). Our analysis is motivated by the desire to test the relevance of these factors in the text genre we treat and the possibility of their combined influence on the choice of the simplest out of a set of synonyms to replace a difficult input word.

We observe a high percentage of named entities (NE) and numerical expressions (NumExp) in our corpus, due to the fact that it is composed of news articles, which naturally abound in this kind of expressions. NEs and NumExps have been discarded from the frequency and length analysis because they are tagged as a whole by Freeling, and this presents us with two difficulties. First, some expressions, such as *30 millones de dólares ('30 million dolars')* or *Programa Conjunto de las Naciones Unidas sobre el VIH/sida ('Joint United Nations Programme on HIV/AIDS')*, are extremely long words (some exceed 40 characters in length) and are not found in the dictionary; thus, we cannot assign them a frequency index. Second, such expressions are not replaceable by synonyms, but require a different simplification approach.

We conduct word length and frequency analysis from two angles. First, we analyse the totality of the words in the parallel corpus. Second, we analyse all lexical units (including multi-word expressions, e.g. complex prepositions) that have been substituted with a simpler synonym. These pairs of lexical substitutions (O-S) have been included in the so-called Lexical Substitution Table (LST) and are used for evaluation purposes (see Section 6).

3.1 Word Length

Analysing the total of 10,507 words (6,595 and 3,912 in the original and simplified sets respectively), we have observed that the most prolific words in both sets are two character words, the majority of which are function words (97.61% in O and 88.97% in S). Two to seven-character words are more abundant in the S set, while longer words are slightly more common in the O set. The S set contains no words with more than 15 characters. Analysis of the pairs in the LST has given us similar results: almost 70% of simple words are shorter than their original counterparts.

On the whole, we can conclude that in S texts there is a tendency towards using shorter words of up to ten characters, with one to five-character words taking up 64.10% of the set and one to ten-character words accounting 95.54% of the content.

3.2 Word Frequency

To analyse the frequency, a dictionary based on the Referential Corpus of Contemporary Spanish (Corpus de Referencia del Español Actual, CREA)[2] has been compiled for the purposes of the Simplext project. Every word in the dictionary is assigned a frequency index (FI) from 1 to 6, where 1 represents the lowest frequency and 6 the highest. We use this resource for the corpus analysis because it allows easy categorisation of words according to their frequency and elegant presentation and interpretation of results. However, in Section 5 this method is abandoned and relative frequencies are calculated based on occurrences of given words in the training corpus, so as to ensure that words not found in the above mentioned dictionary are also covered.

In the parallel corpus, we have documented words with FI 3, 4, 5 and 6, as well as words not found in the dictionary. The latter are assigned FI 0 and termed *rare words*. This category consists of infrequent words such as *intransigencia* (*'intransigence'*), terms of foreign origin, like *e-book*, and a small number of multi-word expressions, such as *a lo largo de* (*'during'*). The latter are recognized as multi-word expressions by Freeling, but are not included in the dictionary as such. The ratio of these expressions with respect to total is rather small (1.08% in O and 0.59% in S), so it should not significantly influence the overall results, presented in Table 1.

Frequency index	Original	Simplified
Freq. 0	10,53%	4,71%
Freq. 3	1,36%	0,74%
Freq. 4	1,35%	1,00%
Freq. 5	6,68%	5,67%
Freq. 6	80,08%	87,88%

Table 1: The distribution of n-frequency words in original and simplified texts.

We observe that lower frequency words (FI 3 and FI 0) are around 50% more common in O texts than in S texts, while the latter are somewhat more saturated in highest frequency words. As a general conclusion we observe that simple texts (S set) make use of more frequent words from CREA than their original counterparts (O set).

In order to combine the factors of word length and frequency, we have additionally analysed the length of all the words in the category of *rare words*. We have found that rare words are largely (72.44% in O and 77.44% in S) made up of seven to nine-character words, followed by longer words of up to twenty characters in O texts (39.42%) and fourteen characters in S texts (29.88%).

We are, therefore, lead to believe that there is a degree of connection between the factors of word length and word frequency, and that these are to be combined when scores are assigned to synonym candidates. In Section 5.1 we propose criteria for determining word simplicity exploiting these findings.

4 Resources

As we already mentioned in Section 2, most attempts to resolve the problem of lexical simplification are concentrated on English and, in recent years, Simple English Wikipedia in combination with the "ordinary" English Wikipedia has become a valuable resource for the study of text

[2] `http://corpus.rae.es/creanet.html`

simplification in general, and lexical simplification in particular. For Spanish, like for most other languages, no comparably large parallel corpora are available.

Some approaches to lexical simplification make use of WordNet (Miller et al., 1990) in order to measure the semantic similarity between lexical items and to find an appropriate substitute. While Spanish is one of the languages represented in EuroWordNet (Vossen, 2004), its scope is much more modest. The Spanish part of EuroWordNet contains only 50,526 word meanings and 23,370 synsets, in comparison to 187,602 meanings and 187,602 synsets in the English WordNet 1.5.

4.1 Corpora

The most valuable resources for lexical simplification are comparable corpora which represent the "normal" and a simplified variant of the target language. Although the corpus described in Section 3 served us as a basis for the corpus study and provided us with gold standard examples for the evaluation presented in Section 6, it is not large enough to train a simplification model. We, therefore, made use of an 8M word corpus of Spanish text extracted from the Web to train the vector models in Section 4.3.

4.2 Thesaurus

We use the Spanish OpenThesaurus (version 2),[3] which is freely available under the GNU Lesser General Public License, for the use with OpenOffice.org. This thesaurus lists 21,831 target words (lemmas) and provides a list of word senses for each word. Each word sense is, in turn, a list of substitute words (and we shall refer to them as *substitution sets* hereafter). There is a total of 44,353 such word senses. The substitution candidate words may be contained in more than one of the substitution sets for a target word. The following is the Thesaurus entry for *mono*, which is ambiguous between the nouns *'ape'*, *'monkey'* and *'overall'*, as well as the adjective *'cute'*.

(a) mono|4
- |gorila|simio|antropoide
- |simio|chimpancé|mandril|mico|macaco
- |overol|traje de faena
- |llamativo|vistoso|atractivo|sugerente|provocativo|resultón|bonito

OpenThesaurus lists simple one-word and multi-word expressions, both as target and substitution units. In the current version of LexSiS, we only treat single-word units, but we plan to include the treatment of multi-word expressions in future versions. We counted 436 expressions of the kind, such as *arma blanca* (*'stabbing or cutting weapon'*) or *de esta forma* (*'in this manner'*). Some of those expressions are very frequent and are used as tag phrases. The treatment of multi-word expressions only requires a multi-word detection module as an additional resource.

4.3 Word Vector Model

In order to measure lexical similarity between words and contexts, we used a Word Vector Model (Salton et al., 1975). Word Vector Models are a good way of modelling lexical semantics

[3] `http://openthes-es.berlios.de`

(Turney and Pantel, 2010), since they are robust, conceptually simple and mathematically well defined. The 'meaning' of a word is represented as the contexts in which it can be found. A word vector can be extracted from contexts observed in a corpus, where the dimensions represent the words in the context, and the component values represent their frequencies. The context itself can be defined in different ways, such as an n-word window surrounding the target word. Whether two words are similar in meaning can be measured as the cosine distance between the two corresponding vectors. Moreover, vector models are sensitive to word senses. For example, vectors for word senses can be built as the sum of word vectors which share one meaning.

We trained this vector model on the 8M word corpus mentioned in 4.1. We lemmatized the corpus with FreeLing (Padró et al., 2010) and for each lemma type in the corpus we constructed a vector, which represents co-occurring lemmas in a 9-word (actually 9-lemma) window (4 lemmas to the left and to the right). The vector model has n dimensions, where n is the number of lemmata in the lexicon. The dimensions of each vector in the model (i.e. the vector corresponding to a target lemma) represent the lemmas found in the contexts, and the value for each component represents to number of times the corresponding lemma has been found in the 9-word context. In the same process, we also calculated the absolute and relative frequencies of all lemmas observed in this training corpus.

5 LexSiS Method

LexSiS tries to find the best substitution candidate (a word lemma) for every word which has an entry in the Spanish OpenThesaurus. The substitution operates in two steps: first the system tries to find the most appropriate substitution set for a given word, and then it tries to find the best substitution candidate within this set. Here the *best* candidate is defined as the simplest and most appropriate candidate word for the given context. As for the simplicity criterion, we apply a combination of word length and word frequency, and for the determination of appropriateness we perform a simple form of word sense disambiguation in combination with a filter that blocks words which do not seem to fit in the context.

In the first step, we check for each lemma if it has alternatives in OpenThesaurus. If this is the case, we extract a vector from the surrounding 9-word window, as described in Section 4.3. Since each word is a synonym to itself (and might actually be the simplest word among all alternatives), we include the original word lemma in the list of words that represent the word sense. We construct a common vector for each of the word senses listed in the thesaurus by adding all the vectors (resulting from Section 4.3) to the words listed in each word sense. Then, we select the word sense with the lowest cosine distance to the context vector. In the second step, we select the best candidate within the selected word sense, assigning a simplicity score and applying several thresholds in order to eliminate candidates which are either not much simpler or seem to differ too much from the context.

5.1 Simplicity

According to our discussion in Section 3, we calculate simplicity as a combination of word length and word frequency. The task of combining them, however, is not entirely trivial, considering the underlying distribution of lengths and frequencies. In both cases simplicity is clearly not linearly correlated to the observable values. We know that simplicity monotonically decreases with length and monotonically increases with frequency, but a linear combination of the two factors not necessarily behaves monotonically as well. What we need is a score for simplicity, such that for all possible combinations of word lengths and frequencies of two words, w_1 and

w_2, $score(w_1) > score(w_2)$ iff w_1 is simpler than w_2. For this reason, we try to approximate the correlation between simplicity and the observable values at least to some degree.

In the case of length, our corpus study showed that a word with length *wl* is simpler than a word with length $wl + 1$. But the degree to which it is simpler depends on the value of *wl*. The corresponding difference decreases with longer values for *wl*. For words with a very high *wl* value, a difference in simplicity between *wl* words and *wl* - 1 words is not perceived any more. In our corpus, we found that very long words (10 characters and longer) were always substituted with much shorter words with an average length difference of 4.35 characters. In medium length range (from 5 to 9 characters), the average difference was only 0.36 characters, and very short original words (4 characters or shorter) did not tend to be shortened in the simplified version at all. For this reason we use the following formula:[4]

$$score_{wl} = \begin{cases} \sqrt{wl - 4} & \text{if } wl \geq 5\,, \\ 0 & \text{otherwise.} \end{cases}$$

In the case of frequency, we make the standard assumption that word frequency is distributed according to Zipf's law (Zipf, 1935); therefore, simplicity must be similarly distributed (when we abstract away from the influence of word length). In order to get a score which associates simplicity to frequency in a way which comes closer to linearity, we calculate the simplicity score for frequency as the logarithm of the frequency count c_w for a given word:

$$score_{\text{freq}} = \log c_w$$

Now the combination of the two values is

$$score_{\text{simp}} = \alpha_1 score_{wl} + \alpha_2 score_{\text{freq}}$$

where α_1 and α_2 are weights. We determined values for α_1 and α_2 in the following way: we manually selected 100 good simplification candidates proposed by OpenThesaurus for given contexts. We only considered cases which were both indisputable synonyms and clearly perceived as being simpler than the original. Then we calculated the average difference between the scores for word length and word frequency between the the original lemma and the simplified lemma, and took these averaged differences as being the average contribution of length and frequency to the receivable *simplicity* of the lemma. This resulted in $\alpha_1 = -0.39$[5] and $\alpha_2 = 1.11$.

[4]The formula for $score_{wl}$ resulted in quite a stable average value for $score_{wl}(w_{\text{original}}) - score_{wl}(w_{\text{simplified}})$ for the different values of wl in the range of word lengths from 7 to 12, when tested on the gold standard (cf 6.3 below). For longer and shorter words this value was still over-proportionally high or low, respectively, but the difference is less pronounced than with alternative formulas we tried, and much smoother than the direct use of wl counts. In addition, 74% of all observed substitutions fell into that range.

[5]Note that word length is a penalizing factor, since longer words are generally less simple. For this reason, the value for α_1 is negative.

5.2 Thresholds

There are several cases in which we do not want to accept an alternative for a target word, even if it has a high simplicity score. First of all, we do not want to simplify frequent words, even if OpenThesaurus lists them. So we set a cutoff point for frequent words, such that LexSiS does not try to simplify words with a frequency higher than 1% (calculated on the training corpus in 4.3). We also discard substitutes where the difference in the simplicity score with respect to the original word is lower than 0.5, because such words can be expected not to be significantly simpler. We achieved this latter value through experimentation.

Many of the alternatives proposed by OpenThesaurus are in reality not acceptable substitutes. We try to filter out words that do not fit into the context by discarding all candidates whose word vector has a distance with a cosine inferior to 0.013, another value achieved through experimentation.

Finally, there are two cases in which the system does not propose a substitute. First, there are cases where none of the substitution candidates have low enough cosine distance to the context vector (with the threshold of 0.013). There are also cases where the highest scoring substitute is the same as the original lemma. In both cases the original word is preserved.

6 Evaluation

In this section we present the experimental set-up employed to evaluate LexSiS by comparing it with two baselines and a gold standard. The evaluation was conducted thoroughly, rating the degree of simplification and the preservation of meaning of the substitutions.

6.1 Baselines

We employ two baselines:

(a) Random: it replaces the target word with a synonym selected randomly from our resource.

(b) Frequency: it replaces a word with its most frequent synonym provided by the thesaurus, presumed to be the simplest, similar to Devlin and Unthank (2006).

6.2 Gold Standard

As the gold standard we used the synonym pairs composed by the manual lexical substitutions extracted from the corpus described in Section 3. Since current lexical simplification systems for English handle only one-word cases (Yatskar et al., 2010; Biran et al., 2011), only single lexical substitutions were taken intro consideration. For instance, the substitution of the word *pinacoteca (‘art gallery’)* by *museo (‘museum’)* in the following sentence:

(b) (O) El visitante puede contemplar los óleos y esculturas que se exponen en la PINACOTECA.
‘The visitor can appreciate the paintings and sculptures showed in the ART GALLERY.’
(S) El visitante puede contemplar los óleos y esculturas que se exponen en el MUSEO.
‘The visitor can appreciate the paintings and sculptures showed in the MUSEUM.’

With that restriction, we found a total of 26 lexical substitutions in the corpus. We discarded collocations and multi-word expressions, as well as single lexical substitutions which supposed a mayor transformation in the sentence structure. For instance, the word *pese a (‘despite’)*, substituted with *sin embargo (‘however’)*, has changed the structure of the sentence itself:

(c) (O) Amnistía subrayó que Manning se encuentra detenido en "custodia máxima", PESE A carecer de antecedentes por violencia o infracciones disciplinarias durante la detención, lo que significa que está atado de pies y manos durante todas las visitas y se le niega la oportunidad de trabajar y salir de su celda.
'Amnesty underlined that Manning is being held in "maximum custody", DESPITE having no history of violence or disciplinary offenses during detention, which means his hands and feet are tied during all visits and he is denied the opportunity to work and leave his cell.'
(S) Bradley Manning ha tenido un buen comportamiento en la cárcel. SIN EMBARGO, Bradley Manning está atado durante las visitas. Tampoco puede trabajar ni salir de su celda.
'Bradley Manning has exhibited good behavior in prison. HOWEVER, Bradley Manning is tied during visits. He cannot work or leave his cell.'

6.3 Evaluation Dataset

We have three different evaluation datasets.

(T-S/B) Target *vs.* **System and Baselines:** This dataset is composed of 50 unique target words, together with their synonyms generated by LexSiS and the baselines. To create this dataset, we first ran our system for lexical simplification through the original texts from the Spanish Simplext Corpus (Bott and Saggion, 2011). This gave us 739 automatic lexical substitutions. We randomly selected 50 sentences that included one lexical substitution. Subsequently, for each of the examples, we generated two baselines and inserted them in the sentences, obtaining a total of 200 sentences. We manually corrected the ungrammatical examples resulting from lexical substitution.

(T-G/S/B) Target *vs.* **Gold, System and Baselines:** This dataset is composed of the examples which were manually simplified in the gold standard and which were also simplified by our system (12 cases out of the 26 cases of single lexical substitution in the Simplext Corpus). For each of these examples, we also generated two baselines, obtaining a total or 48 sentences to evaluate.

(G-T) Gold *vs.* **Target:** This dataset is composed of the remaining lexical simplification examples of our gold standard (14 instances) and their target words. For these examples our system did not propose a simplified example.

We could set up more evaluation data to compare the system and the baselines. However, we considered 50 sentences (a total of 200 sentences for dataset T-S/B, and 276 different sentences if we consider all data sets) to be reasonable because: (i) It was feasible for the annotators to perform the task; and (ii) previous works in lexical simplification used slightly smaller data sets for evaluation - in Yatskar et al. (2010) they use a total of 200 simplification examples, and in Biran et al. (2011) 130 simplification examples are used. Below, we show two examples of a sentence with its original word (O) and the lexical substitution proposed by our system (S).

(d) (O) De la Iglesia merece este GALARDÓN.
'De la Iglesia deserves this AWARD.'
(S) De la Iglesia merece este PREMIO.
'De la Iglesia deserves this PRIZE.'

(e) (O) Cuenta con una AMPLIA oferta de títulos.
'It has a WIDE range of titles.'
(S) Cuenta con una GRAN oferta de títulos.
'It has a GREAT range of titles.'

6.4 Design

We created a multiple choice questionnaire presenting two sentences for each item. Each item contained one sentence with a simplification example and the same sentence with the target word. These sentences were presented in random order to the annotator (i.e., either as Target *vs.* LexSiS/Gold/Baseline or LexSiS/Gold/Baseline *vs.* Target). The labels for each pair of sentences in comparison with the other were: "simpler", "more complex", "equally simple or complex", "they do not have the same meaning", and "I do not know or I do not understand at least one of the sentences". In Table 2 we show an example of one target word and its corresponding substitutions presented in the sentences.

Type	Sentence
Target	Descubren en Valencia una nueva ESPECIE de pez prehistórico. *A new SPECIES of prehistoric fish is discovered in Valencia.*
LexSIS	Descubren en Valencia un nuevo TIPO de pez prehistórico. *A new TYPE of prehistoric fish is discovered in Valencia.*
Frequency Baseline	Descubren en Valencia un nuevo GRUPO de pez prehistórico. *A new GROUP of prehistoric fish is discovered in Valencia.*
Random Baseline	Descubren en Valencia un nuevo LINAJE de pez prehistórico. *A new LINEAGE of prehistoric fish is discovered in Valencia.*

Table 2: Example extracted from dataset (T-S/B) Target *vs.* LexSIS and Baselines.

6.5 Annotators

Three annotators with no previous annotation experience performed the tests using an on-line form. They were all Spanish native speakers, frequent readers and were not the authors of this paper. To measure the inter-annotator agreement of multiple annotators and multiple classes (five classes, one per label), we used the Fleiss' kappa measure (Fleiss, 1971). The three participants annotated all the instances of the three datasets, achieving a Fleiss' kappa score of 0.330. Hence, we can assume we have a fair agreement (Landis and Koch, 1977), comparable with other inter-annotator agreements in related literature (see Section 7).

7 Results and Discussion

In this section we present the results and discuss the performance of LexSiS. We calculate (in percentage) how well the meaning has been preserved, taking into consideration all the instances (Synonym column). The percentage of simpler, equal and more complex synonyms is calculated among the synonymous instances (Simpler Syn., Equal Syn. and Complex Syn. columns, respectively) and among all the instances, taking into consideration the global performance of LexSiS (Simpler Syn. Glob., Equal Syn. Glob. and Complex Syn. Glob. columns, respectively).[6]

System	Synonym (%)	Simpler Syn. (%)	Equal Syn. (%)	Complex Syn. (%)	Simpler Syn. Glob. (%)	Equal Syn. Glob. (%)	Complex Syn. Glob. (%)
Random	65.03	15.13	24.37	58.82	9.94	**16.02**	38.67
Frequency	66.12	**42.98**	19.01	**38.02**	28.42	12.57	**25.14**
LexSiS	**72.49**	40.88	**17.52**	41.61	**29.63**	12.70	30.16
Gold	97.44	71.05	17.11	11.84	69.23	16.67	11.54

Table 3: Evaluation of all the datasets.

[6]The sum of Simpler Syn. Glob., Equal Syn. Glob. and Complex Syn. Glob. columns equals the number of synonyms (Synonym column) and the sum of Simpler Syn., Equal Syn. and Complex Syn. columns equals 100.

In Table 3 we present the results for all the datasets. The percentage of meaning preservation is higher in LexSiS than in the baselines, and LexSiS offers simpler synonyms (29.63%) than the frequency baseline (28.42%). However, 30.16% of the lexical substitutions suggested by our system are found to be more complex by the annotators. One possible reason for this is that the texts used for this study were already simple from a lexical point of view. For instance, we found only 26 manual lexical substitutions in the Simplext corpus, and our system only offered a lexical substitute for 12 of these manual substitutions. Consequently, 53.84% of the manually substituted target words were not simplified by LexSiS, and 78.57% of these manual substitutions were found to be simpler synonyms by the annotators (see Table 4). Even though our gold standard is small, to the best of our knowledge, this is the only existing resource from which we could generate a gold standard for lexical simplification in Spanish.

System	Synonym (%)	Simpler Syn. (%)	Equal Syn. (%)	Complex Syn. (%)	Simpler Syn. Glob. (%)	Equal Syn. Glob. (%)	Complex Syn. Glob. (%)
Gold	100	78.57	16.67	4.76	78.57	16.67	4.76

Table 4: Evaluation of dataset (G-T): Gold *vs.* Target.

We performed an error analysis to try and discover why LexSiS did not account for the remaining instances from the gold standard, and we found five possible reasons: (1) the target word does not occur in the LexSiS dictionary; (2) the target word is already too frequent to be substituted by LexSiS; (3) the target word and its lexical substitution are the same; (4) the lexical substitution proposed by LexSiS has a very similar ranking to the target word, hence the substitution is not performed; and (5) the target word is discarded because its vectorial distance taking into account the original context is too large.

System	Synonym (%)	Simpler Syn. (%)	Equal Syn. (%)	Complex Syn. (%)	Simpler Syn. Glob. (%)	Equal Syn. Glob. (%)	Complex Syn. Glob. (%)
Random	**57.14**	4.76	42.86	47.62	2.86	25.71	28.57
Frequency	47.22	47.06	17.65	35.29	22.22	**8.33**	16.67
LexSiS	55.56	**65.00**	**15.00**	**20.00**	**36.11**	**8.33**	**11.11**
Gold	95.83	58.70	17.39	21.74	61.11	16.67	16.67

Table 5: Evaluation of dataset (T-G/S/B): Target *vs.* Gold, LexSiS and Baselines.

Among the instances from the gold standard not accounted for by LexSiS, our system presents more lexical substitutions synonymous with the target word than the baselines, and a greater amount of simpler synonyms (65.00%) than the frequency (47.06%) and random (4.76%) baselines (see results for dataset (T-G/S/B) in Table 5). We have found that the dataset (T-S/B) is the only case where the frequency baseline presents a higher percentage of suggesting simpler synonyms (31.29%) than LexSiS (27.33%). Nevertheless, LexSiS shows a higher meaning preservation percentage.

System	Synonym (%)	Simpler Syn. (%)	Equal Syn. (%)	Complex Syn. (%)	Simpler Syn. Glob. (%)	Equal Syn. Glob. (%)	Complex Syn. Glob. (%)
Random	66.00	18.18	20.20	60.61	12.08	**13.42**	40.27
Frequency	72.11	**43.40**	18.87	**37.74**	**31.29**	13.61	**27.21**
LexSiS	**76.00**	35.96	**18.42**	45.61	27.33	14.00	34.67

Table 6: Evaluation of dataset (T-S/B): Target *vs.* LexSiS and Baselines.

These results are consistent with the state of the art of lexical simplification for English. In SemEval-2012 shared task for lexical simplification, only one lexical simplification system

among nine systems presented a higher ranking (0.496) than the frequency baseline (0.471), according to a pairwise kappa metric (Specia et al., 2012).

Our results are comparable with the lexical simplification system for English presented in (Biran et al., 2011), whose frequency baseline for English target words (medium frequency) is very similar to ours (42.86%). The results of their system are higher for English and they make use of a comparable corpus. Our inter-annotator agreement rate (Fleiss' kappa score of 0.330) is also consistent with their pairwise inter-annotator agreement, where kappa score was between 0.35 and 0.53.

Our evaluation methodology includes five possible answers for each pair of instances, as in SIMPL method (Yatskar et al., 2010), although we present the annotators with the words in their original context. SIMPL reaches 66% precision for the 100 top pairs using a probabilistic model based on the edit history in simple Wikipedia.

Other methods, which include syntactic operations as well, approach lexical simplification as a monolingual machine translation problem and use machine translation measures to evaluate them (Zhu et al., 2010), together with human judges (Woodsend and Lapata, 2011a).

In summary, LexSiS is the first approach to lexical simplification in Spanish, a language where only syntactic simplification has been previously performed (Bott and Saggion, 2012). Our system uses free resources, such as a dictionary and the Web as corpus. LexSiS is based on the analysis of a small sample of data and it does not require a parallel corpus to function.

8 Conclusion and Future work

In the past few years, automatic text simplification has rapidly become an important technology for the information society, due to its potential application in information access and the benefits it may bring to people with special needs. One important aspect of text simplification is lexical simplification: the selection of simpler/easier substitutes for difficult words. Automatic lexical simplification has, in most cases, been applied to English datasets, with many recent approaches relying on the availability of large parallel simplified and non-simplified documents to train statistical methods.

In this paper we have presented LexSiS, the first implemented lexical simplification system for Spanish, which does not require a parallel corpus for its implementation. LexSiS, rather, uses available texts found on the Web and a freely available Thesaurus. LexSiS uses a word vector model to identify the correct sense of the word to be replaced, and then selects the simplest synonym using simplicity criteria based on word frequency and word length.

We have evaluated three aspects of LexSiS: (i) its ability to identify a correct synonym for the target to be simplified; (ii) its ability to provide a simpler word substitute; and (iii) its performance as a whole. We have compared LexSiS with two baselines, one of which is a hard-to-beat procedure based on word frequency.

For each of the datasets used in the evaluation, LexSiS shows a higher meaning preservation percentage, offering more synonymous lexical substitutions than the baseline. For the overall of the datasets, LexSiS proposes simpler synonyms than the frequency baseline.

We believe our research makes the following contributions: we present a well-grounded procedure for the simplification of Spanish text, which could possibly be a language independent method, due to its small-scale resources that can be found for many other languages. It repre-

sents the first computational implementation of a lexical simplification algorithm for Spanish. Finally, we use a well-designed and viable evaluation methodology for lexical simplification.

Based on the evaluation, we have found various aspects of the method which need to be taken into further consideration. First, we are facing the problem of out-of-dictionary words. One way to overcome this problem is by inducing possible word-substitutes with methods similar to the word vector model we use here, as well as larger datasets. Second, frequency computation in our procedure is not carried out considering word sense disambiguation, but by performing counts disregarding senses. It is possible that taking the word sense into consideration before computing its frequency could improve the results. Finally, in this work we only consider substitution of single lexical units – the identification of multi-word expressions and collocations, such as *poco a poco* (‘*little by little*’), substituted with its simpler synonym *gradualmente* (‘*gradually*’), will be addressed in future work.

Acknowledgments

We would like to thank Ricardo Baeza-Yates for his wise comments and Joan Codina for mathematical advice on the modeling of simplicity. We are also grateful for the annotators for their effort and patience: Toni Carbajo, Roberto Juan Carlini and Alfonso Rello. We also thank the anonymous reviewers for their constructive comments.

The research described in this paper arises from a Spanish research project called Simplext: An automatic system for text simplification (`http://www.simplext.es`). Simplext is led by Technosite and partially funded by the Ministry of Industry, Tourism and Trade of the Government of Spain, by means of the National Plan of Scientific Research, Development and Technological Innovation (I+D+i), within strategic Action of Telecommunications and Information Society (Avanza Competitiveness, with file number TSI-020302-2010-84). We are grateful to fellowship RYC-2009-04291 from Programa Ramón y Cajal 2009, Ministerio de Economía y Competitividad, Secretaría de Estado de Investigación, Desarrollo e Innovación, Spain and to the doctoral fellowship FI-DGR by Generalitat de Catalunya.

References

Aluísio, S. M. and Gasperin, C. (2010). Fostering digital inclusion and accessibility: the PorSimples project for simplification of Portuguese texts. In *Proceedings of the NAACL HLT 2010 Young Investigators Workshop on Computational Approaches to Languages of the Americas*, YIWCALA ’10, pages 46–53, Stroudsburg, PA, USA. Association for Computational Linguistics.

Aluísio, S. M., Specia, L., Pardo, T. A. S., Maziero, E. G., and de Mattos Fortes, R. P. (2008). Towards Brazilian Portuguese automatic text simplification systems. In *ACM Symposium on Document Engineering*, pages 240–248.

Bautista, S., Gervás, P., and Madrid, R. (2009). Feasibility analysis for semiautomatic conversion of text to improve readability. In *The Second International Conference on Information and Communication Technologies and Accessibility*.

Bautista, S., León, C., Hervás, R., and Gervás, P. (2011). Empirical identification of text simplification strategies for reading-impaired people. In *European Conference for the Advancement of Assistive Technology*, Maastricht, the Netherlands.

Biran, O., Brody, S., and Elhadad, N. (2011). Putting it simply: a context-aware approach to lexical simplificaion. In *Proceedings of the 49th Annual Meeting of the Association for*

Computational Linguistics: Human Language Technologies, pages 496–501, Portland, Oregon, USA. Association for Computational Linguistics.

Bott, S. and Saggion, H. (2011). An unsupervised alignment algorithm for text simplification corpus construction. In *Proceedings of the Workshop on Monolingual Text-To-Text Generation*, pages 20–26, Portland, Oregon. Association for Computational Linguistics.

Bott, S. and Saggion, H. (2012). Text Simplification Tools for Spanish. In *Proceedings of the eighth international conference on Language Resources and Evaluation (LREC 2012)*, Istanbul, Turkey. ELRA.

Burstein, J., Shore, J., Sabatini, J., Lee, Y.-W., and Ventura, M. (2007). The automated text adaptation tool. In *Proceedings of the Annual Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. (Demonstrations)*, pages 3–4.

Carroll, J., Minnen, G., Canning, Y., Devlin, S., and Tait, J. (1998). Practical Simplification of English Newspaper Text to Assist Aphasic Readers. In *Proc. of AAAI-98 Workshop on Integrating Artificial Intelligence and Assistive Technology*, pages 7–10.

Caseli, H. M., Pereira, T. F., Specia, L., Pardo, T. A. S., Gasperin, C., and Aluísio, S. M. (2009). Building a brazilian portuguese parallel corpus of original and simplified texts. In *10th Conference on Intelligent Text Processing and Computational Linguistics (CICLing 2009)*.

Chandrasekar, R., Doran, D., and Srinivas, B. (1996). Motivations and methods for text simplification. In *16th International Conference on Computational Linguistics*, pages 1041–1044.

Coster, W. and Kauchak, D. (2011a). Learning to simplify sentences using Wikipedia. In *Proceedings of the Workshop on Monolingual Text-To-Text Generation*, pages 1–9. Association for Computational Linguistics.

Coster, W. and Kauchak, D. (2011b). Simple English Wikipedia: a new text simplification task. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies. Portland, Oregon, USA: Association for Computational Linguistics*, pages 665–669.

De Belder, J., Deschacht, K., and Moens, M.-F. (2010). Lexical simplification. In *Proceedings of Itec2010 : 1st International Conference on Interdisciplinary Research on Technology, Education and Communication*.

Devlin, S. and Unthank, G. (2006). Helping aphasic people process online information. In *Proceedings of the 8th international ACM SIGACCESS conference on Computers and accessibility*, pages 225–226. ACM.

Drndarevic, B. and Saggion, H. (2012). Towards automatic lexical simplification in Spanish: an empirical study. In *Proceedings of the NAACL HLT 2012 Workshop Predicting and improving text readability for target reader populations (PITR 2012)*.

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

Hyönä, J. and Olson, R. (1995). Eye fixation patterns among dyslexic and normal readers: Effects of word length and word frequency. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, 21(6):1430.

Lal, P. and Ruger, S. (2002). Extract-based summarization with simplification. In *Proceedings of the ACL 2002 Automatic Summarization / DUC 2002 Workshop*.

Landis, J. and Koch, G. (1977). The measurement of observer agreement for categorical data. *Biometrics*, pages 159–174.

Miller, G., Beckwith, R., Fellbaum, C., Gross, D., and Miller, K. (1990). Introduction to WordNet: An on-line lexical database*. *International journal of lexicography*, 3(4):235–244.

Padró, L., Collado, M., Reese, S., Lloberes, M., and Castellón, I. (2010). FreeLing 2.1: Five years of open-source language processing tools. In *Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC'10)*, Valletta, Malta.

Rayner, K. and Duffy, S. A. (1986). Lexical complexity and fixation times in reading: Effects of word frequency, verb complexity, and lexical ambiguity. *Memory & Cognition*, 14(3):191–201.

Saggion, H., Gómez-Martínez, E., Etayo, E., Anula, A., and Bourg, L. (2011). Text Simplification in Simplext: Making Text More Accessible. *Revista de la Sociedad Española para el Procesamiento del Lenguaje Natural*, 47.

Salton, G., Wong, A., and Yang, C. (1975). A vector space model for automatic indexing. *Communications of the ACM*, 18(11):613–620.

Siddharthan, A. (2002). An architecture for a text simplification system. In *In LEC02: Proceedings of the Language Engineering Conference (LEC02*, pages 64–71.

Specia, L. (2010). Translating from complex to simplified sentences. In *Proceedings of the 9th international conference on Computational Processing of the Portuguese Language*, pages 30–39, Berlin, Heidelberg.

Specia, L., Jauhar, S., and Mihalcea, R. (2012). Semeval-2012 task 1: English lexical simplification. In *Proceedings of the 6th International Workshop on Semantic Evaluation (SemEval 2012)*.

Turney, P. and Pantel, P. (2010). From frequency to meaning: Vector space models of semantics. *Journal of Artificial Intelligence Research*, 37(1):141–188.

Vossen, P., editor (2004). *EuroWordNet: A Multilingual Database with Lexical Semantic Networks*, volume 17. Oxford Univ Press.

Woodsend, K., Feng, Y., and Lapata, M. (2010). Generation with quasi-synchronous grammar. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 513–523. Association for Computational Linguistics.

Woodsend, K. and Lapata, M. (2011a). Learning to simplify sentences with quasi-synchronous grammar and integer programming. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 409–420. Association for Computational Linguistics.

Woodsend, K. and Lapata, M. (2011b). WikiSimple: Automatic simplification of wikipedia articles. In *Proceedings of the Twenty-Fifth AAAI Conference on Artificial Intelligence (AAAI)*, pages 927–932.

Wubben, S., van den Bosch, A., and Krahmer, E. (2012). Sentence simplification by monolingual machine translation. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics*.

Yatskar, M., Pang, B., Danescu-Niculescu-Mizil, C., and Lee, L. (2010). For the sake of simplicity: Unsupervised extraction of lexical simplifications from wikipedia. In *Proceedings of the Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics*, pages 365–368.

Zhu, Z., Bernhard, D., and Gurevych, I. (2010). A monolingual tree-based translation model for sentence simplification. In *Proceedings of The 23rd International Conference on Computational Linguistics*, pages 1353–1361.

Zipf, G. (1935). *The Psychobiology of Language*. Houghton Mifflin, Boston, MA.

Identification of Social Acts in Dialogue

David B. Bracewell, *Marc T. Tomlinson*, *and Hui Wang*
Language Computer Corporation
Richardson, TX
{david,marc,hui}@languagecomputer.com

Abstract

The emergence of dialogue on social medial neccessitates the development of new dialogue processing models. We argue that to address coherence and to infer the implicatures of social dialogue it is vital to understand the social aspirations of the dialogue participants. One key aspect of understanding social dialogue is to understand the intentions and goals of participants. In this paper, we present 11 social acts that capture a broad number of social intentions and goals. We define social acts as pragmatic speech acts designed to give insight into the socio-cognitive processes that individuals unconsciously go through when communicating in dialogue. Identification of the social acts is done using a combination of a generative model in which utterances are generated from gappy patterns, which define a given social act, and a series of binary classifiers. Our experimentation shows that we can capture these social acts with an overall F-measure of 50.4%.

Keywords: dialogue, social actions, online communication, speech acts, social goals, social implicature.

Proceedings of COLING 2012: Technical Papers, pages 375–390,
COLING 2012, Mumbai, December 2012.

1 Introduction

Traditional approaches to dialogue processing have been primarily focused on task-based (Grosz, 1978; Traum and Hinkelman, 1992) dialogue. In addition, the recognition of speech acts has proved useful for identifying the structure of the dialogue which takes place in formal meetings (Shriberg et al., 2004) where the dialogue is often a function of the job or position of the participants. Theories of the coherence of discourse and discourse relations (Barzilay and Lapata, 2005; Byron and Stent, 1998; Hobbs, 1979, 1985; Mann and Thompson, 1988; Marcu and Echihabi, 2002) have proved useful for the semantic interpretation of discourse. However, in the world of Twitter, Facebook, and other social media where people voluntarily join in the conversation, dialogue is often focused on the social engagements between participants. These *social dialogues* are often not driven by a common group task, goal or purpose, but by the social aspirations of the participants. As an example let us examine the following excerpt of dialogue from a political debate forum:

> **A)** Seriously, how can anybody still support this president's economic policies? How can HE just continue to do the same things and why can't HE take responsibility for the lousy economy???
>
> **B)** Not to worry, last time we had a democrat president that was this bad, we had a huge victory!
>
> **C)** Being neither a Republican or Democrat I wonder if a Republican administration would have done any better? We know McCain's proposed tax and spending policies during the 2008 election would have led to even larger deficits than what Obama was proposing at the time. Not to mention the bailouts and TARP of Bush.
>
> **D)**What about the showing that when Obama took office, the economy was spiraling down with -9% real rate, losing 700,000+ jobs a month, skyrocketing unemployment, and the stock markets were crashing in the worst recession in 80 years. Not to mention the destruction of the housing market. Oh yeah and we were headed straight for a depression.

In the example shown above, the participants do not form a unified group working toward a common goal or task, but are instead splintered into subgroups which have their own agenda. The dialogue only progresses and stays coherent because the participants wish to further their own goals, e.g. further their bond with others in their subgroup, demonstrate their opposition to other groups, and influence the undecided. These social goals represent an individual's task in a task-oriented dialogue. Individuals construct a plan to accomplish their task and carry out their plan through social actions.

In order to address the coherence and to infer social implicatures from social dialogue it is vital to determine the social intentions and goals of the dialogue participants. The representation of social dialogue plays an important role in facilitating the inference of social intentions and goals. We believe the seminal work on attentions, intentions and the structure of discourse by Grosz and Sidner (1986) is best suited for the inference of social goals. A central component of this approach is the intentional structure, which takes into account the purpose of discourse segments. In social dialogue the purpose of a discourse segment is to further the social goal of the dialogue participants. Thus, by understanding the discourse segments, social goals can be inferred.

A straightforward approach to using Grosz and Sidner's theory for inferring social implicatures is to use the prevailing methods in dialogue processing. Topic modeling, such as Blei et al. (2003), can be used for identification of linguistic structure where topic shifts (Cassell et al., 2001) break the larger dialogue into dialogue segments. Dialogue acts (Allen and

Core, 1997; Stolcke et al., 1998; Bunt et al., 2010), which inform the intentions of dialogue, can be employed to infer the social goals of the participants. The attentional state can be captured using local coherence (Barzilay and Lapata, 2005; Byron and Stent, 1998). The social implicatures of the dialogue can then be inferred through the intentional structure, i.e. the social intentions and goals, and the attentional state, i.e. the focus of the dialogue and its participants.

However, our early experimentation revealed that this straightforward approach of using Grosz and Sidner's framework with prevailing dialogue processing techniques fails to capture the complexities of human social interactions and is incapable of reliably inferring the social implicatures of dialogue. The primary factor for this failure is that traditional dialogue acts are not capable of capturing the nuances of the social intentions and goals of the dialogue participants. Instead of focusing solely on the dialogue, we must also focus on the dialogue participants and how their social aspirations constrain their dialogue. Thus, we must look at the *social intentional structure* of dialogue which models the social intentions and goals of the dialogue participants and how the participants perceive the social intentions of others. The question then becomes: *How can the social intentional structure be captured?*

We propose to use *social acts* for inferring the social intentions and goals of dialogue participants which act to define the social intentional structure of the dialogue. Social acts capture the complex social actions individuals signal through their utterances. While most dialogue acts have some social overtones, they fail to adequately interpret the speaker's social goals. In contrast, the definitions of social acts are specifically designed to take into account participants' social cognition which constrains their dialogue facilitating the inference of social intentions and goals from their communication. We identify a set of 11 social acts, listed in section 3, that capture a variety of social goals and intentions. These social acts come from literature in the fields of psychology and organizational behavior and are motivated by work in dialogue understanding.

As with work in dialogue acts, we identify social acts at the utterance level. We employ a generative model for discovering gappy patterns which capture generalized cue phrases for each of the social acts. A gappy pattern consists of one or more words in between which there can exist gaps, or wildcards, which match any word. Each gap has an associated width which determines how many words the gap can match. The gappy patterns are used as features in a binary classifier. Each social act is associated with a classifier which determines if the social act is manifested or not in an utterance.

2 Related Work

Research understanding the intentionality of dialogue and dialogue has a long history. Some of the earliest work in dialogue processing is on speech acts. Speech acts are actions performed by individuals when making an utterance. Austin (1962) formalized the concept of speech acts by separating them into three classes: (1) *locutionary*, (2) *illocutionary*, and (3) *perlocutionary*. Much of the work in speech acts has been focused on illocutionary acts due to the work of Searle (1969).

Dialogue acts are specialized speech acts which include the internal structure, such as grounding and adjacency pairs, of a dialogue. There are a number of schemes for coding dialogue acts, such as DAMSL (Allen and Core, 1997), VERBMOBIL (Jekat et al., 1995), and DIT++ (Bunt et al., 2010). The DAMSL coding scheme defines dialogue acts that are

forward looking, which are extensions of speech acts, and which are backward looking, which relate the utterance to previous utterances. Frameworks like DIT++ have extended the typical coverage of dialogue acts to encompass a boarder set of acts, such as social obligations. However, when dialogue act schemes incorporate socially motivated acts often they do not fully take into account the multitude of purposes, social intentions, and ultimately the social implicatures of these acts. For example, in the statement "get me a cup of coffee", speech acts would focus on identifying the set of actions that would result from the utterance - presumably the target of the utterance physically going to get a cup of coffe for the speaker. In DIT++ the example utterance would most likely be labeled as "Instruct" which is void of any social implicatures resulting from the instruction. In contrast, social acts reflect the social intention of an utterance focusing on the social implicature of the statement, which in the case of the example utterance is that the speaker is indicating their power over the target.

A vast amount of research has been focused on the coherence of dialogue(Barzilay and Lapata, 2005; Byron and Stent, 1998; Hobbs, 1979; Mann and Thompson, 1988). Mann and Thompson Mann and Thompson (1988) introduce Rhetorical Structure Theory (RST), which was originally developed during the study of automatic text generation. They posit that the coherence of a text is attributed to the rhetorical relations between non-overlapping texts called the nucleus and satellite. The definition of the relations are not morphological or syntactic, but instead are focused on function and semantics. Discourse Representation Theory (DRT) (Kamp, 1984) provides a framework for the semantic understanding of discourse. DRT models the cognitive state of the reader, or hearer, of the discourse using discourse representation structures which convert the discourse into a logical form made up of referents and conditions. In social dialogue we must model and understand the speaker's cognitive state, which informs their social intentions and constrains their actions facilitating the progression and coherence of the dialogue.

Grosz and Sidner (1986) posit a structural approach to dialogue understanding where dialogue is broken into three constituents: *linguistic structure*, *intentional structure*, and *attentional state*. The linguistic structure encompasses how utterances combine together to form larger segments of dialogue. The intentional structure is defined using a single dialogue purpose and multiple dialogue segment purposes. The dialogue purpose is the overarching motivation for the dialogue. For social dialogue, the dialogue purpose should infer the social implicatures of the dialogue. Dialogue segment purposes are sub-components of the larger dialogue purpose and define the intention of a single dialogue segment. In social dialogue, one would expect the dialogue segment purpose to relate to the social intentions of the participants. The final structural component, the attentional state is a property of the dialogue and acts to keep track of the current focus of the dialogue. When dealing with social dialogue the attentional state is influenced by the participants and their social intentions.

The inference of social implicatures and identification of social goals through the use specialized social acts has been the focus of recent research. Bramsen et al. (2011) examined how individuals change their language usage depending on the status of the individual with whom they are communicating. In particular, they examine the use of *upspeak* and *downspeak* for signaling power relationships where *upspeak* is a sign that an individual is communicating with someone of higher status and *downspeak* is a sign that an individual is communicating with someone of lesser power. Danescu-Niculescu-Mizil et al. (2012) exam-

ined the use of coordination, often referred to as mimicry, for inferring power relationships. They showed that individuals are more likely to coordinate with individuals of higher status, i.e. those who have more power, than those with lower or equal status. Bracewell et al. (2011) examined a number of social acts for inferring whether two dialogue participants have a collegial relationship.

Other research has focused on the annotation of social acts. Tomlinson et al. (2012) examined the manifestation of a set of social acts in Arabic for inferring pursuits of power by participants. Bracewell et al. (2012) created an annotated corpus of *collegial* and *adversarial* social actions. Bender et al. (2011) created an annotated corpus of social acts relating to *authority claims* and *alignment moves* for determining authority and influence.

Related work is also found in the methods for identifying dialogue acts. Petukhova and Bunt (2011) examined using Bayes Net and Ripper for classification of high level dialogue acts in the DIT++ schema. They reported F-measures ranging from 62% to 95.1% for the AMI meeting corpus. Webb and Ferguson (2010) introduced an automated method for the extraction of cue phrases for identification of dialogue acts. They obtained an identification accuracy of almost 81% on the switchboard corpus when using cue phrases extracted from a portion of the switchboard corpus and an accuracy of almost 71% when using cue phrases extracted from the ICSI-MRDA corpus. Our approach combines the use of cue phrases and classification. We first extract cue phrases in the form of gappy patterns using a generative model. Then the cue phrases are used in a binary classifier which determines if an utterance is a manifestation of the associated social act.

3 Social Acts

Social interaction is one of the primary reasons for dialogue. Even predominantly task oriented dialogue (e.g. "let's go to the movies") has many possible social implications. The most apparent is the expression of a desire to establish or reaffirm a bond between the individuals. In order to reliably infer these social implications, the social intentions and goals for the dialogue participants must be taken into account.

We label the dialogue segment purpose, or the social intentions of an utterance, as social act. Social acts are pragmatic speech acts that signal a dialogue participant's social intentions. There are a number of social goals which a participant may have, including (1) maintaining an existing role, such as being an authority (Mayfield and Rose, 2011), in power (Bramsen et al., 2011), or collegial (Kim and Galstyan, 2010; Bracewell et al., 2011); (2) gaining a new role, such as by pursuing power (Tomlinson et al., 2012); or (3) maintaining or altering the role or status of another individual in the dialogue. Social acts can be signaled by a variety of cue phrases as well as through a dialogue participant's observation or violation of social norms, or expectations of socially appropriate responses.

The set of acts presented in this section have been derived from work in psychology on power, status, and leadership (Anderson et al., 2001; French and Raven, 1959; Keltner et al., 2008; Owens and Sutton, 2001; Smith and Galinsky, 2010), as well as on conflict and cooperation (Brewer and Gardner, 1996; Deutsch, 2011; Jehn and Mannix, 2001). This set of social acts was designed to have broad coverage, but is not to be taken as an exclusive set. Figure 2 gives an example of a dialogue marked with the social acts.

Agreement can act as an affordance to an individual or as a means to establish solidarity between individuals. Likewise disagreement can act as a way of undermining or challenging

Agreement	Statements that a group member makes to indicate that he/she shares the same view about something another member has said or done.
Challenge Credibility	Attempts to discredit or raise doubt about another group member's qualifications or abilities.
Disagreement	Statements a group member makes to indicate that he/she does not share the same view about something another member has said or done.
Disrespect	Inappropriate statements that a group member makes to insult another member of the group.
Establish Credibility	Statements that a speaker makes to demonstrate his/her knowledge or personal experience in order to make him/herself look better in the eyes of the group.
Managerial Influence	Statements that a speaker makes to control the discussion with the goal of increasing sway over the group.
Mediation	Attempts made by a group member to resolve a conflict occurring between other group members.
Relationship Conflict	Personal, heated disagreement between individuals.
Solidarity	Statements that a group member makes to strengthen the group's sense of community and unity.
Supportive Behavior	Statements of personal support that one group member makes toward another.
Task Conflict	Disagreement over the manner in which a task is performed or over the outcome of the task.

Figure 1: The set of eleven social acts.

credibility. However, because of the special status of agreement and disagreement we consider them as two separate social acts.

Agreement can be manifested through simple phrases, such as "I agree", through negations of disagreement, such as "I am not disagreeing with you", and through more complex phrases, such as "What Adam says is in principle correct." Similarly, disagreement is manifested through simple "I disagree" phrases as well as negations of agreement, such as "I definitely do not agree with what you said."

Challenging credibility can be used by an individual to lower the status of other group members (Owens and Sutton, 2001). These challenges can be in demands to prove credibility, such as "prove your lies" and aggressive accusing questions, such as "what does that have to do with what we are talking about?". Challenging credibility can also occur through gossip, such as "X doesn't know what he is talking about". This tactic can be used by group members to moderate the power of a leader who has overstepped their boundaries (Keltner et al., 2008).

Disrespected individuals often feel they have been unjustly treated due to the disrespectful action, causing a social imbalance between them and the perpetrator (Miller, 2001). This social imbalance causes a power differential between the two individuals, thus giving the perpetrator power over the individual. Examples of disrespect include "You are a gigantic hypocrite you know that?" and "Do you speak English well?"

Establishing credibility reflects an attempt by an individual to demonstrate their credibility and fitness for leadership (Keltner et al., 2008). Evidence for establishment of credibility manifests itself in many different ways. The most common in our data set is an explicit mention of the individual's credentials, such as "I am a physicist". Alternatively a person can demonstrate their credibility by providing the group with cited information, such as

A) Propose that this page be moved to East Timor Defence Force as this is the closest translation of Forças de Defesa de Timor Leste [*Managerial Influence*]. **I have worked in Timor Leste as a government advisor, including with FDTL, and have never heard anybody ever refer to the FDTL as Military of East Timor** [*Establish Credibility*].

B) As I understand it, 'East Timor Defence Force' is considered outdated [*Managerial Influence*]. **While it was commonly used when the force was established, almost all english-language publications now use 'F-FDTL'.** [*Managerial Influence*] 'Military of East Timor' is a generic name, and **I agree that it's rarely used and not a great title.** [*Agreement*] **I'd prefer 'Timor Leste Defence Force' as this seems to be the direct translation, but this would be inconsistent with the other Wikipedia articles on the country.** [*Managerial Influence*] **Should we be bold and move this article to 'Timor Leste Defence Force'?** [*Establish Solidarity*]

A) I so totally agree with you. [*Agreement*] **'Timor Leste Defence Force' is it.** [*Agreement*] **The only reason I did not propose that was the failure to change the country page from East Timor to Timor Leste, a decision that I feel was extremely discourteous of Wikipedia considering the government's specific request that it be referred to as Timor Leste.** [*Managerial Influence*] **If you have worked there you will know that everybody uses 'Timor Leste', even the ADF but the Australian DFAT uses East Timor although the more enlightened Kiwi embassy call it TL.** [*Establish Credibility*] **I suggest we leave it for 48 hours to see if anyone has any strong feelings and then change it to ' Timor Leste Defence Force' with diverts from F-FDTL and FDTL.** [*Managerial Influence*]

Figure 2: Social acts tagged for an excerpt of a discussion taken from a Wikipedia Talk page.

"Article 10.5 paragraph 3 says...". Finally an individual can justify their opinion through the use of logic or citation of personally relevant anecdotes.

Managerial influence is used by individuals to signal that they are a leader. Examples of managerial influence include "Can we focus the discussion" and "Are we still trying to find out where the scholarly consensus is on the matter of Lukan authorship?" Figure 2 has a number of examples of managerial influence, such as A proposing to move the page and B giving factual reasons why "Military of East Timor" is an incorrect name for the page.

A person in power often acts as a mediator for disputes between other group members. Mediation itself is an attempt to resolve a conflict occurring between other group members. Individuals performing mediation may already be in position of power. Examples of mediation include "You really need to back off and take a deep breath" and "Let's just all keep calm, yes?"

Relationship conflict is a personal, heated disagreement between individuals (Jehn and Mannix, 2001). Individuals exhibiting relationship conflict are being adversarial. Examples of relationship conflict include "your arrogant blathering" and "I consider it offensive for you to assert that i insist on turning every interaction into a personality conflict.'

Further, language indicative of a desire for group solidarity encapsulates the establishment and maintenance of shared group membership. Group membership can be expressed at either the relational level (e.g. Father, co-worker, etc.) or the collective level (e.g. single mothers) (Brewer and Gardner, 1996). Language indicative of a desire for group solidarity demonstrates that an individual identifies with the group, an important characteristic of leaders (Keltner et al., 2008) and cooperators (Deutsch, 2011). This solidarity can be expressed explicitly (e.g. "We're all in this together"), covertly (e.g. as through the use of inclusive first-person pronouns), or through unconscious actions and linguistic cues, such as the use of in-group jargon, certain syntactic constructions, and mimicry.

Supportive behavior, or cooperation towards a common goal, is an example of collegiality. This type of behavior lies at the center of group dynamics. Cooperation is correlated with both overall group performance and managerial ratings of group effectiveness (Campion et al., 1996). Evidence for cooperation manifests itself in many different ways. Classically, there is the notion of cooperation on a physical task (e.g. one person helping another lift a heavy weight), or cooperation through social support (e.g. Mary says, "John's decision is excellent").

There are also more subtle, unconscious examples of cooperation between individuals, which can demonstrate a certain degree of collegiality between the individuals. One example is cooperation for the effective use of language and the building of dialogue (Garrod and Pickering, 2004). Dialogue is a complicated interaction that requires commitment from both parties. In order to maintain a stable conversation, participants must be willing to expend cognitive effort to listen, understand, and form a relevant response that advances the dialogue. The degree to which participants are able to maintain a cohesive dialogue should be reflected in the collegiality of the participants. If one participant is not cooperating, the dialogue will not progress.

Task conflict often arises during power struggles in a group where one individual is attempting to overtake the position of another (Jehn and Mannix, 2001). It is defined as disagreement over the manner in which a task is performed or over the outcome of the task. Task conflict can be manifested by actions performed to undo or challenge other's work toward a task, such as "I reverted your all your changes." Additionally, it may manifest itself as taking sides or stating positions around a conflict, such as "So yes, I will not be editing but I will be monitoring to see if some other naive soul wishes to and try to support them (and revert the vandalism that happens from time to time)."

4 A Generative Model for Identifying Social Acts

A system for the recognition of intentions and goals is the foundation for the understanding of social implicatures in social dialogue. Here we present a generative model for identifying social acts at the utterance level. The model generates utterances exhibiting a social act as a series of gappy patterns. A gappy pattern consists of one or more words in between which there can be gaps, or wildcards, which match any contiguous sequence of non-whitespace characters. Associated with the gap is a width, which determines how many words the gap can match. An example of a gappy pattern with a gap width of one extracted for a three utterances is shown in Figure 3.

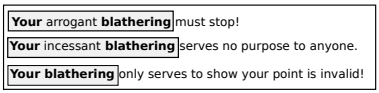


Figure 3: An example of a gappy pattern consisting of a single gap which can capture zero or one words and three utterances for which the pattern matches.

The generative model we employ is a modified version of the model introduced by Gimpel and Smith (2011) for machine translation. The main difference is that our model is supervised whereas Gimpel and Smith's model is unsupervised. The supervision in our

model helps to guide the generative process in discovering gappy patterns related to a specific social act.

We assume that we are given a set of utterances $u_{1:k}$ where each utterance is made up of n-words, $w_{1:n}$, and a set of labels $l_{1:k}$ such that label l_i is associated with utterance u_i. Following the terminology of Gimpel and Smith (2011), gappy patterns are represented over the words in the utterances as a color where each word in the utterance has a color assignment, i.e. there is a vector of color assignments $c_{i:n}$ for each utterance. A color C_j is a set made up of the c_i color-word associations such that $C_j = i : c_i = j$. A pattern is built from each color C_j by concatenating the words assigned to the color from left to write inserting gaps between non-adjacent words.

The generative story for a single utterance entails sampling the following:

1. The number of words, n, in an utterance as a $Poisson$ distribution with parameter β.
2. The number of unique colors in an utterance given a $Uniform$ distribution.
3. The color c_i for each word w_i in the utterance as a $Uniform$ distribution.
4. The probability of the pattern associated with each color C_j for utterances with label l as a $Multinominal$ with parameter μ.

Thus, to generate patterns we must calculate the probability of generating an utterance of length n, with m colors, label l, and color assignments $c_{1:n}$ as:

$$p(w_{1:n}, c_{1:n}, m|\beta, \mu) = \frac{1}{Z}(\frac{\beta^n}{n!}e^{-\beta})(\frac{1}{n})(\frac{1}{m})^n \prod_{j=1}^{m} p_\mu(\pi(C_j) \mid l)$$

where Z is a normalization factor.

The multinomial distribution, p_μ, is modeled using a Dirichlet process. A Dirichlet process can be treated as a probability distribution over random distributions which facilitates an unbounded set of parameters $\mu \sim DP(\alpha, P_0)$, where α is the concentration parameter and P_0 is the base distribution. The base distribution is made up of: (1) a $Poisson$ distribution with parameter υ over the number of words in the utterance; (2) a uniform distribution for each word; (3) a uniform distribution over the number of gaps given the number of words; and (4) a uniform distribution over the arrangement of gaps given the number of gaps and words.

Gibbs sampling is used to sample the posterior distribution $p(\{c^{(i)}, m^{(i)}\}_{i=1}^{U} \mid \{w^{(i)}\}_{i=1}^{U}, \upsilon, \alpha)$, where U is the total number of utterances. Gibbs sampling is a form of Markov Chain Monte Carlo (MCMC), which is used to obtain a sequence of random samples from a joint probability distribution of two or more random variables. In particular, Gibbs sampling is used when direct sampling is prohibitive. The Gibbs sampler makes repeated iterations. During each of the iterations it samples a new color for each of the color assignments (c_i). A new color is assigned to c_i by first removing the current color and then choosing from either one the other m colors in the utterance or a creating a new color. The probability of choosing a new color is proportional to:

$$\frac{N_{\pi(\{i\})} + \alpha P_0(\pi(\{i\}) \mid l)}{N + \alpha}$$

where $N_{\pi(\{i\})}$ is the count of pattern π over all utterances with label l and N is the total count of all the patterns. The probability of assigning an existing color j to c_i is proportional to:

$$\frac{N_{\pi(C_j \cup \{i\})} + \alpha P_0(\pi(C_j \cup \{i\}) \mid l)}{N_{\pi(C_j)} + \alpha P_0(\pi(C_j) \mid l)}$$

where $C_j \cup \{i\}$ states that c_i is being added to C_j.

After discovering gappy patterns using the described generative model, we build a binary classifier for each of the social acts we wish to identify. In particular, we use a logistic regression classifier to the learn a set of weights for each of the gappy patterns, which denotes the discriminatory ability of the pattern in identifying the social act. A social act is manifested in an utterance when $H(z) = 1$, where $H(z)$ is calculated as:

$$H(z) = \begin{cases} 1, & \frac{1}{1+e^{-z}} > 0.5 \\ 0, & \frac{1}{1+e^{-z}} \leq 0.5 \end{cases}$$

where $z = \sum_{i=1}^{m} w_i * \phi(\pi_i, w_{i:n})$ and ϕ returns 1 if pattern π_i is present in the utterance made up of words $w_{1:n}$. An utterance is then assigned all social acts whose accompanying classifier results in an $H(z) = 1$.

5 Data Collection & Annotation

We constructed a corpus of 215 social dialogues extracted from English Wikipedia talk pages, public forums, and chat transcripts. A total of 21,067 utterances were extracted from the social dialogues. On average each utterance contained 18.7 words.

A web-based interface was constructed for annotation. The interface listed for a single social dialogue all of the utterances in the order in which they appeared in the dialogue along with speaker information. Social acts were annotated by through the use a drop-down list and allowed for an arbitrary number of social acts to be assigned to an utterance.

Each utterance was annotated by two annotators, who were trained linguists, as either being a manifestation or not of one or more of the eleven social acts described in section 3. In total, 8,149 (38.7%) of the total utterances had at least one of the eleven social acts annotated. On average each utterance was assigned with 0.98 social acts.

We first looked at the inter-annotator agreement for if an utterance exhibited any social act or not. The results are listed in Table 1. The micro-averaged mutual F-Measure was 94.0% which broke down as 84.0% F-Measure for "exhibited" and a 94.0% F-Measure for "Not Exhibited." The results show that expert annotators can reliably determine the presence or absence of social actions in utterances.

Next, we examined the inter-annotator agreement rate for each of the individual social acts. Table 2 shows the number of utterances annotated for each social act, the Cohen's kappa (Cohen, 1960), and the mutual F-measure.

As seen in table 2, the kappa values range from 0.13 to 0.53. In contrast, the kappa values for dialogue acts have been reported as high as 0.76 for ANSWER and as low as 0.15 for COMMITTING-SPEAKER-FUTURE-ACTION (Allen and Core, 1997). More recent work in dialogue act annotation has been performed by Geertzen and Bunt (2006) who report kappa

	F-Measure
Exhibited	84.0%
Not Exhibited	94.0%
micro-Averaged	94.0%
macro-Averaged	89.0%

Table 1: The mutual F-Measure for an utterance exhibiting or not exhibiting any social act.

	# Annotated	Kappa	F-Measure
Agreement	295	0.38	0.76
Challenge Credibility	1,113	0.36	0.33
Disagreement	434	0.46	0.71
Disrespect	367	0.24	0.54
Establish Credibility	364	0.53	0.45
Managerial Influence	2,486	0.23	0.16
Mediation	167	0.26	0.52
Relationship Conflict	399	0.13	0.21
Solidarity	100	0.52	0.42
Supportive Behavior	269	0.36	0.68
Task Conflict	2,802	0.35	0.31

Table 2: The number of annotations and kappa per social act.

values between 0.21 and 1.0 for the top level of a hierarchical dialogue act scheme (Bunt et al., 2010). However, they only calculated kappa for utterances in which both annotators had assigned a dialogue act, i.e. utterances where only one annotator assigned a dialogue act were ignored. In contrast, we calculated our kappa values for all utterances where at least one annotator assigned a social act.

Other work in social acts have seen kappa values in a similar range, such as the Bender et al. (2011) who report kappa values from 0.13 to 0.63. Given the complexities presented by annotating the social intentions of dialogue participants, we believe that the kappa values reported here are acceptable given the accompanying F-Measures.

6 Experimental Results

The utterances in the corpus were labeled with a social act if it was assigned by either of the annotators. The reason for this was two-fold: (1) The definitions of the social acts can be interpreted differently depending on internal thresholding (e.g. how hostile does a remark need to be in order to be classified as a Relationship Conflict?) and no interpretation is truly incorrect. (2) Given the sparsity of annotation for some of the social acts (e.g. Solidarity only having 100 total annotations) made it necessary to include any instance. Experiments were then performed using a standard 80/10/10 split where 80% of the data was used for training, 10% for development, and 10% for testing. We examined an n-gram based approach for comparison to the gappy patterns.

For the gappy pattern method, we constrained the gap width to 2, meaning that the maximum number of words a gap can consume was two. In addition, the sampling process was ran for 1,000 iterations with a burn-in of 50 iterations. A minimum probability of

70% was needed to initiate a new color and a minimum probability of 40% was needed to propagate a color. These parameters were tuned using the development set.

The N-gram based method is a simplification of the gappy pattern method, i.e. it is patterns without gaps. First, n-grams were extracted from the annotated data. Second, the n-grams were pruned using information gain where the exact number of features retained was determined using the development set. Finally, the remaining n-grams were used as binary features in an Support Vector Machine (SVM) classifier with a linear kernel. SVMs are frequently used in text classification and have been shown to give good results for dialogue acts (Hu et al., 2009). We examined using unigrams (1-gram) and unigrams and bigrams (1+2-grams). Other size bigrams were tried as well as incorporating part-of-speech, but resulted in extremely poor performance.

	Precision	Recall	F-Measure
Gappy	49.5%	51.3%	50.4%
1-gram	43.8%	43.2%	43.5%
1+2-grams	38.4%	37.8%	38.1%

Table 3: Micro averaged precision, recall, and F-measure for identifying the 11 social acts using an 80/20 split.

Table 3 lists the micro-averaged precision, recall, and F-measure for the identification of the 11 social acts. As can be seen in the table, the gappy-pattern based approach had an increase of 6.9% in F-Measure over the best n-gram approach (1-gram). This included an increase in precision of 5.7% and in recall of 8.1%. The unigram and bigram (1+2-grams) based method performed the worst, mostly due to the size of data. In contrast, the gappy pattern approach was able to learn a mix of patterns of varying lengths and gaps (up to 2) that were able to separate the social acts. Figure 4 shows example patterns for each of the social acts that were discovered by the generative model and then weighted as a positive indicator by logistic regression.

Conclusion

In this work we have begun a process of revisiting classical theories of dialogue coherence and understanding. Our initial efforts have been to show how the social intentions behind dialogue segments can be understood. Understanding the social goals of individuals is critical to properly parsing dialogue in a modern age of social media. We introduced a set of social acts designed to capture the social intentions of dialogue participants. Our results show that social acts can be reliably understood by annotators and that a novel method for detecting speech acts, based on a generative method for identifying gappy patterns, can achieve results consistent with work in recognizing dialogue acts.

This work creates a foundation for building models of the higher level intentional structure of social dialogue and attention. The intentional structure around the social acts could provide valuable insight into the social goals of a dialogue and the dialogue participants. It is also critical that future work addresses the way in which social attention is modulated within a coherent dialogue.

Agreement	Supportive Behavior	Challenge Credibility
exactly	support	tring bullshit
maybe	works beautifully	been spite
certainly	woman support	reverting move
fair	appreciate	fixing
Disagreement	**Disrespect**	**Relationship Conflict**
violate	simply cowardice	insult <gap> against
your <gap> unacceptable	fooling <gap> fallacy	simple
misinformative	hahaha	amazingly <gap> decision
not agree	quaking	arrogant blathering
Mediation	**Task Conflict**	**Establish Credibility**
insulting each	threshold	myself
false accusations	article title	difference between
former <gap> violates	sources	lead <gap> reflect
should resolve	rebutting reliable	giving <gap> revert
Managerial Influence	**Solidarity**	
discussion	definitely improvement	
also <gap> problematic	good point	
information should	improving thanks	
contentious <gap> problem	reliable	

Figure 4: Example patterns for each social act that were discovered using the generative model and then determined to be a positive indicator by logistic regression.

Acknowledgment

This research was funded by the Office of the Director of National Intelligence (ODNI), Intelligence Advanced Research Projects Activity (IARPA), and through the U.S. Army Research Lab. All statements of fact, opinion or conclusions contained herein are those of the authors and should not be construed as representing the official views or policies of IARPA, the ODNI or the U.S. Government.

References

Allen, J. and Core, M. (1997). Draft of DAMSL: Dialog Act Markup in Several Layers.

Anderson, C., John, O. P., Keltner, D., and Kring, A. M. (2001). Who attains social status? Effects of personality and physical attractiveness in social groups. *Journal of Personality and Social Psychology*, 81(1):116–132.

Austin, J. L. (1962). *How to do things with words*, volume 7 of *The William James lectures*. Harvard University Press.

Barzilay, R. and Lapata, M. (2005). Modeling local coherence: an entity-based approach. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, ACL '05, pages 141–148, Stroudsburg, PA, USA. Association for Computational Linguistics.

Bender, E. M., Morgan, J. T., Oxley, M., Zachry, M., Hutchinson, B., Marin, A., Zhang, B., and Ostendorf, M. (2011). Annotating social acts: authority claims and alignment moves in wikipedia talk pages. In *Proceedings of the Workshop on Languages in Social Media*, LSM '11, pages 48–57, Stroudsburg, PA, USA. Association for Computational Linguistics.

Blei, D. M., Ng, A., and Jordan, M. (2003). Latent dirichlet allocation. *JMLR*, 3:993–1022.

Bracewell, D., Tomlinson, M., Brunson, M., Plymale, J., Bracewell, J., and Boerger, D. (2012). Annotation of adversarial and collegial social actions in discourse. In *Proceedings*

of the Sixth Linguistic Annotation Workshop, pages 184–192, Jeju, Republic of Korea. Association for Computational Linguistics.

Bracewell, D. B., Tomlinson, M., Shi, Y., Bensley, J., and Draper, M. (2011). Who's playing well with others: Determining collegiality in text. In *Proceedings of the 5th IEEE International Conference on Semantic Computing (ICSC 2011)*, Palo Alto. IEEE Computer Society.

Bramsen, P., Escobar-Molano, M., Patel, A., and Alonso, R. (2011). Extracting social power relationships from natural language. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1*, HLT '11, pages 773–782, Stroudsburg, PA, USA. Association for Computational Linguistics.

Brewer, M. B. and Gardner, W. (1996). Who is this "We"? Levels of collective identity and self representations. *Journal of Personality and Social Psychology*, 71(1):83–93.

Bunt, H., Alexandersson, J., Carletta, J., Choe, J.-W., Fang, A. C., Hasida, K., Lee, K., Petukhova, V., Popescu-Belis, A., Romary, L., Soria, C., and Traum, D. R. (2010). Towards an iso standard for dialogue act annotation. In *LREC*.

Byron, D. and Stent, A. (1998). A preliminary model of centering in dialog. In *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics - Volume 2*, ACL '98, pages 1475–1477, Stroudsburg, PA, USA. Association for Computational Linguistics.

Campion, M., Papper, E., and Medsker, G. (1996). Relations between work team characteristics and effectiveness: A replication and extension. *Personnel psychology*, 49(2):429–452.

Cassell, J., Nakano, Y. I., Bickmore, T. W., Sidner, C. L., and Rich, C. (2001). Non-verbal cues for discourse structure. In *Proceedings of the 39th Annual Meeting on Association for Computational Linguistics*, ACL '01, pages 114–123, Stroudsburg, PA, USA. Association for Computational Linguistics.

Cohen, J. (1960). A Coefficient of Agreement for Nominal Scales. *Educational and Psychological Measurement*, 20(1):37–46.

Danescu-Niculescu-Mizil, C., Lee, L., Pang, B., and Kleinberg, J. (2012). Echoes of power: language effects and power differences in social interaction. In *Proceedings of the 21st international conference on World Wide Web*, WWW '12, pages 699–708, New York, NY, USA. ACM.

Deutsch, M. (2011). Cooperation and competition. *Conflict, Interdependence, and Justice*, pages 23–40.

French, J. R. P. and Raven, B. (1959). The Bases of Social Power. In Cartwright, D., editor, *Studies in Social Power*, volume 35 of *Studies in social power*, chapter 9, pages 150–167. Institute for social research.

Garrod, S. and Pickering, M. J. (2004). Why is conversation so easy? *Trends in Cognitive Sciences*, 8(1):8–11.

Geertzen, J. and Bunt, H. (2006). Measuring annotator agreement in a complex hierarchical dialogue act annotation scheme. In *Proceedings of the 7th SIGdial Workshop on Discourse and Dialogue*, SigDIAL '06, pages 126–133, Stroudsburg, PA, USA. Association for Computational Linguistics.

Gimpel, K. and Smith, N. A. (2011). Generative Models of Monolingual and Bilingual Gappy Patterns. *Proceedings of the Sixth Workshop on Statistical Machine Translation*, (2009):512–522.

Grosz, B. J. (1978). *Understanding Spoken Language*, chapter Discourse Analysis. Elsevier Science.

Grosz, B. J. and Sidner, C. L. (1986). Attention, Intention, and the Structure of Discourse. *Computational Linguistics*, 12(3):175–204.

Hobbs, J. R. (1979). Coherence and coreference *. *Cognitive Science*, 3(1):90–67.

Hobbs, J. R. (1985). On the Coherence and Structure of Discourse. In *CSLI 85-37*.

Hu, J., Passonneau, R. J., and Rambow, O. (2009). Contrasting the interaction structure of an email and a telephone corpus: a machine learning approach to annotation of dialogue function units. In *Proceedings of the SIGDIAL 2009 Conference: The 10th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, SIGDIAL '09, pages 357–366, Stroudsburg, PA, USA. Association for Computational Linguistics.

Jehn, K. A. and Mannix, E. A. (2001). The Dynamic Nature of Conflict: A Longitudinal Study of Intragroup Conflict and Group Performance. *Academy of Management Journal*, 44(2):238.

Jekat, S., Klein, R., Maier, E., Maleck, I., Mast, M., Berlin, T., Quantz, J. J., and Quantz, J. J. (1995). Dialogue acts in verbmobil. Technical report.

Kamp, H. (1984). A theory of truth and semantic representation. In Groenendijk, J., Janssen, T. M. V., and Stokhof, M., editors, *Truth, Interpretation and Information: Selected Papers from the Third Amsterdam Colloquium*, pages 1–41. Foris Publications, Dordrecht.

Keltner, D., Van Kleef, G. A., Chen, S., and Kraus, M. W. (2008). A reciprocal influence model of social power: Emerging principles and lines of inquiry. *Advances in experimental social psychology*, 40:151–192.

Kim, J. and Galstyan, A. (2010). Towards modeling social and content dynamics in discussion forums. In *Proceedings of the NAACL HLT 2010 Workshop on Computational Linguistics in a World of Social Media*, WSA '10, pages 13–14, Stroudsburg, PA, USA. Association for Computational Linguistics.

Mann, W. C. and Thompson, S. A. (1988). Rhetorical Structure Theory: Toward a Functional Theory of Text Organization. *Text*, 8(3):243–281.

Marcu, D. and Echihabi, A. (2002). An unsupervised approach to recognizing discourse relations. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, ACL '02, pages 368–375, Stroudsburg, PA, USA. Association for Computational Linguistics.

Mayfield, E. and Rose, C. P. (2011). Recognizing Authority in Dialogue with an Integer Linear Programming Constrained Model. In *Computational Linguistics*, pages 1018–1026. Association for Computational Linguistics.

Owens, D. and Sutton, R. (2001). Status contests in meetings: Negotiating the informal order. *Groups at work: Theory and research*, 14:299–316.

Petukhova, V. and Bunt, H. (2011). Incremental dialogue act understanding. In *Proceedings of the Ninth International Conference on Computational Semantics*, IWCS '11, pages 235–244, Stroudsburg, PA, USA. Association for Computational Linguistics.

Searle, J. R. (1969). *Speech Acts: An Essay in the Philosophy of Language*. Cambridge University Press.

Shriberg, E., Dhillon, R., Bhagat, S., Ang, J., and Carvey, H. (2004). The ICSI Meeting Recorder Dialog Act (MRDA) Corpus. In Strube, M. and Sidner, C., editors, *Proceedings of the 5th SIGdial Workshop on Discourse and Dialogue*, pages 97–100, Cambridge, Massachusetts, USA. Association for Computational Linguistics.

Smith, P. and Galinsky, A. (2010). The nonconscious nature of power: Cues and consequences. *Social and Personality Psychology Compass*, 4(10):918–938.

Stolcke, A., Shriberg, E., Bates, R., Coccaro, N., Jurafsky, D., Martin, R., Meteer, M., Ries, K., Taylor, P., and Van Ess-Dykema, C. (1998). Dialog Act Modeling for Conversational Speech. In *Applying Machine Learning to Discourse Processing*, pages 98–105. AAAI Press.

Tomlinson, M., Bracewell, D. B., Draper, M., Almissour, Z., Shi, Y., and Bensley, J. (2012). Pursing power in arabic on-line discussion forums. In *Proceedings of the Eighth Conference on International Language Resources and Evaluation*.

Traum, D. R. and Hinkelman, E. A. (1992). Conversation acts in task-oriented spoken dialogue. *Computational Intelligence*, 8(3):575–599.

Webb, N. and Ferguson, M. (2010). Automatic extraction of cue phrases for cross-corpus dialogue act classification. In *Proceedings of the 23rd International Conference on Computational Linguistics: Posters*, COLING '10, pages 1310–1317, Stroudsburg, PA, USA. Association for Computational Linguistics.

Robust, lexicalized native language identification

Julian BROOKE *Graeme HIRST*
University of Toronto, Department of Computer Science, Toronto, Canada
jbrooke@cs.toronto.edu, gh@cs.toronto.edu

ABSTRACT
Previous approaches to the task of native language identification (Koppel et al., 2005) have been limited to small, within-corpus evaluations. Because these are restrictive and unreliable, we apply cross-corpus evaluation to the task. We demonstrate the efficacy of lexical features, which had previously been avoided due to the within-corpus topic confounds, and provide a detailed evaluation of various options, including a simple bias adaptation technique and a number of classifier algorithms. Using a new web corpus as a training set, we reach high classification accuracy for a 7-language task, performance which is robust across two independent test sets. Although we show that even higher accuracy is possible using cross-validation, we present strong evidence calling into question the validity of cross-validation evaluation using the standard dataset.

KEYWORDS: Native language identification, text classification, evaluation.

Proceedings of COLING 2012: Technical Papers, pages 391–408,
COLING 2012, Mumbai, December 2012.

1 Introduction

Native language identification (Koppel et al., 2005) is a task in which features of the second language (L2) texts written by non-native speakers of various different native language (L1) backgrounds are used to identify those backgrounds. One potential application is as a facet of author profiling, which can be used to identify those who misrepresent themselves online (Fette et al., 2007). Another is as a preprocessing step to language learner error correction (Leacock et al., 2010): for example, Rozovskaya and Roth (2011) use L1-specific information to improve their preposition-correction system, and recent work in collocation correction relies on the specific forms present in the writer's native language (Chang et al., 2008; Dahlmeier and Ng, 2011).

As a distinct task in computational linguistics, native language identification has been reasonably well-addressed (Koppel et al., 2005; Tsur and Rappoport, 2007; Wong and Dras, 2009), and in fact there has been a flurry of recent activity (Kochmar, 2011; Golcher and Reznicek, 2011; Wong and Dras, 2011; Brooke and Hirst, 2011; Wong et al., 2012). Though a wide range of feature types has been explored—with conflicting results—the evaluation of these feature sets has been fairly uniform: training and testing in one of several small corpora of learner essays (Granger et al., 2009; Yannakoudakis et al., 2011; Lüdeling et al., 2008), which are unfortunately quite expensive to collect. A notable problem with these corpora with respect to native language identification, however, is a clear interaction between native language and essay topic. Generally speaking, the solution in previous work has been to avoid the use of lexical features that might carry topical information, limiting feature sets to syntactic and phonological phenomena. There are two reasons to be critical of this approach. First, there are almost certainly kinds of language transfer (Odlin, 1989), i.e. transfer related to lexical choice, that are being overlooked. Second, and more troubling, is that avoiding the lexicon is not fully effective as a means of countering the effects of topic: some recent work indicates that variation in topic also has significant influence on non-lexical features (Golcher and Reznicek, 2011; Brooke and Hirst, 2011), calling into question the reliability of previous results that assume otherwise.

The approach we present here resolves this tension by requiring training and test sets that are independently sampled. Although corpora may have some form of confounding variation that may artificially inflate or (in some cases) lower performance relative to other samples from the same corpus, any variation that is consistent across very distinct corpora is likely to be a true indicator of L1. Although we test on the typical essay corpora used by other researchers, we train on a very different dataset, a large but messy corpus of journal entries scraped from a language learner website. Without the distraction of (irrelevant) topic bias, we can test the efficacy of lexical features, including *n*-grams and dependencies. We also test a number of options at the level of the classifier, most notably a multiclass support vector machine (SVM) decision-tree classifier that leverages the genetic relationships among languages, and a simple but elegant method for adapting an SVM classifier to the test corpus without integrating the confounding variation found there. Our best classifier with lexical and syntactic features provides results that compare well with previously-reported single-corpus performance; we also present, however, evidence that calls into question the validity of these previous results, showing that topic bias within the corpus is having a major effect and that indeed the performance of models built in the topic-biased ICLE corpus is not robust, regardless of the features chosen.

2 Related Work

The earliest focused work on native language detection was by Koppel et al. (2005). They classified texts from the International Corpus of Learner English (ICLE) into one of five (European) native language backgrounds using support vector machines. They described their feature set as stylistic; features included the frequency of function words, rare POS bigrams, letter *n*-grams, and spelling errors. They reported a performance of just over 80% on the task using the full feature set.

Other work on the ICLE includes that of Tsur and Rappoport (2007), who were concerned with identifying phonological language transfer; they focused on the construction of character *n*-gram models, reporting 66% accuracy with just these sub-word features, with only a small drop in performance when the dominant topic words in each sub-corpus (as identified using *tf-idf*) were removed. Wong and Dras (2009) investigated particular types of syntactic error: subject-verb disagreement, noun-number disagreement, and determiner problems, relating the appearance of these errors to the features of relevant L1s. However, they reported that these features do not help with classification, and they also note that character *n*-grams, though effective on their own, are not particularly useful in combination with other features. In follow-up work, Wong and Dras (2011) attained the best results to date, 80% performance on a 7-language task, by including syntactic production rules. Recent work by Wong et al. (2012) and Swanson and Charniak (2012) has explored the use of statistical grammatical induction techniques—Adaptor Grammars in the former case, Tree Substitution Grammars in the latter—to select better syntatic features for classification.

The work of Kochmar (2011) is distinct from those above in a number of ways: she used a different corpus of essays, derived from the Cambridge Learner Corpus[1], and concentrated on pairwise (SVM) classification within two European language sub-families. An exhaustive feature analysis indicated that character *n*-gram frequency is the most useful feature type for her task; unlike Wong and Dras (2011), syntactic production rules provided little benefit. With respect to lexical features, Kochmar presented some results using word *n*-grams, but regarded them as attributable to topic bias in the corpus. Error-type features (e.g. spelling, missing determiner) as provided by the corpus annotation offered little improvement over the high performance offered by the distributional features (e.g. POS/character *n*-grams).

Golcher and Reznicek (2011) used a string-distance metric to identify the native language of German learners in the Falko corpus (Lüdeling et al., 2008), and contrasted this with a topic classification task in the same corpus. Even after taking steps to mitigate topic bias (removing the influence of the words in the title), the usefulness of the three feature types that they investigated (word token, word lemma, and POS) was remarkably similar across the two tasks, with the word features dominating in both cases. Surprisingly, the effect of POS was higher in topic classification than it is on L1-classification. Earlier work of ours (Brooke and Hirst, 2011) also tested the confounding effect of topic in the context of native language identification. To motivate the use of new corpora for future research, we segregated a portion of the ICLE by topic and found that all the core features used by Koppel et al. for L1-identification showed significant drops in performance when topic-segregated 2-fold cross-validation is compared to standard (randomized) 2-fold cross-validation. This was particularly true of character *n*-grams.

[1] http://www.cup.cam.ac.uk/gb/elt/catalogue/subject/custom/item3646603/Cambridge-International-Corpus-Cambridge-Learner-Corpus

Finally, we note that native language identification has also been included as an element of larger author profiling studies (Estival et al., 2007; Garera and Yarowsky, 2009). A closely related task is the identification of translated texts and/or their language of origin (Baroni and Bernardini, 2006; van Halteren, 2008; Koppel and Ordan, 2011), though the tasks are distinct because the learners included in native language identification studies are usually at a level of linguistic proficiency below that of a professional translator (who in any case may be writing in his or her L1, rather than an L2) and are not operating under the requirement of faithfulness to some original text. Distinguishing whether or not a text is non-native (Tomokiyo and Jones, 2001) is also a related task, but most work in the area of L1 identification, including ours, assumes that we already know that a text was produced by a non-native speaker.

3 Corpora

Our training corpus is a set of 154,702 English journal entries collected from the Lang-8 language learner website.[2] In all, 65 L1s are represented, but only 14 languages have more than 1000 entries, with Asian languages being overrepresented (the website is based in Japan). Users may write whatever they want in their journal, and there are a variety of text types (some learners post assignments or translation questions, for instance), though the majority can be described as short personal narratives. The English proficiency of these learners varies widely; other users of the site can comment on journal entries and offer suggestions to improve the text. In the corpus, the entries are tagged for username, title, native language (which is provided by the user when they register), and date and time of posting. The average length of an entry is about 150 word tokens after our pre-processing, which strips HTML tags and non-ASCII characters prior to parsing—words with non-ASCII characters are ultimately disregarded during feature selection. Since these texts are relatively short compared to our test sets, for our purposes here we append consecutive short texts of writers with the same L1 (often the same author) until they are at least 250 tokens in length, which results in an average length of 431 tokens.

The International Corpus of Learner English (Granger et al., 2009), or ICLE, is a set of 6085 non-native speaker essays collected from university students at institutions around the world; v2.0 represents includes 16 L1 backgrounds, mostly European. Other variables tagged in the corpus are topic, genre (argumentative or literary), setting (timed writing or not), age, gender, and educational institution, all of which vary in unpredictable ways throughout the corpus. As already mentioned, a major problem with the ICLE is topic variation, which is both unnaturally strong and often arbitrary; for practical reasons, the different parts of the corpus were collected by EFL instructors in different countries, who chose a small, often fairly distinct set of topics for their students (of a particular L1 background) to write about. To investigate the proficiency level of students, the creators tested a sample of each native language for English level using the *Common European Framework* (CEF), showing that while learners in the corpus are generally at least of intermediate proficiency, the percentage of advanced learners is very different for different L1 backgrounds, another potential confound. The average text length in the ICLE is 617 words.

Our third learner corpus is a small sample of the First Certificate in English (FCE) portion of the Cambridge Learner Corpus, which has recently been released for the purposes of

[2]The URL is http://lang-8.com. We do not have permission to distribute the corpus directly; following Sharoff (2006), we will release a list of web URLs and software which can be used to recreate the corpus.

L1	Corpus		
	Lang-8	ICLE	FCE
Japanese	59156	366	81
Chinese	38044	982	66
French	1414	347	146
Spanish	3080	251	200
Italian	1072	392	76
Polish	1549	365	76
Russian	7159	276	83

Table 1: Number of texts in learner corpora, by L1.

essay scoring evaluation (Yannakoudakis et al., 2011); 16 different L1 backgrounds are represented. Each of the 1244 texts consists of two short answers in the form of a letter, a report, an article, or a short story, each tagged with the score provided by a trained examiner. The texts are also marked for specific usage errors, though we stripped this information in our pre-processing step. The average length of the texts in the FCE corpus is 428 words, or about 200 words less than the ICLE.

For this study, we selected the seven languages which had sufficient numbers in all three corpora, i.e. at least 1000 texts in the Lang-8 corpus, 200 texts in the ICLE, and 50 texts in the FCE. Table 1 shows, for each L1, the number of texts present in each corpus. For testing in the ICLE, we use 200 from each set, and a separate set of 50 per L1 is used for our bias adaptation method. For testing in the FCE, we use 50 texts, and 15 texts for bias adaptation.

4 Classifier Experiments

We split our main experiments into two parts. In our initial investigation, we found that using the full set of feature types, to be described later in Section 5, provided near-optimal results. Given that exploring the exhaustive set of combinations is not feasible in this space, we elect to first take the full feature set as fixed and turn our attention to higher-level classifier options, establishing the best among those options before we proceed with a feature analysis.

4.1 Classifier Options

Our experiments included testing the following options:

Balanced training (bal) vs. cost weight (cost) Statistical classifiers generally depend on having similar class distributions in training and testing sets, an assumption which is violated here. There are two simple ways to handle this problem: either balancing the training sets by discarding extra training data, or training the classifier with using different cost weights for different classes, promoting classification of rarer classes to the level expected in the (balanced) test data. We use the cost weight equation from Morik et al. (1999).

Binary (bin) vs. frequency (freq) features Previous work has mostly used normalized frequency rather than binary occurrence in a text as the feature value used for classification; Wong and Dras (2011) are an exception, but they do not justify that choice.

SVM vs. MaxEnt classifier Support vector machines were a popular option in previous work, but Wong and Dras (2011) report better performance with a Maximum Entropy (MaxEnt) classifier. A full discussion of these two machine learning methods is omitted here, though we note that (pairwise) SVMs are generally conceptualized as a hyperplane which maximizes the margin between classes in the feature space, while MaxEnt is a multinomial logistic model built by constrained maximization of the probability of the training data. For SVM classification (see below), we use LIBLINEAR (Fan et al., 2008), which is optimized for linear kernel classification of large datasets; except as explicitly mentioned below, we present results using default parameter settings (which were found to give good results). Feature vectors are normalized to the unit circle (Graf and Borer, 2001). For MaxEnt we follow Wong and Dras (2011) in using MegaM.[3]

Regularization parameters In the context of building a robust classifier for cross-corpus classification, the regularization of the model (Alpaydin, 2010), i.e. the degree to which the classifier increases in complexity to fit the training data, is of obvious relevance. For SVMs, the key parameter is C, which controls the penalty for misclassified examples in the training set: a large value of C means these errors have a higher influence on the objective function, promoting more complex models that minimize error but may result in overfitting. For the MaxEnt classifier, the λ parameter controls the influence of a Gaussian prior on the feature weights: low values of λ correspond to an imprecise prior, allowing the feature weights to fit the data. We tuned the corresponding parameter for each classifier configuration using 7-class task performance in the development set for each test corpus.[4]

Multiclass SVM type While MaxEnt has a natural multiclass interpretation, an SVM decision plane is appropriate only for binary choice. A standard approach to multiclass SVM is to combine multiple pairwise SVM classifiers (Hsu and Lin, 2002). Two general options in this vein are *one vs. one* (1v1), where $n(n-1)/2$ individual classifiers (for n classes), each trained on one pair of classes, are combined, and *one vs. all* (1va), where n classifiers are trained by separating one class from all the others. The winner of 1va is obviously the class with the highest margin (distance from the decision plane), but for 1v1 it is typically the class which is chosen by the most classifiers (ties are broken in favor of the highest margins). A third, novel option is made possible by the genetic relationships among languages in our test set: an SVM binary decision tree (tree), presented in figure 1.[5] Note that tree classifiers have a significant performance advantage over both 1va and 1v1 classifiers with respect to the number of classifiers required ($n-1$), and an advantage over the 1va classifiers with respect to the average size of the training sets used to build those classifiers. Finally, Crammer and Singer (2002) have proposed a multiclass SVM classifier based on class prototypes (pro) rather than hyperplane boundaries, and we also test this option (as implemented in LIBLINEAR).

Bias adaption, pairwise (adS) Since there are significant differences in the genre, domain, and quality of texts across our training and test corpora, some form of domain adaption (Daumé and Marcu, 2006; Bruzzone and Marconcini, 2010) would almost certainly be

[3] http://www.cs.utah.edu/~hal/megam/

[4] Since the C parameter is selected once for each configuration based on the 7-class task, some results that we would otherwise expect to be equivalent, e.g. the 2-class SVM classifers, actually vary slightly.

[5] There is some controversy in the literature about the genetic relationship amongst Romance languages; see the discussion by Kochmar (2011).

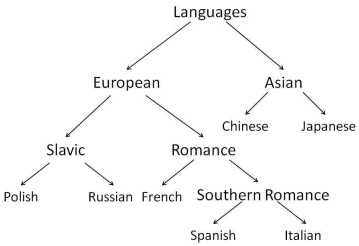


Figure 1: Binary decision tree for SVM experiments

helpful. However, even unsupervised forms of transfer learning (Pan and Yang, 2010) are likely to take advantage of those confounding factors that prompted us to reject within-corpus evaluation; we believe that any change to the feature weights based on samples from the same corpus that our test set is drawn from is ultimately self-defeating in this context. However, there is one key parameter to these that is not a feature weight: the bias. In pairwise SVM, changing the bias slides the hyperplane, changing only the total number of positive (or negative) features required to make a classification, not the individual influence of a particular feature (i.e. the sign of a feature weight). With respect to its effect (changing the balance of classes), it is closely related to our cost factor option above; however, whereas the cost factor is a parameter used during training, we shift the bias using our own iterative process after the model is built, using a sample from the same corpus as the test set (a development set).[6] Our algorithm is as follows: we first initialize our step size to the absolute value of the original bias, and then we iteratively modify the bias, adding or subtracting the present step size such that we are moving in the direction of a distribution where the ratio of classes predicted in our development set is the same as in the final test set, reclassifying the data after each step.[7] If we overshoot the desired ratio, we halve the step size, and continue until we reach the desired ratio or the predicted ratio does not change for 10 iterations. We do this separately for each test set with the corresponding development set.

Bias adaption, multi (adM) The MaxEnt and SVM prototype classifiers also have bias terms that can be optimized, but unlike the pairwise classifiers they cannot be dealt with one at a time; optimizing the bias for one class will affect the others in unpredictable ways. We proceed with the same basic algorithm as the pairwise classifier, but we do this for all bias terms simultaneously, i.e. all biases are adjusted in a single step. Each bias has a separate step size, and the optimization ends when the entire distribution is correct or nothing has changed in 10 iterations. We also implemented this for SVM 1va, i.e. interpreting it as a single multiclass classifier rather than a set of pairwise classifiers.

[6]Admittedly, we could accomplish this with additional parameter tuning, but there are both practical and principled reasons for doing it this way: it is much faster to modify the biases directly rather than retraining the model, and, more importantly, we want to preserve the original feature weights; we require that they do not reflect exposure to the confounds of the testing corpus in any way.

[7]This requires knowledge of that distribution. However, it is otherwise unsupervised in that we are only concerned with the distribution of predictions: we do not use the true class values except to create the appropriate subsets for the SVM 1v1 and SVM tree classifiers.

Configuration	Asian		European		All	
	ICLE	FCE	ICLE	FCE	ICLE	FCE
Chance baseline	50.0	50.0	20.0	20.0	14.3	14.3
(1) SVM *1v1* cost bin	95.2	86.0	50.0	40.4	58.7	50.3
(2) SVM *tree* cost bin	95.2	86.0	48.7	41.3	59.4	49.4
(3) SVM 1va cost bin	**96.5**	86.0	54.8	44.0	61.6	50.8
(4) SVM *pro* cost bin	95.0	85.0	55.6	42.8	62.4	50.8
(5) *MaxEnt* cost bin	95.0	85.0	56.6	44.8	63.7	42.3
(6) SVM *tree*-adS cost bin	95.2	**88.0**	64.4	57.2	73.7	57.4
(7) *MaxEnt*-adM cost bin	95.0	86.0	68.2	64.4	74.0	60.8
(8) SVM *1v1*-adS cost bin	95.5	88.0	67.9	66.8	74.2	65.7
(9) SVM 1va-*adS* cost bin	95.0	88.0	71.6	67.6	77.8	**66.5**
(10) SVM *pro*-adM cost bin	95.7	87.0	71.1	66.4	77.3	64.0
(11) SVM 1va-adM cost bin	95.0	86.0	**71.7**	**68.0**	**78.0**	65.7
(12) SVM 1va-adM *bal* bin	79.8	75.0	63.1	60.4	66.8	59.1
(13) SVM 1va-adM cost *freq*	95.2	83.0	66.8	57.6	74.9	53.1

Table 2: Native language classification accuracy (%) for varying classifier options. Bold indicates best result in column, italics indicates difference from the pivot classifier (11).

4.2 Results

Table 2 shows the results of our experiments. In addition to the full 7-language task accuracy (the 'All' columns), we also present results classifying the two major subgroups; note that these are distinct tasks, e.g. for European it is the accuracy of a 5-language task, not the accuracy of the classification of those 5 languages within the 7-language task (see Figure 1 for our language classification schema). However, in our discussion, we focus on results for the full 7-language task. The upper part of Table 2 includes various key classifier options, ordered by their 7-way ICLE accuracy, while the bottom includes other options; the best classifier (11) is used as a pivot between the two.[8] The aspect(s) of the configuration that are different from the pivot are in italics, and the best results in each column are in bold. For each classifier, we report the results using the best C or λ values from an initial series of runs using the development set.

Unsurprisingly, we see better results when we use all the data at our disposal (11), rather than forcing balanced test cases (12). This result is useful, though, because it indicates that our consistently high performance in distinguishing Chinese and Japanese elsewhere in Table 2 is a result of that extra data, and not other factors, i.e. the fact that unlike our other language groupings, Chinese and Japanese do not belong to a single genetic language family (Comrie, 1987). Also clear is the preference for binary (11) rather than frequency-based (13) feature values: one possible explanation is that, in these relatively short texts, there is high variability in normalized frequencies, and a simpler metric, by having less variability, is easier for the classifier to leverage. In general, slightly less regularization (high C, low λ) values were preferred, though most were reasonably close to the default values; tuning made little difference, particularly for the SVM classifiers.

[8] The effects of the options in each of the two parts of the table are fairly independent, so for simplicity of presentation we test them separately.

Between the two main classifier types, the MaxEnt classifier was, with the appropriate choice of λ (5), the best performing classifier in the ICLE when no bias adaption was used; it was, however, worse than almost all of our SVM options in the main 7-language classification task when bias tuning was allowed (7). This does not appear to be a failure of the adaption algorithm, but rather a real distinction between the two classifiers: our experience is that the SVM classifiers are less robust, i.e. more prone to errors when training and test sets differ significantly, but they can be easily recalibrated for optimal performance with a relatively small amount of information. Here, we show that changing the bias alone is enough for major gains across all the SVM types (6,8–11), results which are statistically significant.

Our novel binary tree classifier (2,6) is competitive but ultimately performs poorly compared than other options, suggesting that the simplicity of the classifier does come with a trade-off in performance. The 1va classifiers (3,9,11) are consistently better than 1v1 (1,8), while the performance of the prototype-based SVM (4,10) is nearly indistinguishable from 1va. This is somewhat surprising, since we might expect a 1v1 or prototype approach to be able to better deal with the commonalities and differences among languages than the 1va, which lumps diverse languages into a single 'other' category. With respect to the 1va classifier, it does not seem to matter much whether pairwise (9) or single classifier (11) bias tuning is used; the latter gave us the best 7-class performance in the ICLE (and we use it as our best classifier), but the former gave slightly better performance in the FCE. In the ICLE, the difference between the best bias-adapted 1va classifier and the 1v1, tree, and MaxEnt classifiers is statistically significant (χ^2 test, $p < 0.001$).

5 Feature Analysis

5.1 Features

Our model includes the following feature types:

Function words A common feature in stylistic analysis. Our list of 416 common English words comes from the LIWC (Pennebaker et al., 2001).

Character n-grams (unigrams, bigrams, and trigrams) For bigrams and trigrams, the beginning and end of a word are treated as special characters.

Word n-grams (unigrams and bigrams) Note that word n-grams are a superset of function words. Punctuation is included.

POS n-grams (unigrams, bigrams, and trigrams) POS tagging is provided by the Stanford Parser V1.6.9 (Klein and Manning, 2003), also used by Wong and Dras (2011).

POS/function mixture n-grams (bigrams and trigrams) Wong et al. (2012) report better results with POS n-grams that retain the identity of individual function words rather than using their part of speech.

CFG productions Context-free grammar production rules, as provided by the Stanford parser. Lexical production rules are not included.

Dependencies Dependencies consist of two lexical items and the syntactic relationship between them. Also produced by the Stanford parser (de Marneffe et al., 2006).

Features	Asian		European		All	
	ICLE	FCE	ICLE	FCE	ICLE	FCE
Chance baseline	50.0	50.0	20.0	20.0	14.3	14.3
(1) Function words	72.7	71.0	40.3	37.2	35.6	36.0
(2) Character *n*-grams	78.3	63.0	37.5	28.8	37.4	22.6
(3) POS *n*-grams	86.8	78.0	47.9	50.0	52.9	44.3
(4) POS/function *n*-grams	93.3	85.0	60.6	56.8	67.4	59.4
(5) CFG productions	78.5	72.0	46.9	43.2	49.7	41.1
(6) Dependencies	94.0	79.0	49.8	46.8	61.4	45.1
(7) Word *n*-grams	94.3	89.0	71.1	66.8	77.1	68.3
(8) Syntactic Features	94.3	87.0	60.1	61.2	68.1	65.1
(9) Lexical Features	95.2	86.0	71.0	67.6	77.8	67.1
(10) Lexical+Syntactic	**96.0**	90.0	72.3	66.4	78.4	68.2
(11) All features	95.0	86.0	71.7	68.0	78.0	65.7
(12) (4)+(7)	95.5	90.0	**72.5**	66.8	**79.3**	**70.0**
(13) (4)+(7), no proper nouns	94.5	87.0	69.6	67.2	76.5	65.7
(14) (4)+(7), df ≥ 20	95.0	86.0	71.3	**68.4**	77.3	65.4
(15) (4)+(7), IG > 0	89.5	**93.0**	69.5	66.4	76.5	65.7

Table 3: Native language classification accuracy (%), by feature set. Bold indicates best result in column.

Syntactic Features POS *n*-grams, POS/function mixture *n*-grams, and CFG productions.

Lexical Features Word *n*-grams and dependencies.

Proper Nouns Not actually a separate feature, proper nouns are included by default in character and word *n*-grams as well as dependencies. They are obviously relevant to the task, but there are applications (e.g. forensic profiling) where they might not be appropriate, since they do not directly indicate language transfer from the L1 but rather reflect real-world correlations between native language and country of residence, etc. Here, we report results with all proper nouns excluded from consideration for all relevant features.

Feature Selection Wong and Dras (2011) tested feature selection based on information gain, but it provided no improvement in performance. For practical reasons, we have included by default a simple frequency-based feature selection; only features that appear in 5 different texts in the training set are included. Even with this restriction, our feature set has almost 800,000 features. Here, we test the effect of a higher frequency cutoff (at 20), and limiting our set to features with positive information gain.

5.2 Analysis

Again, we focus on the results of the full 7-language task (the 'All' columns). Clearly, all the feature types can be used to distinguish native language: each of the results in Table 3 is well above a chance baseline, though function words (1) and character *n*-grams (2) give a fairly modest performance individually. Compared to these, production (5) rules are markedly more useful, a result which is compatible with the conclusions of Wong and

Dras (2011). Nonetheless POS (3) and in particular mixed POS/function words *n*-grams (4) offer even better performance, despite being somewhat simpler. Compared to the latter of these, the usefulness of lexical dependencies (6) is muted, and shows a very inconsistent performance across the two test sets. Word *n*-grams (7), however, alone account for almost all of the accuracy we see when all features are combined.

Adding the POS features and CFG productions (8) generally boosts performance, suggesting that the syntactic features may not be entirely redundant, while the combination of the lexical features also provides a small improvement in the 7-language ICLE task, though the FCE is worse (9). Further adding the syntactic features to the lexical features increases performance for most of the tasks (10), while including character *n*-grams tends to degrade performance (11). Finally, we exhaustively tested feature combinations and found that the best performing for the 7-language task used only the two best individual feature types, POS/function word mixtures and lexical *n*-grams, though the differences among all the options containing lexical *n*-grams are not statistically significant (12).

When we remove proper nouns (13), there is a modest drop in performance, indicating that they had some positive role in the classification, but the benefits of using lexical features goes well beyond proper nouns. Additional frequency-based feature selection (14) has a small, mostly negative effect, as does restricting features to those with positive information gain (15). In general, we see no evidence that a simpler model is preferred in this case, though if speed is a concern one can be used without too much loss.

We also looked briefly at the individual lexical features that were useful based on their information gain in the training set. One thing that was immediately evident is that some common, entirely correct English words and expressions were extremely helpful for distinguishing native languages. For example, the phrase *decide to* was ranked high: we note that in at least one language in our set (French), a closely analogous cognate construction *decider de* exists, whereas another language, Chinese, has no analogous construction, since the verb that most closely means *decide to* (*jueding*) is phonetically dissimilar, has no element corresponding to *to*, is more common as a noun, and in fact is pragmatically associated only with major decisions, often in a legal context (closer to the English *make a decision to*). By default, learners will prefer forms that correspond to those from their L1 (Odlin, 1989), and lexical features are key to identifying this kind of language transfer.

6 ICLE-training Experiments

One of the primary motivations for our cross-corpus approach to NLI is the confounding variation found in the ICLE corpus. In this section, we turn to using the ICLE as a training corpus in order to highlight these problems, particularly those relevant to 'stylistic' features, which have been thought of as immune to these effects. The first experiment, the results of which are presented in Table 4, consists of two types of 2-fold cross-validation in the ICLE corpus: the first is standard, randomized cross-validation, while in the second, the two folds (of 700 texts each) are segregated by essay prompt; essays based on a given prompt are in one fold or the other.[9] For this we use the 1va classifier without any bias adaption, which is unnecessary in the case of cross-validation.

Within the ICLE, we see in the 'Difference' column of Table 4 the consistent effects of essay

[9]This experiment is possible only in the ICLE, since titles in the Lang-8 are freely chosen by each writer, and there is little variety of prompts in the FCE.

Features	Random	Segregated	Difference
Chance baseline	14.3	14.3	–
(1) Function words	58.0	46.7	−11.3
(2) Character *n*-grams	51.2	48.2	−3.0
(3) POS *n*-grams	83.3	72.2	−11.1
(4) POS/function *n*-grams	87.6	79.2	−10.4
(5) CFG productions	86.1	79.7	−6.4
(6) Dependencies	89.1	77.1	−12.0
(7) Word *n*-grams	94.3	81.3	−13.0
(8) All (1–7)	90.4	81.6	−8.8

Table 4: ICLE within-corpus experiment classification accuracy (%), by feature set.

prompt on classification, across all kinds of features. The effects on lexical features (6,7) are, not surprisingly, most pronounced, but other popular features are also implicated to varying degrees. The effectiveness of various features under both conditions roughly mirrors the results in the previous section, though there are a few notable exceptions: for instance, production rules (5) were more useful here than in the Lang-8 trained cross-corpus experiments; this is interesting since many of the most recent results in the ICLE (Wong and Dras, 2011; Swanson and Charniak, 2012) make use of these grammatical features. Surprisingly, character *n*-grams were the least affected, a contrast from our earlier work on ICLE topic bias (Brooke and Hirst, 2011), though there remains little doubt that they are inferior features for this task. Lexical *n*-grams are ultimately the most preferred feature (7), even when topic effects are partially[10] controlled for.

We also present cross-corpus experiments with the FCE and a language-balanced 150-text portion of the Lang-8 corpus as test sets. As with our training set, this test set consists of combined texts, this time with a minimum length of 500, making the texts of comparable length to those in the ICLE. We create another set of 50 texts for bias adaption. In the latter experiment, we also include a special set of features: the POS/function mixture 5-grams which were selected by the adaptor grammars of Wong et al. (2012), providing superior performance over exhaustive enumerations. Since these features were derived from the ICLE, they could not be defensibly used in other experiments (i.e. with the ICLE as a test set), but we can test their usefulness here. Since the original experiment involved cross-validation, there are in fact 5 different sets; our set consists of the union of these sets.[11]

The cross-corpus results in Table 5 are strikingly lower than the within-ICLE results. They also compare poorly to our earlier cross-corpus results in this paper. Part of this difference is, of course, the effect of the much-larger Lang-8 dataset, though the balanced result in Table 2 (12), uses a very similar amount of data (as measured in tokens) from the Lang-8 but attains a much better FCE classification accuracy (roughly 20% better). The POS/function mixture

[10]There are more pervasive topic and genre effects that segregating by prompt does not resolve. For instance, a large number of the Japanese texts are personal narratives, each with a different title, while in the Russian texts there is a particular focus on the literature of various authors, and in the Chinese texts there is a discussion of the advantages or disadvantages associated with certain government policies.

[11]We originally intended to take the intersection, but in fact the intersection of the feature sets is empty; no single feature was useful in every fold.

Features	Lang-8	FCE
Chance baseline	14.3	14.3
(1) Function words	27.6	20.0
(2) Character *n*-grams	29.7	24.0
(3) POS *n*-grams	37.0	32.8
(4) POS/function *n*-grams	40.2	33.4
(5) CFG productions	32.5	31.4
(6) Dependencies	30.7	25.1
(7) Word *n*-grams	50.8	35.7
(8) All (1–7)	46.8	39.1
(9) Adaptor grammar *n*-grams	40.9	30.8

Table 5: ICLE-training cross-corpus classification accuracy (%), by feature set.

features (9) derived using adaptor grammars do reasonably well, but are only marginally better than exhaustive mixture features (4) in the Lang-8 test set, and are markedly worse than a number of other features in the FCE. Again, lexical *n*-grams (7) are obviously the best individual feature type.

7 Discussion

The results in the previous section highlight the problematic nature of within-corpus evaluation in general, and the inadequacy of the ICLE as a training corpus in particular. It is unclear to what extent previous results on this task are influenced by these effects, but we believe there is at least reason to be skeptical of some of the conclusions. In particular, sophisticated feature selection techniques which have been the focus of recent work may result in models which perform better in the ICLE, but which have little or no benefit beyond that particular corpus. We believe more attention should be paid to the overall validity of NLI experiments, rather than to specific technical approaches. One interesting open question is whether features such as proper nouns, which are of obvious but somewhat trivial benefit, should be excluded. Certainly, we would argue that lexical features in general are far too important to the task to simply be discarded; our experiments here suggest that their usefulness goes well beyond proper nouns and is not simply a reflection of topic.

Though higher performance is clearly possible using cross-validation, our Lang-8 trained model does reasonably well in both our testing corpora; the results are fairly consistent, and the difference can be attributed to the smaller size of the FCE texts. It is clear that factors such as the choice of classifier and the size of the dataset play some role, though the most obvious improvement came from the use of our bias adaptation technique, which uses a small amount of data from a test corpus to improve the model; this was particularly effective for SVMs. Importantly, this method keeps the feature weights constant, a necessary precondition when the testing corpus has known arbitrary biases. Given the variation in text size, genres, and learner proficiency, some kind of adaption is clearly necessary to get competitive results, though our experiments using it with the ICLE as training data suggest the method cannot overcome a problematic training set.

We note that our still sizable error rate on this task may in fact be due to a learner proficiency effect; on inspection, one of the authors (a native speaker of English) found that some of

the European texts were nearly indistinguishable from native writing. As suggested by the statistics provided in the ICLE manual (Granger et al., 2009), many of these learners are highly proficient, and thus they might have completely integrated the norms of their L2, making them legitimately indistinguishable. We also tested the correlation between essay scores in the FCE and our classification accuracy, and found a small negative correlation, suggesting that those who scored better were harder to classify; text length, though, was a confounding factor, since longer texts got better scores and are also easier to classify. Finally, we also noticed that French was the most consistently misclassified language, by a significant margin; this could be due, in part, to the historical connection between French and English that makes French L1 transfer somewhat less distinct, whereas distant languages like Chinese and Japanese are easy discerned, an effect we saw even when the training sets were balanced. In general, we think the relationship between proficiency, distances between languages, and L1 classification merits further study.

One important strength of the current work is the training dataset, which, unlike many learner corpora resources, is fully accessible via the web (and growing!). The coverage of European languages is poor, however, and since large amounts of data are necessary to fully leverage the potential of lexical features, one future direction would be to look for even more inexpensive ways of finding learner texts, perhaps by collecting English texts that appear on otherwise non-English websites. Armed with larger datasets, we would like to move beyond classification of a handful of L1s, moving towards a system that can identify influence from a full range of common L1 backgrounds.

8 Conclusion

In this paper, we have demonstrated the feasibility of a cross-corpus approach to the development of native language identification systems, sidestepping the problem of within-corpus confounds to test the efficacy of relevant options. The most striking result is the dominance of shallow lexical features in our best classifier, even in comparison to high-performing, sophisticated feature types such as syntactic productions; also important is some degree of domain adaption, and we present a simple but highly effective method. We have also highlighted the not-insignificant role that other choices play in the classifier performance; for instance, the *one vs. all* classifier, which has been somewhat maligned in SVM multiclass comparisons (Hsu and Lin, 2002; Duan and Keerthi, 2005), provides the best performance in both test corpora when our simple bias adaption method is applied. We have also presented new evidence that within-corpus evaluations techniques are problematic, and that the validity of results that use the ICLE in this manner need to be re-evaluated.

Acknowledgements

Thanks to Gabriel Murray for bringing the Lang-8 website to our attention, and Jojo Wong and Mark Dras for providing their adaptor grammar features. This work was financially supported by the Natural Sciences and Engineering Research Council of Canada.

References

Alpaydin, E. (2010). *Introduction to Machine Learning*. The MIT Press, 2nd edition.

Baroni, M. and Bernardini, S. (2006). A new approach to the study of translationese: Machine-learning the difference between original and translated text. *Literary and Linguistic Computing*, 21:259–274.

Brooke, J. and Hirst, G. (2011). Native language detection with 'cheap' learner corpora. Presented at the 2011 Conference of Learner Corpus Research (LCR2011).

Bruzzone, L. and Marconcini, M. (2010). Domain adaptation problems: A DASVM classification technique and a circular validation strategy. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32:770–787.

Chang, Y.-C., Chang, J. S., Chen, H.-J., and Liou, H.-C. (2008). An automatic collocation writing assistant for Taiwanese EFL learners: A case of corpus-based NLP technology. *Computer Assisted Language Learning*, 21(3):283–299.

Comrie, B., editor (1987). *The World's Major Languages*. Oxford University Press, Oxford.

Crammer, K. and Singer, Y. (2002). On the algorithmic implementation of multiclass kernel-based vector machines. *Journal of Machine Learning Research*, 2:265–292.

Dahlmeier, D. and Ng, H. T. (2011). Correcting semantic collocation errors with L1-induced paraphrases. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing (EMNLP '11)*, pages 107–117, Edinburgh, Scotland, UK.

Daumé, H. and Marcu, D. (2006). Domain adaptation for statistical classifiers. *Journal of Artificial Intelligence Research*, 26:101–126.

de Marneffe, M.-C., MacCartney, B., and Manning, C. D. (2006). Generating typed dependency parses from phrase structure parses. In *Proceedings of the 5th International Conference on Language Resources and Evaluation (LREC '06)*, Genova, Italy.

Duan, K.-B. and Keerthi, S. S. (2005). Which is the best multiclass SVM method? An empirical study. In *Proceedings of the Sixth International Workshop on Multiple Classifier Systems*, pages 278–285.

Estival, D., Gaustad, T., Pham, S. B., Radford, W., and Hutchinson, B. (2007). Author profiling for English emails. In *Proceedings of the 10th Conference of the Pacific Association for Computational Linguistics (PACLING '07)*, pages 263–272.

Fan, R.-E., Chang, K.-W., Hsieh, C.-J., Wang, X.-R., and Lin, C.-J. (2008). LIBLINEAR: A library for large linear classification. *Journal of Machine Learning Research*, 9:1871–1874.

Fette, I., Sadeh, N., and Tomasic, A. (2007). Learning to detect phishing emails. In *Proceedings of the 16th International World Wide Web Conference (WWW '07)*, pages 649–656, Banff, Alberta, Canada.

Garera, N. and Yarowsky, D. (2009). Modeling latent biographic attributes in conversational genres. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP (ACL-IJCNLP '09)*, pages 710–718, Singapore.

Golcher, F. and Reznicek, M. (2011). Stylometry and the interplay of title and L1 in the different annotation layers in the Falko corpus. In *Proceedings of Quantitative Investigations in Theoretical Linguistics 4*, Berlin.

Graf, A. B. A. and Borer, S. (2001). Normalization in support vector machines. In *Proceedings of the 23rd DAGM-Symposium on Pattern Recognition*, pages 277–282.

Granger, S., Dagneaux, E., Meunier, F., and Paquot, M. (2009). *International Corpus of Learner English (Version 2)*. Presses Universitaires de Louvain, Louvain-la-Neuve.

Hsu, C.-W. and Lin, C.-J. (2002). A comparison of methods for multi-class support vector machines. *IEEE Transactions on Neural Networks*, 13:415–425.

Klein, D. and Manning, C. D. (2003). Accurate unlexicalized parsing. In *Proceedings of the 41st Meeting of the Association for Computational Linguistics*, pages 423–430.

Kochmar, E. (2011). Identification of a writer's native language by error analysis. Master's thesis, University of Cambridge.

Koppel, M. and Ordan, N. (2011). Translationese and its dialects. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics (ACL '11)*, Portland, Oregon.

Koppel, M., Schler, J., and Zigdon, K. (2005). Determining an author's native language by mining a text for errors. In *Proceedings of the 11th ACM SIGKDD International Conference on Knowledge Discovery in Data Mining (KDD '05)*, pages 624–628, Chicago, Illinois, USA.

Leacock, C., Chodorow, M., Gamon, M., and Tetreault, J. (2010). *Automated Grammatical Error Detection for Language Learners*. Morgan & Claypool.

Lüdeling, A., Doolittle, S., Hirschmann, H., Schmidt, K., and Walter, M. (2008). Das Lernerkorpus Falko. *Deutsch als Fremdsprache*, 42:67–73.

Morik, K., Brockhausen, P., and Joachims, T. (1999). Combining statistical learning with a knowledge-based approach — a case study in intensive care monitoring. In *Proceedings of the Sixteenth International Conference on Machine Learning*, ICML '99, pages 268–277.

Odlin, T. (1989). *Language Transfer*. Cambridge University Press.

Pan, S. J. and Yang, Q. (2010). A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering*, 22(10).

Pennebaker, J. W., Francis, M. E., and Booth, R. J. (2001). *Linguistic Inquiry and Word Count (LIWC): LIWC2001 Manual*. Erlbaum Publishers, Mahwah, NJ.

Rozovskaya, A. and Roth, D. (2011). Algorithm selection and model adaptation for ESL correction tasks. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics (ACL '11)*, Portland, Oregon.

Sharoff, S. (2006). Open-source corpora: using the net to fish for linguistic data. *International Journal of Corpus Linguistics*, 11(4):435–462.

Swanson, B. and Charniak, E. (2012). Native language detection with tree substitution grammars. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (ACL '12)*, pages 193–197, Jeju, Korea.

Tomokiyo, L. M. and Jones, R. (2001). You're not from 'round here, are you?: naïve Bayes detection of non-native utterance text. In *Proceedings of the second meeting of the North American Chapter of the Association for Computational Linguistics on Language technologies (NAACL '01)*, pages 1–8, Pittsburgh, Pennsylvania.

Tsur, O. and Rappoport, A. (2007). Using classifier features for studying the effect of native language on the choice of written second language words. In *Proceedings of the Workshop on Cognitive Aspects of Computational Language Acquisition (CACLA '07)*, pages 9–16, Prague, Czech Republic.

van Halteren, H. (2008). Source language markers in EUROPARL translations. In *Proceedings of the 22nd International Conference on Computational Linguistics (COLING '08)*, pages 937–944, Manchester, UK.

Wong, S.-M. J. and Dras, M. (2009). Contrastive analysis and native language identification. In *Proceedings of the Australasian Language Technology Association Workshop*, pages 53–61.

Wong, S.-M. J. and Dras, M. (2011). Exploiting parse structures for native language identification. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing (EMNLP '11)*, pages 1600–1610, Edinburgh, Scotland, UK.

Wong, S.-M. J., Dras, M., and Johnson, M. (2012). Exploring adaptor grammars for native language identification. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL '12)*, Jeju, Korea.

Yannakoudakis, H., Briscoe, T., and Medlock, B. (2011). A new dataset and method for automatically grading ESOL texts. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics*, pages 180–189, Portland, Oregon.

Identifying Urdu Complex Predication via Bigram Extraction

Miriam Butt[1] *Tina Bögel*[1]
Annette Hautli[1] *Sebastian Sulger*[1] *Tafseer Ahmed*[2]
(1) Universität Konstanz, Germany
(2) University of Karachi, Pakistan
firstname.lastname@uni-konstanz.de[1], tafseer@uok.edu.pk[2]

ABSTRACT
A problem that crops up repeatedly in shallow and deep syntactic parsing approaches to South Asian languages like Urdu/Hindi is the proper treatment of complex predications. Problems for the NLP of complex predications are posed by their productiveness and the ill understood nature of the range of their combinatorial possibilities. This paper presents an investigation into whether fine-grained information about the distributional properties of nouns in N+V CPs can be identified by the comparatively simple process of extracting bigrams from a large "raw" corpus of Urdu. In gathering the relevant properties, we were aided by visual analytics in that we coupled our computational data analysis with interactive visual components in the analysis of the large data sets. The visualization component proved to be an essential part of our data analysis, particular for the easy visual identification of outliers and false positives. Another essential component turned out to be our language-particular knowledge and access to existing language-particular resources. Overall, we were indeed able to identify high frequency N-V complex predications as well as pick out combinations we had not been aware of before. However, a manual inspection of our results also pointed to a problem of data sparsity, despite the use of a large corpus.

KEYWORDS: Urdu, complex predicates, visualization, bigrams, corpus.

Proceedings of COLING 2012: Technical Papers, pages 409–424,
COLING 2012, Mumbai, December 2012.

1 Introduction

A problem that crops up repeatedly in shallow and deep syntactic parsing approaches to South Asian languages like Urdu/Hindi[1] is the proper treatment of complex predications. Whereas verbal expressions in European languages are mostly realized with a single predicate (e.g. 'to remember'), South Asian languages tend to use combinations of more than one element to express an action (e.g., memory+do = 'remember'). In Urdu, only about 700 simple verbs exist (Humayoun 2006) — the remaining verbal inventory consists of complex predications. Complex predicates (CPs) are encountered frequently in general language use, as well as in newspaper corpora. Thus, any NLP application, whether shallow or deep, whether its goal be parsing, generation, question-answering or the construction of lexical resources like WordNet (cf. Bhattacharyya 2010), encounters complex predication sooner rather than later.

While the productive capability as well as constraints on V+V combinations are now comparatively well understood from a theoretical point of view (e.g, see Hook 1974; Butt 1995, 2010 and references therein), the constraints governing N+V, ADJ+V as well as P+V combinations are less well understood (the standard reference for N+V is Mohanan 1994, there is no standard reference for the other types). This is despite the fact that these latter three are very productive. Indeed, very little is known overall about P+V combinations (with Raza 2011 providing a first description), but as our results show, they do occur in newspaper corpora (section 5).

With respect to NLP applications the frequency and productivity of complex predications means that it is not possible to construct a static list of N/ADJ/P+V combinations, rather, there must be a way in which one can dynamically predict which kinds of combinations are possible and which should be impossible. In a recent paper, Ahmed and Butt (2011) propose that the combinatory possibilities of N+V combinations are in part governed by the lexical semantic compatibility of the noun with the verb. They propose an initial classification based on a small corpus study. If they are right, then lexical resources such as WordNet or lexica used in deep grammars could be augmented with semantic specifications or feature information that can then be used to determine dynamically whether a given N+V combination is licit or not.

For this paper, we took Ahmed and Butt (2011) as a point of departure and explored whether we could confirm and perhaps extend their results with respect to a larger corpus study. Given that to date no high-quality annotated large-scale corpus for Urdu exists,[2] we decided to experiment with a large-scale 7.9 million newspaper corpus of Urdu we have collected.

The idea was to take advantage of statistical methods and proceed per standard methods currently embraced in the field. That is, use an available corpus that should in principle be large enough to adequately reflect language use and to extract bigrams from this corpus in order to identify patterns in N+V combinations and to use our knowledge of the extracted patterns in further NLP applications.

In pursuing this experiment, we were indeed able to adduce new information about combinatory possibilities in CPs (section 5). However, our experiment also provides a cautionary tale with respect to diving into a corpus "blindly", i.e, assuming that mere statistical analysis will provide good enough results and any noise due to language particular considerations will simply wash out if the corpus is large enough. Some of the difficulties we encountered had to do with the

[1] Urdu is an Indo-Aryan language spoken primarily in Pakistan and parts of India, as well as in the South Asian diaspora. It is structurally almost identical to Hindi, although the orthographies differ considerably.

[2] However, some data is becoming available via the Hindi-Urdu Treebank (Bhatt et al., 2009) and a large-scale, balanced corpus for Urdu will be released soon (Urooj et al., 2012).

non-standardized nature of Urdu orthography, some with the structure of Urdu (section 3) and some with the complex nature of the data (section 4).[3]

With respect to the complex interrelationships between our data, we were able to achieve significant improvement by using novel methods coming from the field of *visual analytics* (Card et al., 1999; Thomas and Cook, 2005; Keim et al., 2008, 2010). Instead of trying to make sense of bare numbers, we used a visualization tool that maps figures to colors and therefore makes the statistical analysis immediately accessible via easy to process visual means. The visualization allowed us to assess our complex data "at-a-glance", to take corrective measures and to generate new hypotheses as to CP formation and the validation of existing hypotheses.[4]

The paper is organized as follows: section 2 presents relevant background information with respect to the linguistic phenomenon of CPs in section 2.1, followed by related work in section 2.2. Sections 3 and 4 present the details of our study. We describe the steps with respect to the corpus preparation and bigram extraction (section 3.1), the statistical basis for the collocation analysis (section 3.2), the clustering performed on N+V bigrams (section 3.3) and the visualization used (section 4). We also reflect on the depth of the language particular knowledge that was necessary and discuss our results in section 5. We were not in fact able to achieve our initial goal of confirming or extending Ahmed and Butt's original hypothesis. This turns out to be due to data sparsity, even with a 7.9 million token corpus. However, we were able to identify previously unreported information about highly productive combinations and gain further structural insights into the language, particularly due to our use of novel methods coming from the field of visual analytics.

2 Motivation and Background

2.1 Urdu and complex predicates

As already mentioned, an important means of expressing verbal concepts in Urdu is the usage of CPs. There is not one single way of forming CPs, rather there are several different kinds of V+V CPs, different kinds of ADJ+V combinations and different classes of N+V CPs (Butt, 1995; Mohanan, 1994). A discussion of the various combinatory possibilities and types of classes identified so far would lead us too far afield, but see Ahmed et al. (2012) for some examples of each type. Moreover, CPs are highly productive with respect to their combinatorial possibilities and can also be stacked on top of one another. This means that the compilation of a static list of possible combinations of different types of CPs is not feasible as it does not solve the challenges inherent in capturing the syntactic combinatory possibilities and the semantic interpretability via computational means — simply searching a corpus for all possible (and potentially infinitely many) combinations cannot do justice to the dynamic and manifold combinatory possibilities.

For the purpose of this paper, we decided to focus on N+V CPs, i.e., CPs which are formed by combining a noun and a verb. The verb is generally called a *light verb* (Mohanan, 1994), as the noun provides the main predicational content, while the verb specifies additional information

[3] In light of these difficulties, one reviewer encouraged us to explore the possibility of using treebanks. However, the existing definitive treebank for Urdu/Hindi does not do well with annotating complex predicates, merely noting that several bits of a clause are "part-ofs" another word or constituent (Bhatt et al., 2009). This part-of relation is not confined to CPs and when it does indicate a CP, it does not specify what kind. Treebanks which aim to do better in this respect are in the process of being built (Hautli et al., 2012; Ahmed et al., 2012). Indeed, the work reported on here was partly motivated by our on-going treebank effort.

[4] The visualization tool we used was designed by Christian Rohrdantz as part of an on-going cooperation. We would also like to thank him for a very useful discussion of the material in this paper.

about the action/event, like whether the action was agentive or telic. On the morphosyntactic side, the light verb determines the case marking on the subject, controls agreement patterns and provides information about tense and aspect. Some standard examples are in (1).[5]

(1) a. لڑکی نے کہانی یاد کی

laṛki=ne kəhani yad k-i
girl.F.Sg=Erg story.F.Sg.Nom memory.F.Sg.Nom do-Perf.F.Sg
'The girl remembered a/the story.' (lit.: 'The girl did memory of the story.')

b. لڑکی کو کہانی یاد ہے

laṛki=ko kəhani yad hɛ
girl.F.Sg=Dat story.F.Sg.Nom memory.F.Sg.Nom be.Pres.3P.Sg
'The girl remembers/knows a/the story.' (lit.: 'Memory of the story is at the girl.')

c. لڑکی کو کہانی یاد ہوی

laṛki=ko kəhani yad hu-i
girl.F.Sg=Dat story.F.Sg.Nom memory.F.Sg.Nom be.Part-Perf.F.Sg
'The girl came to remember a/the story.'
(lit.: 'Memory of the story became to be at the girl.')

In all of the examples in (1), it is evident that the noun and the verb form a single predicational element. The object *kəhani* 'story' is thematically licensed by the noun *yad* 'memory', but it is not realized as a genitive, as would be typical for arguments of nouns (and as in the English literal translations). Rather, *kəhani* 'story' functions as the syntactic object of the joint predication (see Mohanan 1994 for details on the argument structure and agreement patterns).

Mohanan (1994) already identified two subclasses of N+V CPs: one in which the light verb agrees with the noun (and, confusingly, the noun is both a syntactic object and a part of the verbal predication) and one in which the verb does not agree with the noun (and instead agrees with some other NP in the clause; (1) is of the non-agreeing type). Ahmed and Butt (2011) propose three further classes that cut across this morphosyntactic distinction. Their classification is based on the observation that while about five different verbs can function as light verbs in N+V CPs, not all nouns are necessarily compatible with each of these verbs.

The examples in (1) represent just one class of N+V CPs. This class is compatible with all of the possible light verbs and is identified as a smallish class in Ahmed and Butt (2011). (1) shows the combination with three of these. In (1a) the noun *yad* 'memory' is combined with the light verb *kər* 'do'. In this case the subject must be ergative and the overall reading is one of an agentive, deliberate remembering. In (1b), in contrast, *laṛki* 'girl' is already taken to be in the state of remembering the story. The difference between (1b) and (1c) is one of stative vs. eventive, so that in (1b), *laṛki* 'girl' is already taken to be in the state of remembering the story (and not actively entering a state of remembering the story). In (1c) the light verb is the participial form of *ho* 'be' and essentially means 'become'.

Table 1 summarizes the results presented in Ahmed and Butt (2011). Class A refers to examples as in (1) and this type seems to encompass what is known as *psych-predications*, i.e, actions of remembering, thinking, feeling, etc. However, not all nouns are as versatile as *yad* 'memory'.

[5]The Urdu script, which is also provided in (1), is based on the Arabic script and is written from right to left.

One type, Class B in Table 1, does not allow the subject to be non-agentive. That is, it does not allow combinations with those light verbs that require a dative subject, cf. (1b–c). This pattern is illustrated with an example in (2) and this class was identified as by far the largest class in Ahmed and Butt (2011).

	Light Verb			
N+V Type	kar 'do'	hɛ 'be'	hU- 'become'	**Analyis**
CLASS A	+	+	+	psych-predications
CLASS B	+	−	+	only agentive
CLASS C	+	+	−	do not allow subject to be an undergoer

Table 1: Classes of nouns identified by Ahmed and Butt (2011)

(2) a. بلال نے مکان تمیر کیا
bɪlal=ne mɑkan tɑmir ki-ya
Bilal.M.Sg=Erg house.M.Sg.Nom construction.F.Sg do-Perf.M.Sg
'Bilal built a/the house.'

b. بلال کو مکان تمیر ہ/ہوا
*bɪlal=ko mɑkan tɑmir hɛ/hu-a
Bilal.M.Sg=Dat house.M.Sg.Nom construction.F.Sg be.Pres.3.Sg/be.Part-Perf.M.Sg
'Bilal built a/the house.'

The third class allows dative subjects in principle, but not when they are the undergoer of the action (cf. (1c)). An example with the noun *ɪntɪzar* 'wait' is given in (3). Other nouns that pattern similarly are *taslim* 'acceptance' and *bɑrdaʃt* 'tolerance'. This was again identified as a fairly small class in Ahmed and Butt (2011).

(3) a. بلال نے نادیا کا انتزار کیا
bɪlal=ne nadya=ka ɪntɪzar ki-ya
Bilal.M.Sg=Erg Nadya.F.Sg=Gen.M.Sg wait.M.Sg do-Perf.M.Sg
'Bilal waited for Nadya.'

b. بلال کو نادیا کا انتزار ہ/ہوا
bɪlal=ko nadya=ka ɪntɪzar hɛ/*hu-a
Bilal.M.Sg=Dat Nadya.F.Sg=Gen.M.Sg wait.M.Sg be.Pres.3.Sg
'Bilal is waiting/*waited for Nadya.'

As already mentioned, while the classes identified by Ahmed and Butt (2011) seem promising, the corpus work was done manually and was limited to a total of 45 nouns. The goal of our experiment was to expand the search space by using automatized methods, to thus extract information about a significant number of nouns and to be able to confirm, expand or revise Ahmed and Butt's proposal. In particular, understanding the semantic constraints on N+V CP formation in more detail would be welcome for further NLP applications.

2.2 Related work

Most related work on South Asian languages with the focus on an automatic extraction of complex predicates has been done for Hindi (Mukerjee et al., 2006; Chakrabarti et al., 2008; Sinha, 2009) and Bengali (Das et al., 2010; Chakraborty and Bandyopadhyay, 2010), with the predominant aim of identifying and extracting CPs from corpora or treebanks. Mukerjee et al. (2006) identify Hindi CPs based on the statistical correspondence between English verbs and Hindi multiword expressions (MWEs), using the parallel EMILLE corpus in a sentence-aligned version. The assumption here is that a verb in English will project onto a CP MWE in Hindi most of the times.[6] Bhattacharyya (2010), in presenting a WordNet for Hindi, lists the ∼100 most frequently occurring CPs. This is welcome information to be included in a WordNet. However, this static list ultimately does not do justice to either the overall productivity of CPs, nor does it provide details as to their syntactic/semantic patterning.

This paper has different aims than previous work in that we are not interested in merely identifying CP constructions in a large corpus, but are trying to understand more about their syntactic and semantic properties. To this end, we follow the classic assumption of Levin (1993) that semantic predicational classes can be identified on the basis of a study of the syntactic contexts the predicates occur in (cf. also Schulte im Walde 2009; Raza 2011).

3 Methodology

In this section, we describe the methodology used for harvesting CP candidates from a raw Urdu text corpus and for identifying classes among the candidate CPs via a simple bigram analysis in conjunction with techniques from visual analytics. We also discuss the problems we discovered to be associated with working with an unannotated "raw" corpus for a language like Urdu. Our overall results make a strong case for the prioritization of (mainly manual) high quality resource building — it seems that significant progress with entirely shallow methods cannot be achieved unless high quality, linguistically informed resources can be drawn upon.

3.1 Corpus

A prerequisite for our experiment is access to a large corpus for Urdu. No large corpus for Urdu (annotated or not) is publicly available to date (but see Urooj et al. 2012). We therefore decided to use a 7.9 million word corpus we have been harvesting from the BBC Urdu website[7] for a number of months. The corpus consists of news articles on various different topics, e.g., entertainment, multimedia, science, sports. These articles were automatically collected and parsed into raw text using the Perl HTML module.[8] Inspection of the corpus showed that the BBC Urdu script encoding is particularly clean and systematic in comparison to other Urdu newspaper sites. We therefore did not clean or preprocess the corpus with respect to punctuation, orthography or other normalization issues.

[6] One reviewer wonders whether we could not have used comparable/parallel data from Wikipedia to help us in identifying semantic classes among nouns. This is a potential line of avenue to pursue, but one that is not taken on lightly. Take the pair English-Urdu, for example. Most of the simple verbal predications do correspond to some kind of complex predication in Urdu, but different kinds, none of which are well understood. In addition, English also contains N-V constructions such as *take a bath* whose collocational properties are not very well understood either. It is unclear whether looking at two sets of data for which the collocational constraints within each set of data is not well understood will be able to yield useful results (or results that could not have been achieved manually with less effort).

[7] http://www.bbc.co.uk/urdu/

[8] http://search.cpan.org/dist/HTML-Parser/Parser.pm

3.2 Bigram Collocation Extraction

As a first pass, we went through the corpus and extracted all the bigrams containing the four verbs *kɑr* 'do', *ho* 'be', *hu-* 'become' and *rɑkʰ* 'put'. We decided to extend our set of light verbs beyond the ones used in Ahmed and Butt (2011) by including *rɑkʰ* 'put' in the hopes of arriving at a finer-grained picture of the distribution of the nouns in N+V CPs. We thus looked for all instances of these verbs (in all of their conjugated forms) and extracted them plus any word immediately preceding them. These bigrams were stored and their frequency was recorded. This procedure yielded four initial lists of bigrams, one for each of the four verbs *kɑr* 'to do', *ho* 'to be', *hu* 'to become' and *rɑkʰ* 'to put'.

A manual inspection of these lists revealed that while we were finding N+V CPs of the type we were looking for, most of the highly frequent bigrams were either junk or not the kinds of combinations we were trying to find. We also had an issue with low frequency in that many bigrams were recorded just once or twice. Since Urdu allows quite a bit of scrambling and also allows the nouns to be scrambled away from the light verbs, it was clear from the outset that we would not necessarily net all of the instances of N+V CPs that occur in the corpus. However, we were not prepared for the amount of false hits we did get.

Closer inspection of the bigram lists revealed that many of the false hits were due to certain case markers, conjunctions and pronouns, which all occur frequently before our set of verbs. This is actually to be expected, given the structure of the language (see section 3.4 for discussion). Since our set of verbs is very versatile in that they can not only act as light verbs in N+V CPs, but also function as main verbs and as auxiliaries, many of our top bigrams turned out to be verb-verb combinations of one type or another. Although these are all valid bigrams, they are unwanted noise in the context of our investigation. Further errors were introduced by punctuation and tokenization or white-space issues (see section 3.4).

As a consequence, the initial bigram lists, which consisted of 16033 possible combinations, were pruned. For one, all bigrams which appeared less than 6 times in the corpus were removed. We assumed that most of them were tokenization errors and other machine processing lapses. We also removed all those bigrams which had a negative X^2 value (see below). This reduction left us with only around 4500 bigrams. In a second step, we constructed a list of stop words from various sources. For one, we removed any bigrams containing problematic closed class items such as case markers, conjunctions and pronouns. For another, we used a verb list containing all the conjugated forms of about 700 simple verbs in Urdu (15285 verb forms in total, from the verb conjugator in Raza 2011) in order to remove all those bigrams which we could identify as verb+verb combinations via this verb list.

After applying these steps, we were left with just 2154 candidate bigrams from the original 16033 bigram possibilities drawn from the raw corpus. These candidate bigrams still contained problematic items (see section 4), but identifying and removing them at this stage would have involved intense manual inspection and labor. We therefore decided to work with this list of bigrams for further analysis by different methods.

As a first analysis step, the association strength between the bigram members was computed using the Chi-Square (X^2) measure. This ensures that the more often one of our light verbs occurs with a certain word compared to all other words, the stronger the association is and the higher the bigram is ranked among the group of bigrams. The statistics were computed by

means of the UCS toolkit.[9] We decided to use the X^2 association measure to determine the positive or negative association between the words in the bigram for two reasons. First, papers using comparatively sized corpora have reported encouraging results for similar experiments (Ramisch et al., 2008; Kizito et al., 2009; Hautli and Sulger, 2011). Second, initial manual comparison between bigram lists ranked according to all measures implemented in the UCS toolkit revealed the most convincing results for the X^2 test. Based on the ranking, we reduced the list of bigrams and discarded all bigram instances with either a negative X^2 or a frequency below 6 (see above). A negative X^2 value indicates a negative word association, i.e. the bigram members do not occur together very often in comparison to their frequency in other bigrams.

At this point, we still had our four separate lists of bigrams for the verbs *kɑr* 'do', *ho* 'be', *hu-* 'become' and *rɑkʰ* 'put', but the lists are now of ranked bigrams. Manual inspection revealed that the top items in these lists now did contain N+V combinations of the type that Ahmed and Butt (2011) were looking at, among them also some of the nouns that were discussed in that paper. Our various steps of filtering and ranking thus did allow us to extract a list of strongly associated lexical items of the right type. The next step was to proceed to an analysis of our extracted data. Since our interest lay in the determination of possible classes of N+V CPs, we decided to run a clustering algorithm on our data.

3.3 Automatic clustering

Based on the filtered and ranked lists of bigrams from section 3.2, we investigated the occurrences of words with light verbs across the different types of bigram combinations, i.e. we combined the information of the four lists into one list recording the light verb behavior of every single word. That is, we had different classes of words: words occurring with all four light verbs or words with only *kɑr* 'do' and *ho* 'be', *hu-* 'become' and *rɑkʰ* 'put' or words occurring with various combinations of these verbs. Table 2 shows an exemplary matrix of four nouns (*hasıl* 'achievement', *alan* 'announcement', *bat* 'talk' and *ʃuru* 'beginning' in that order) and their relative frequency of co-occurrence with the four light verbs, i.e. out of all occurrences of noun 1 (*hasıl* 'achievement') with one of the four light verbs, the relative frequency of it occurring with *kɑr* 'do' is 0.771.

ID	Noun	Rel. freq. with *kɑr*	Rel. freq. with *ho*	Rel. freq. with *hu*	Rel. freq. with *rɑkʰ*
1	حاصل	0.771	0.222	0.007	0.000
2	اعلان	0.982	0.011	0.007	0.000
3	بات	0.853	0.147	0.000	0.000
4	شروع	0.530	0.384	0.086	0.000

Table 2: Relative frequencies of co-occurrence of nouns with light verbs

Note that although the motivation for our experiment stemmed from an interest in N+V CPs, our bigrams in fact contain all kinds of POS in combination with our set of four verbs. This is a direct and unavoidable consequence of using an untagged "raw" corpus.

Based on the pattern of relative co-occurrence with the four light verbs, the results were clustered automatically. This was done using a data mining platform developed at the University

[9] http://www.collocations.de; see Evert (2004) for documentation.

of Konstanz: KNIME.[10] We used a k-means clustering algorithm in order to assign each noun to a cluster. We found that a number of five clusters minimized the average distance between the nouns and the cluster centers. However, both the table of numbers as illustrated in Table 2 as well as the clusters were difficult to evaluate in this form. We therefore decided to experiment with visualization techniques as they are currently being pioneered in computational linguistics (e.g., Dörk et al. 2008; Collins 2010; Mayer et al. 2010). Before moving to a discussion of these techniques in section 4, we take a step back and consider the language particular knowledge we had to rely on so far.

3.4 Discussion: Language Particular Issues

As mentioned in section 3.2, our first pass at bigram extraction resulted in a large number of false hits. Upon some reflection on the structure of Urdu, this was only to be expected.

Urdu is a language which is not particularly morphologically complex (in comparison with Native American or Australian Aboriginal languages, for example), but it does use a significant amount of morphology. One unfortunate feature from the perspective of NLP is that the same material is used for several different purposes. For example, *-a*, *-i* and *-e* are morphemes used to mark gender and number on nouns, adjectives, verbs, participles as well as the genitive case. Additionally, there is significant homonymy with respect to frequently used words. For example, one that had a significant impact is the perfect masculine form of *kɑr* 'to do', *kɪya*, which is written the same way as the interrogative pronoun *kya* 'what'. Another example is the perfect feminine form *ki* of *kɑr* 'do', which is written the same way as the complementizer 'that' as well as the feminine singular form of the genitive case marker.

The genitive case marker in general posed a problem. It is structurally a clitic (Butt and King, 2004) and is written as a separate word in Urdu. Given that it is a genitive case marker, it is naturally found adjacent to nouns. For us this meant that we extracted many bigrams which turned out to be collocations of the feminine singular genitive marker and a noun. We therefore decided to remove all instances of bigrams with *ki*. This meant that we probably lost many "good" bigram candidates, but we did not see a way of filtering out the "bad" instances of the genitive while keeping the good instances of the perfect feminine singular of *kɑr* 'do'.

As already mentioned, our set of four verbs *kɑr* 'do', *ho* 'be', *hu-* 'become' and *rɑkʰ* 'put' can be employed as simple verbs in Urdu as well and 'do', 'be', and 'become' can also function as auxiliaries. This meant that our initial bigram extraction netted many V+V sequences in which an item of our set of four verbs occurred as an auxiliary after a main verb. We dealt with this by employing a list of verbs along with all of their inflections that was constructed as part of the work done by Raza (2011). Since this encompassed a total of 15285 verb forms, having access to this already existing resource was invaluable.

With respect to verb conjugation, we also naturally normalized over the bigrams we looked for. That is, we looked for a total of 238 different forms of our set of four verbs, but normalized the different inflected versions to just the stem form for purposes of bigram storage. This is necessary for the task at hand as the inflectional variability would cause a futile co-occurrence analysis. However, it also means that we lose some information with respect to being able to understand whether a given bigram was really associated with a genitive marker or the feminine singular form of 'do', for example.

[10] http://www.knime.org

Finally, our initial list of bigrams contained instances of words with punctuation attached to them. We naturally removed these; however, there are other problems arising with respect to the Urdu script that are not dealt with as easily. For one, there are several different ways of spelling certain words in Urdu. One preprocessing step that we could have done is to run a normalization module across the corpus. However, this also requires specialized knowledge about the language/orthography and this source of errors was not large enough for us to take this step. Similarly, our bigram counts contain instances of words which have not been spaced correctly. The Urdu script is such that each letter has joined and non-joined versions. The conditions governing when to use a joined vs. non-joined version of a letter are fairly complex.

For example, take the third noun (*bat*) from Table 2. The first "letter" is a combination of a 'b' and an 'a' (the joined forms), the second letter is the non-joined version of 't'. In order to differentiate between spaces within words and spaces between two words, two different types of spaces have been defined. One is a normal space, the other is zero-width non-joiner (HTML code: ‌). However, authors are not always consistent in their use, thus giving rise to errors in the corpus, which we again cannot deal with without adding time-consuming manual inspection coupled with deep language-particular knowledge to the process. These errors thus remain in the bigram list we use for the analysis detailed in the next section.

4 Analysis via Visualization

Visual Analytics is based on the tight coupling of algorithms for automatic data analysis and interactive visual components (Thomas and Cook, 2005; Keim et al., 2010). The idea is to exploit human perceptive abilities to support pattern detection (Card et al., 1999). This involves the mapping of data dimensions to eight *visual variables* (Bertin, 1983), namely *position* (two variables x and y), *size*, *value*, *texture*, *color*, *orientation* and *shape*. While some numerical data dimensions can be mapped directly to one visual variable, other data features may require complex layout algorithms that project a combination of multiple data dimensions to a combination of visual variables, e.g., to the combination of the two powerful positional variables x and y. Finally, a data analyst should be able to manipulate the visual display interactively for different perspectives on the data, following Shneiderman's Visual Information-Seeking mantra "Overview first, zoom and filter, then details on demand" (Shneiderman, 1996).

The purposes of visualizing data are manifold. On the one hand, visualizations can be used to achieve an overview of complex data sets. On the other hand, the visualization approach can serve as a starting point for interactively exploring data, ideally detecting hidden patterns. In addition, new hypotheses can be generated and existing hypotheses verified.

The visualization employed in this paper uses the visual variable *color* and encodes the relative frequency of occurrences with different light verbs. This relative frequency is mapped onto a linear saturation scale, i.e. the higher the relative frequency, the more saturated the color. Figure 1 shows a reference visualization on the top, where the saturation is exemplified with the relative frequencies of 0.25, 0.5, 0.75 and 1.0 in the columns from left to right. Below the reference visualization, the relative frequencies of the data in Table 2 are encoded visually. The left-most color column shows the relative frequencies with *kɑr* 'do', the next columns show the relative frequencies with the light verbs *ho* 'be', *hu* 'become' and *rɑkʰ* 'put', respectively. The lexical item on the left hand side is the transliterated version of the Urdu input in Table 2, using the transliterator of Bögel (2012), following the transliteration scheme by Malik et al. (2010).[11]

[11] Short vowels are not encoded in the Urdu script, the transliterator puts a default "*" in places where a short

Figure 1: Visualization of the relative frequencies in Table 2

The visualization tool allows for an interactive exploration of the data in that the human investigator can scroll through the list of visually encoded N+V combinations, with the possibility of zooming in to get a detailed view on a restricted set of nouns, as well as zooming out to see a greater number of combinations and their overall behavior. Mousing over a colored box reveals the relative frequency of the N+V CP.

4.1 Visual Analysis: Round 1

The best clustering result for the clustering method described in section 3.3 was obtained with a specification of five clusters. These are numbered 0–4. Cluster 4 is a large cluster of about 1100 words. This cluster contained just those sets of words which occured only or almost only with *ho* 'be'. A manual inspection of this cluster showed that all of these words were either errors, false hits containing inflected verbs or words like 'what' and 'how many' or items like 'small', 'teacher', 'market', 'tomorrow' (the vast majority). The last category are items that occur in run-of-the-mill copula contexts like 'He is a teacher.'. This class is of no interest to us as there are no useful combinatory constraints to be discovered. We therefore decided to discard this cluster from our bigram list in its entirety.

Cluster 3 contained a small set of words that occurred with *hu* 'become' 100% of the time. All of these were false hits and were discarded. Similarly, Cluster 1 was a comparatively small cluster that consisted of words occurring mainly with *rɑkʰ* 'put'. Manual inspection showed that all the words ocurring with *rɑkʰ* 'put' a 100% of the time were false hits: the words were all objects of the main verb 'put' and not CPs of any kind. Again, we culled our bigram list to remove this set.

The other two clusters are more of a mixed bag. Cluster 2 has very large number of items that occur only with *kɑr* 'do'. Figure 2 shows the top part of Cluster 2 on the left. This contains nouns already identified as belonging to the ones combining in the type of N-V CP we are interested in. However, this class contains desired results as well as false hits which cannot be separated from one another on a visual (or purely numerical basis). Culling down this cluster would involve intense manual labor, which we decided to forego.

Similarly, Cluster 0 contains desired results as well as false hits. Again, these are difficult to discern visually, though there are clearer subclusters because this cluster contains those words which occur with all the four light verbs (at different levels of frequency). In Figure 2 *hasil* 'achievement', *ʃuru* 'beginning', and *koʃɪʃ* 'struggle' are the kinds of nouns we are interested in. On the other hand, *xɑtɑm* 'finished' and *jari* 'continued' are adjectives.

vowel is expected. The ambiguous characters *vao* and *ye* (consonant or vowel) are represented with ⟨vao⟩ and ⟨ye⟩, respectively. In case of entries with '???', the automatic transliteration could not find an adequate transliteration — this can be due to typing errors or unusual vowel/consonant combinations of English loan words. We needed to use a transliterated version of our bigram lists as the visualization tool we used could not deal with UTF-8 input.

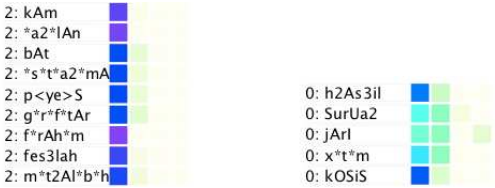


Figure 2: Visualization of the top of Cluster 2 (left) and Cluster 0 (right)

As part of our quick visual inspection, we also noted several types of unusual patterns found in the clusters, such as the ones illustrated in Figure 3 for the verb *ʊt̪ʰa* 'raise'. These often turned out to be false hits. The visualization thus allowed for a quick and easy identification of further errors. All unusual patterns that were identified as errors via this method of analysis were also removed from our bigram list. Detecting these types of errors would have been neigh impossible without an easily accessible visual representation.

Figure 3: Visually prominent pattern of a false hit

All in all, after the manual selection process, we were left with only 1090 instances of bigrams. This list still contains false hits, but removing these would take intensive manual labor. We therefore decided to rerun our automatic clustering algorithm on the (partially) cleaned data.

4.2 Visual Analysis: Round 2

The new clustering again yielded the best results with a specification of five clusters. The visualization shows that the cleaning and culling described in the previous section has had a positive effect, but that even the remaining 1090 bigrams still contain a good amount of false hits which would need to be culled on the basis of intense manual inspection. Nevertheless, the overall results are encouraging as different types of nouns are indeed being distributed over the different clusters.

As shown in Figure 4, Cluster 0 still contains *hasil* 'achievement' and *koʃɪʃ* 'struggle'. Items from the previous Cluster 2 also now appear here: *bat* 'talk', *ɪstemal* 'use' and *peʃ* 'presentation/happening'. On the other hand, *ʃuru* 'beginning', *xɑtɑm* 'finished' and *jari* 'continued' are now in Cluster 3, which mainly consists of adjectives, but also a few of the nouns we are interested in. While ADJ-V CPs were not the target of this paper, this cluster contains potentially useful information with respect to ADJ+V CPs.[12]

The same is true of Cluster 1: it consists of mainly adjectives, but there is a smattering of some of the nouns we are interested in as well as a number of spatial terms including postpositions.

Cluster 4 contains words which occur almost exclusively with *rɑkʰ* 'stay' and this set still does

[12]But note that Cluster 0 also contains adjectives, for instance *gɪrɪftar* 'arrested' in Figure 4.

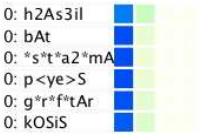


Figure 4: Visualization of the top of Cluster 0

not contain much that is of interest within the scope of this paper. On the other hand, Cluster 2 contains many nouns which occur in N-V CPs and indeed, contains many of the nouns that Ahmed and Butt (2011) identified as belonging to Class 2. A comparison of our results with that of Ahmed and Butt (2011) shows that while we did not find all of the nouns used in Ahmed and Butt, of the nouns that we did find most are Class B nouns (no dative subjects allowed). These are distributed over Clusters 0 and 2, while the Class A nouns (full range of light verb use) are found in Clusters 0, 1 and 3 (no Class C nouns were found in our study).

5 Results and Discussion

In this paper, we set out to see if we could find fine-grained information about the distributional properties of nouns in N+V CPs by extracting bigrams from a large "raw" corpus of Urdu. In addition to finding target instances of N+V combinations, our work has also resulted in lists of possible ADJ+V combinations and, in particular, the realization that P+V combinations are highly frequent. The prepositions involved are, for instance, 'front', 'back', 'on' and 'beside' and in combination with our set of verbs mean something like 'place on/beside/front/back' (with 'do' and 'put') or 'be/become on/beside/front/back' (with 'be' and 'become'). Given that these combinations are highly frequent, it is imperative that more be understood about P+V.

Unfortunately, this cannot be accomplished via our data because we actually have a problem with *data sparsity*. Manual inspection of some of the CPs we found showed that not all possible combinations of N/ADJ/P+V were in fact attested in the corpus. That is, our bigram lists will need to be complemented by manual native speaker work (traditional lexicography).[13]

We also find it remarkable that out of a corpus of 7.9 million words, we are at this point left with only 1090 bigrams (and are aware that a portion of these bigrams would still need to be culled as they represent errors or false hits). This problem of data sparsity could perhaps be ameliorated by using a different type of corpus, but we suspect that for our type of enterprise, data sparsity might be a problem regardless of how large or different a corpus is chosen — one cannot guarantee that all possible combinations will indeed be attested in any given corpus. On the other hand, one result that is very clear is that any word that occurred only in conjunction with *ho* 'be', *hu-* 'become' or *rak*^{h} 'put' was in fact not one that would occur in CPs. This means that one can exclude those words from candidate lists of nouns for N+V CPs in future work.

Our paper also makes contributions with respect to two methodological points. In order to be analyze our data more perspicuously, we experimented with new methods from *visual analytics*. This experimentation was successful as it allows for quick visual analysis of our data sets, which in turn also enables a non-labor-intensive way of further cleaning and culling our data. In

[13] One reviewer would like an indication of how many valid CPs we might have missed. Since we have no idea how many CPs were indeed contained in our corpus, we cannot say. And as N-V CPs are combinatorily dynamic, there are potentially infinitely many of these. All we can say is how many of the potentially infinitely many combinations we did find.

particular, the visual analysis allowed us to be able to quickly assimilate information about complex interrelationships in our data: which types of verbs does the word in question occur with with what level of frequency and how does this compare to other words in the list?

The visualization component proved to be an essential part of our data analysis. Another essential component turned out to be our language-particular knowledge and access to language-particular resources. The idea that we could work more or less "blindly" with a large corpus did not pan out. Rather, at every step we needed to be aware of language particular issues with respect to orthography, morphology and syntax. We could not have done our work without the use of a list of all the conjugated forms of 700 simple verbs (Raza, 2011) or a transliterator (Bögel, 2012) (needed to massage the input for the visualization component). We thus conclude that while we were partly successful in achieving what we set out to do, our work would have been able to be yield more precise results if we had access to standardized language-particular resources. On the other hand, our "blind" extraction of patterns yielded new insights into what kinds of CPs are highly frequent in newspaper corpora in addition to N+V CPs and which therefore need to be paid attention to with respect to NLP applications and the creation of language particular resources. This pertains in particular to P+V and ADJ+V combinations, about which not much is known either from a computational or from a theoretical perspective.

References

Ahmed, T. and Butt, M. (2011). Discovering Semantic Classes for Urdu N-V Complex Predicates. In *Proceedings of the international Conference on Computational Semantics (IWCS 2011)*, pages 305–309.

Ahmed, T., Butt, M., Hautli, A., and Sulger, S. (2012). A Reference Dependency Bank for Analyzing Complex Predicates. In *Proceedings of the Eight International Conference on Language Resources and Evaluation (LREC'12)*, pages 3145–3151, Istanbul, Turkey. European Language Resources Association (ELRA).

Bertin, J. (1983). *Semiology of Graphics*. University of Wisconsin Press.

Bhatt, R., Narasimhan, B., Palmer, M., Rambow, O., Sharma, D., and Xia, F. (2009). A Multi-Representational and Multi-Layered Treebank for Hindi/Urdu. In *Proceedings of the Third Linguistic Annotation Workshop*, pages 186–189, Suntec, Singapore. Association for Computational Linguistics.

Bhattacharyya, P. (2010). IndoWordNet. In *Proceedings of the Seventh Conference on International Language Resources and Evaluation (LREC'10)*, pages 3785–3792.

Bögel, T. (2012). Urdu – Roman Transliteration via Finite State Transducers. In *Proceedings of FSMNLP'12*.

Butt, M. (1995). *The Structure of Complex Predicates in Urdu*. Stanford: CSLI Publications.

Butt, M. (2010). The Light Verb Jungle: Still Hacking Away. In Amberber, M., Harvey, M., and Baker, B., editors, *Complex Predicates in Cross-Linguistic Perspective*, pages 48–78. Cambridge University Press, Cambridge.

Butt, M. and King, T. H. (2004). The Status of Case. In Dayal, V. and Mahajan, A., editors, *Clause Structure in South Asian Languages*, pages 153–198. Kluwer Academic Publishers, Berlin.

Card, S. K., Mackinlay, J. D., and Shneiderman, B., editors (1999). *Readings in Information Visualization: Using Vision to Think*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.

Chakrabarti, D., Sarma, V. M., and Bhattacharyya, P. (2008). Hindi Compound Verbs and their Automatic Extraction. In *Proceedings of COLING 2008*, pages 27–30.

Chakraborty, T. and Bandyopadhyay, S. (2010). Identification of Reduplication in Bengali Corpus and their Semantic Analysis: A Rule-Based Approach. In *Proceedings of the Workshop on Multiword Expressions: from Theory to Applications (MWE 2010)*, pages 72–75.

Collins, C. (2010). *Interactive Visualizations of Natural Language*. PhD thesis, University of Toronto.

Das, D., Pal, S., Mondal, T., Chakraborty, T., and Bandyopadhyay, S. (2010). Automatic Extraction of Complex Predicates in Bengali. In *Proceedings of the Workshop on Multiword Expressions: from Theory to Applications (MWE 2010)*, pages 37–45.

Dörk, M., Carpendale, S., Collins, C., and Williamson, C. (2008). VisGets: Coordinated Visualizations for Web-based Information Exploration and Discovery. *IEEE Transactions on Visualization and Computer Graphics (Proceedings of the IEEE Conference on Information Visualization (InfoVis '08))*, 14(6):1205–1213.

Evert, S. (2004). *The Statistics of Word Cooccurrences: Word Pairs and Collocations*. PhD thesis, IMS, University of Stuttgart.

Hautli, A. and Sulger, S. (2011). Extracting and Classifying Urdu Multiword Expressions. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Langauge Technologies (ACL-HLT '11): Student Session*, pages 24–29.

Hautli, A., Sulger, S., and Butt, M. (2012). Adding an Annotation Layer to the Hindi/Urdu Treebank. *Linguistic Issues in Language Technology*, 7(3):1–18s.

Hook, P. E. (1974). *The Compound Verb in Hindi*. The University of Michigan, Center for South and Southeast Asian Studies.

Humayoun, M. (2006). Urdu Morphology, Orthography and Lexicon Extraction. Master's thesis, Department of Computing Science, Chalmers University of Technology.

Keim, D. A., Kohlhammer, J., Ellis, G., and Mansmann, F., editors (2010). *Mastering The Information Age - Solving Problems with Visual Analytics*. Goslar: Eurographics.

Keim, D. A., Mansmann, F., Schneidewind, J., Thomas, J., and Ziegler, H. (2008). Visual Analytics: Scope and Challenges. In *Visual Data Mining: Theory, Techniques and Tools for Visual Analytics*, Lecture Notes in Computer Science, pages 76–91. Springer.

Kizito, J., Fahmi, I., Sang, E. T. K., Bouma, G., and Nerbonne, J. (2009). Computational Linguistics and the History of Science. In Dibattista, L., editor, *Storia della Scienza e Linguistica Computazionale*. FrancoAngeli.

Levin, B. (1993). *English Verb Classes and Alternations. A Preliminary Investigation*. The University of Chicago Press.

Malik, M. K., Ahmed, T., Sulger, S., Bögel, T., Gulzar, A., Raza, G., Hussain, S., and Butt, M. (2010). Transliterating Urdu for a Broad-Coverage Urdu/Hindi LFG Grammar. In *Proceedings of the Seventh Conference on International Language Resources and Evaluation (LREC'10)*.

Mayer, T., Rohrdantz, C., Butt, M., Plank, F., and Keim, D. (2010). Visualizing Vowel Harmony. *Journal of Linguistic Issues in Language Technology (LiLT)*, 4(2).

Mohanan, T. (1994). *Argument Structure in Hindi*. CSLI Publications.

Mukerjee, A., Soni, A., and Raina, A. M. (2006). Detecting Complex Predicates in Hindi using POS Projection across Parallel Corpora. In *Proceedings of the Workshop on Multiword Expressions: Identifying and Exploiting Underlying Properties (MWE '06)*, pages 28–35.

Ramisch, C., Schreiner, P., Idiart, M., and Villavicencio, A. (2008). An Evaluation of Methods for the Extraction of Multiword Expressions. In *Proceedings of the Workshop on Multiword Expressions: Towards a Shared Task for Multiword Expressions (MWE '08)*, pages 50–53.

Raza, G. (2011). *Subcategorization Acquisition and Classes of Predication in Urdu*. PhD thesis, University of Konstanz.

Schulte im Walde, S. (2009). The Induction of Verb Frames and Verb Classes from Corpora. In Lüdeling, A. and Kytö, M., editors, *Corpus Linguistics. An International Handbook*. Mouton de Gruyter.

Shneiderman, B. (1996). The Eyes Have It: A Task by Data Type Taxonomy for Information Visualizations. In *Proceedings of the 1996 IEEE Symposium on Visual Languages*, pages 336–, Washington, DC, USA. IEEE Computer Society.

Sinha, R. M. K. (2009). Mining Complex Predicates in Hindi Using a Parallel Hindi-English Corpus. In *Proceedings of the 2009 Workshop on Multiword Expressions, ACL-IJCNLP 2009*, pages 40–46.

Thomas, J. J. and Cook, K. A. (2005). *Illuminating the Path: The Research and Development Agenda for Visual Analytics The Research and Development Agenda for Visual Analytics*. National Visualization and Analytics Center.

Urooj, S., Jabeen, F., Adeeba, F., Parveen., R., and Hussain, S. (2012). Urdu Digest Corpus. In *Proceedings of the Conference on Language and Technology 2012*, Lahore, Pakistan.

Native Language Identification Using Recurring N-grams – Investigating Abstraction and Domain Dependence

Serhiy BYKH *Detmar MEURERS*
Seminar für Sprachwissenschaft, Universität Tübingen
{sbykh,dm}@sfs.uni-tuebingen.de

ABSTRACT
Native Language Identification tackles the problem of determining the native language of an author based on a text the author has written in a second language. In this paper, we discuss the systematic use of recurring n-grams of any length as features for training a native language classifier. Starting with surface n-grams, we investigate two degrees of abstraction incorporating parts-of-speech. The approach outperforms previous work employing a comparable data setup, reaching 89.71% accuracy for a task with seven native languages using data from the International Corpus of Learner English (ICLE). We then investigate the claim by Brooke and Hirst (2011) that a content bias in ICLE seems to result in an easy classification by topic instead of by native language characteristics. We show that training our model on ICLE and testing it on three other, independently compiled learner corpora dealing with other topics still results in high accuracy classification.

TITLE AND ABSTRACT IN GERMAN

Muttersprachenerkennung mittels rekurrenter N-Gramme – Untersuchungen zur Abstraktion und Domänenabhängigkeit

Die Muttersprachenerkennung befasst sich mir der Erkennung der Muttersprache eines Autors auf der Basis eines Textes, der von diesem Autor in einer Zweitsprache verfasst worden ist. In der vorliegenden Arbeit diskutieren wir die systematische Verwendung rekurrenter N-Gramme aller Längen als Features für das Trainieren eines Muttersprachen-Klassifizierers. Beginnend mit oberflächenbasierten N-Grammen, untersuchen wir zwei Stufen der Abstraktion unter Verwendung von Wortarten. Unser Ansatz liefert eine Klassifikationsgenauigkeit von 89.71% für Texte aus dem International Corpus of Learner English (ICLE) mit sieben unterschiedlichen Muttersprachen und übertrifft somit die bisherigen Ergebnisse auf vergleichbaren Daten. Ferner untersuchen wir die Behauptung von Brooke und Hirst (2011), dass inhaltliche Aspekte des ICLE zu einer einfacheren Klassifikation der Texte nach dem Thema anstatt nach der Muttersprache führen könnten. Wir zeigen, dass ein auf ICLE Daten trainiertes Modell auch bei Tests auf drei unabhängig erstellten Lernerkorpora eine hohe Klassifikationsgenauigkeit ermöglicht.

KEYWORDS: Native Language Identification, Author Profiling, Text Classification, Second Language Acquisition, Learner Corpora.

KEYWORDS IN GERMAN: Muttersprachenerkennung, Autoren-Profiling, Textklassifkation, Zweitspracherwerb, Lernerkorpora.

Proceedings of COLING 2012: Technical Papers, pages 425–440,
COLING 2012, Mumbai, December 2012.

1 Introduction

Inferring characteristics of an author by automatically analyzing that author's texts is a task that is increasingly drawing attention in recent years. Traits such as gender, age, level of education or native language are some of the properties targeted thus far (e.g., Koppel et al., 2005; Estival et al., 2007; Wong and Dras, 2009).

The work presented in this paper examines one particular characteristic, namely the author's *native language*, with the task being to infer it from a text written in a second language. So we explore the task of *Native Language Identification (NLI)*, which is of interest for a number of reasons. The impact of one's native language on a second language is studied in *Second Language Acquisition* (SLA) research, aimed at understanding how languages are acquired and how language works in general. Of particular relevance here is the notion of Transfer: *"Transfer is the influence resulting from similarities and differences between the target language and any other language that has been previously [...] acquired."* (Odlin, 1989, p. 27). Given the increasing availability of second language corpora with different native languages as well as powerful classification and evaluation techniques, it becomes viable to empirically explore and verify hypotheses regarding the existence and nature of L1 Transfer. Complementing the conceptual relevance for SLA, NLI also is of practical relevance for applications such as systems for profiling phishing emails (Estival et al., 2007) or in the context of learner modeling for intelligent language tutoring systems (Amaral and Meurers, 2008).

NLI started to attract interest less than ten years ago (Koppel et al., 2005), so the area still is quite young, with fundamental questions waiting to be addressed: Is the L1 Transfer effect strong and distinctive enough across domains to support an automatic classification with a reasonable degree of reliability for the typical available document lengths? Which language properties are the most appropriate ones to use as classifier features for the given task and can they reliably be identified? How well can a surface-based approach fare in the task and what is the effect of abstracting away, e.g., to distributional classes such as parts-of-speech (POS)?

In this paper, we consider the NLI task as a text classification problem with the different native languages as the classes. Inspired by the variation n-gram approach to corpus annotation error detection (Dickinson and Meurers, 2003, 2005; Boyd et al., 2008), we will follow a data-driven approach based on *recurring n-grams* as features in a machine learning setup capable of handling large feature spaces. The aim of our work is to contribute a piece to the overall puzzle to solve, starting with a particular take on surface features, recurring word-based n-grams of any size, and exploring the effect of incrementally introducing POS as abstractions. In the second part of the paper, we then explore the generalizability of our results across corpora.

2 Related Work

Koppel et al. (2005) used a subset of the first version of the *International Corpus of Learner English (ICLE)* (Granger et al., 2002) as data set. The ICLE corpus consists of essays written by non-native English speakers at a similar level of English proficiency, namely higher intermediate to advanced. Koppel et al. included texts for five native languages: Bulgarian, Czech, French, Russian and Spanish. Each native language was represented by 258 essays. They used a *Support Vector Machine* (SVM) as classifier and defined features based on the occurrence of function words, character-based n-grams, rare POS bi-grams as well as some error types (e.g., certain spelling errors). Testing was performed using 10-fold cross-validation. The best classification accuracy of 80.2% was obtained using all of the mentioned features combined.

Tsur and Rappoport (2007) replicated Koppel et al. (2005) and investigated the hypothesis that the choice of words in the second language is strongly influenced by the frequency of native language syllables. In support of their hypothesis, the authors report that an approximation using character bi-grams alone allows classification accuracy of about 66%. Since the corpus contains learner essays on several different topics, they also investigated whether the classification with such surface features is influenced by a content bias. Using a variant of the *Term Frequency - Inverted Document Frequency* content analysis metric, they conclude that if a content bias exists in the corpus, it only has a minor effect.

Estival et al. (2007) used a corpus of English emails as data incorporating three native languages, namely English, Arabic, Spanish, and considered a range of different demographic as well as psychometric traits including the native language for author profiling purposes. They used a wide range of features at different levels: character-based features such as frequency of punctuation marks, lexical features such as function words as well as POS, and some features at the structural level such as paragraph breaks. Using *Information Gain* as feature selection technique and *Random Forest* classification, they obtained an accuracy of 84.22%.

Wong and Dras (2009) used the second version of the ICLE corpus (Granger et al., 2009) as data and compiled a data set consisting of seven native languages, namely Bulgarian, Czech, French, Russian, Spanish, Chinese and Japanese, each represented by 70 essays for training and 25 essays for testing (plus 15 additional essays for development). On the one hand, they employed lexical features, such as function words, frequently used character-based uni-, bi-, tri-grams as well as rare and most frequently used POS bi- and tri-grams. On the other hand, they used three syntactic error types as features: misuse of determiners as well as subject-verb and noun-number disagreement. Using an SVM classifier they obtained an accuracy of 73.71%. Extrapolating to a larger training set, they argue that this result is consistent with the findings reported by Koppel et al. (2005). However, the syntactic features used in their study did not improve the results obtained by employing lexical features alone.

Wong and Dras (2011) extended their previous work by investigating more general syntactic features compiled on the basis of parse trees, namely horizontal slices as well as cross-sections of parse trees. These features were used along with the set of lexical ones of Wong and Dras (2009). Using the same data set as Wong and Dras (2009) and a *Maximum Entropy* classifier, they obtained a classification accuracy up to 81.71%, showing that incorporating more sophisticated syntactic features can improve the results.

Brooke and Hirst (2011) conducted several experiments employing two different corpora, namely the ICLE and the Lang-8. The second corpus was compiled by the authors themselves based on the data available on `http://lang-8.com`. This web site contains short personal journal entries of different kinds (personal narratives, requests for translations of particular phrases, etc.), which are posted by English learners in order to obtain feedback from native speakers. Compared to the ICLE corpus, there is a disproportionately high number of contributors from Eastern Asia, the level of English proficiency seems to be significantly lower, and little is known about the context of the writing for Lang-8 (e.g., there is no specification of time or resources used). To obtain texts from Lang-8 which are comparable in size to those in ICLE, Brooke and Hirst (2011) created texts consisting of multiple Lang-8 entries. In their computational approach, they use character, word, and POS-based uni- and bi-grams (excluding proper nouns in case of word-based n-grams) as well as some function words as features. Based on a dataset from ICLE and Lang-8 consisting of seven native languages, namely Chinese, Japanese, French,

Spanish, Italian, Polish and Russian, with each of them represented by 200 texts from each of the two corpora, they conducted experiments using an SVM classifier in a single-corpus evaluation (using 10-fold cross-validation) and a cross-corpus evaluation (training on the one corpus, testing on the other). The single-corpus evaluation on ICLE data yielded an accuracy of 93.8% using all the features together, yet only 25% when training and testing on the Lang-8 data. The results of cross-corpus evaluation were very low, at 15.7% to 22.9%. Based on these results, Brooke and Hirst (2011) argue that a strong content bias is present in ICLE, allowing an easy classification by topic instead of by native language. However, it remains unclear whether the poor Lang-8 results are not due to the properties of this specific corpus, which seems to be highly heterogeneous and incoherent, and whether the poor cross-corpus evaluation results are of general importance or due to the nature of the Lang-8 corpus. Brooke and Hirst then explore the usefulness of artificial learner corpora, which they compiled using machine translation of native language data. The results yield up to 67% in a setup with two native languages. Brooke and Hirst (2012) extend their previous work and show that using automatically translated word bi-grams in combination with a new L1 Transfer metric yields up to 48.3% in a setup with four native languages when tested on ICLE data. The accuracies are far below those reported previously, but the approach promises a low content bias.

3 Data

For our first, core study we use a subset of the *International Corpus of Learner English* (ICLE v. 2; Granger et al., 2009). The overall ICLE corpus consists of 6,085 essays written by English learners of 16 different native language backgrounds. They are at a similar level of English proficiency, namely higher intermediate to advanced and of about the same age. Following the setup of Wong and Dras (2009), we randomly select a set of essays from the same seven native languages – namely, Bulgarian, Czech, French, Russian, Spanish, Chinese, and Japanese – and we use the same data split with 70 essays for training and 25 essays for testing for each of the languages, resulting in a total of 490 essays for training and 175 for testing. As in Wong and Dras (2009), we only included essays between 500 and 1000 words in length. We tokenized the essays and removed all punctuation marks, special characters and capitalization. Thus each essay is represented as an array of lower-case words.

To get a better sense of how well our approach performs, we conducted ten experiments. We select the data for each of them randomly from the full set of ICLE essays within the mentioned length range. We thus are able to observe the variance of the results based on ten randomly selected samples from the overall corpus subset matching the described criteria. We first describe one of the ten experiments in detail and then turn to the overall ten experiments.

4 Features

Different from previous research, in this study we explore *recurring n-grams of all occurring lengths* as classifier features. By *recurring* we here mean all n-grams that occur in at least *two* different essays of the training set d (the test set is held out, i.e., not considered for determining the features). *Of all occurring lengths* means all recurring n-grams up to the maximum possible n value occurring in d, i.e., all n-grams with $1 \leq n \leq max_n(d)$.

On the one hand, we use recurring *word-based* n-grams directly, i.e., the surface forms. On the other hand, we explore two different classes of recurring *POS-based* n-grams as a generalization, based on a POS tagged version of the corpus using the PennTreebank tagset (Santorini, 1990). In sum, we define our features based on the following three classes of recurring n-grams:

Word-based n-grams (word n-grams): strings of words, i.e., the surface forms

- $n = 1$: *analyzing, attended, ...*
- $n = 2$: *aspect of, could only, ...*
- $n = 3$: *is capable of, the assumption that, ...*
- ...

POS-based n-grams (POS n-grams): *all words* are converted to the corresponding POS tags

- $n = 1$: *nnp, md, nns, vbd, ...*
- $n = 2$: *nns md, nn rbs, nn rbr, cc wdt, vbp jjr, vbp jjs, ...*
- $n = 3$: *cd wdt md, vbp nn md, dt rbr cc, nn jj in, ...*
- ...

Open-Class-POS-based n-grams (OCPOS n-grams)[1]**:** *nouns, verbs, adjectives* and *cardinal numbers* are converted to the corresponding POS tags

- $n = 1$: *far, vbz, much, jj, ...*
- $n = 2$: *nn whenever, jj well, jjs vbd, vbg each, nn always, ...*
- $n = 3$: *vbp currently jj, only to the, cd vbz jj, vb if there, ...*
- ...

We explore the whole range of n values as well as all possible $[1, n]$ intervals. Figure 1 depicts the counts of *different n-grams* for each n (for uni-grams, bi-grams, tri-grams, etc.) and Figure 2 for each $[1, n]$ interval (for uni-grams alone, uni-grams and bi-grams together, uni-grams, bi-grams and tri-grams together, etc.). There are large differences in terms of feature counts, depending on the particular n-gram class and the value of n used. The figures show that increasing the number of different POS tags leads to more possible different features (up to about 160,000 in our setup). The reason for that is the ability of POS to bridge some break points in the word sequences (i.e., places where different words occur, thus ending a recurrent surface n-gram) and hence to lead to more longer n-grams. Thus the n-grams including POS tags may also reach higher n values: For the word-based n-grams $max_n(d) = 29$, whereas POS-based n-grams reach $max_n(d) = 30$ in the training set used.

As expected, the feature counts fall rapidly as the n value passes a certain (n-gram class dependent) threshold (see Figure 1). Longer n-grams may potentially contain some specific information not contained in the shorter ones – they may capture, e.g., transliterations of native idioms (Milton and Chowdhury, 1994). So we do not discard any features a priori. The aim is to investigate up to which value of n the n-grams may be worth considering despite being rare.

We use binary feature vectors as classifier input, i.e., each essay is represented by a vector containing $\{0, 1\}$ values. If an essays contains a particular n-gram, then the corresponding value in the vector is 1, and 0 otherwise. Since the n-gram frequencies (especially in case of the longer ones) are rather low, we consider such a representation to be a reasonable simplification.

5 Tools

To extract all recurring n-grams, we implement a dynamic programming algorithm collecting all n-grams of length n based on the n-grams collected for $n-1$. The algorithm terminates once no n-grams for a given length can be found in the given data. To obtain the n-gram classes incorporating POS tags, we used the *OpenNLP* POS-tagger (`http://opennlp.apache.org`).

[1] Similar representations are also used by Baroni and Bernardini (2006) for the identification of "translationese".

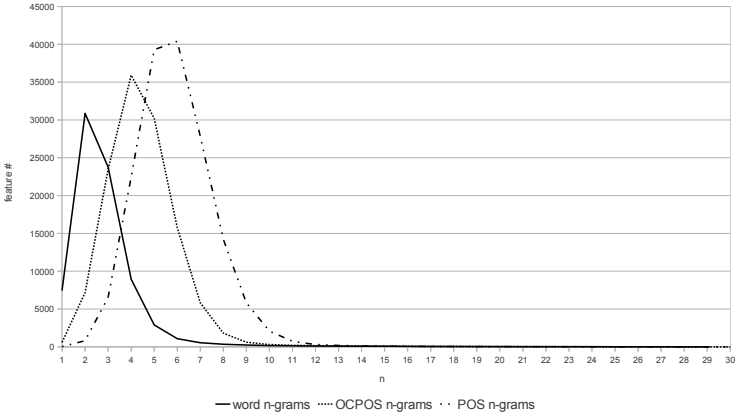


Figure 1: Feature counts for single n n-gram settings for the single ICLE sample

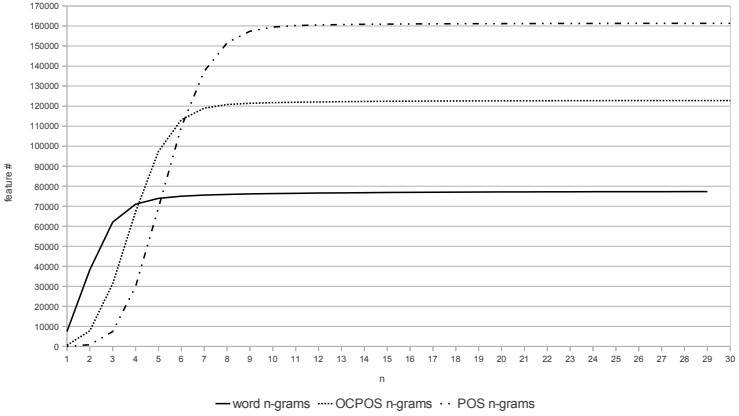


Figure 2: Feature counts for $[1, n]$ n-gram settings for the single ICLE sample

To choose the classifier to use, we conducted several preliminary tests employing different machine learning tools. We explored using *TiMBL* (Daelemans et al., 2007), which provides an implementation of the k-NN algorithm, incorporating a range of distance metrics. We then tested different Support Vector Machines (SVMs), which are well-known for their ability to handle large feature sets: *WEKA SMO* (Platt, 1998; Hall et al., 2009), *LIBSVM* (Chang and Lin, 2011), and *LIBLINEAR* (Fan et al., 2008). In our trials, the *LIBLINEAR* classifier yielded by far the best results and was in addition usually faster than the others as well. Hence, we employ the *LIBLINEAR* classifier in our study.

6 Results

The classification results for all feature settings are presented in Figures 3 and 4.

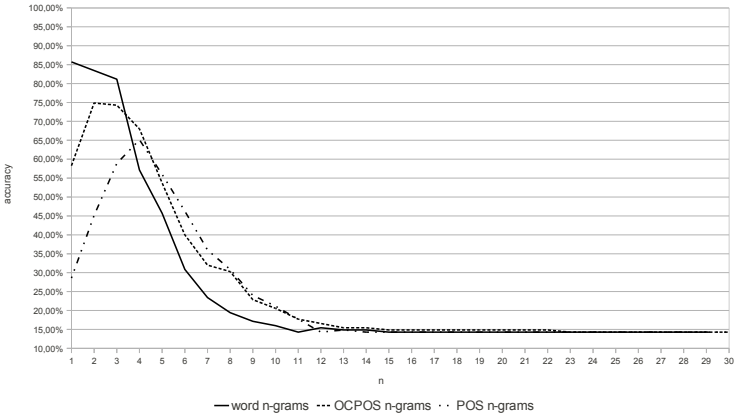


Figure 3: Results for single n n-gram settings for the single ICLE sample

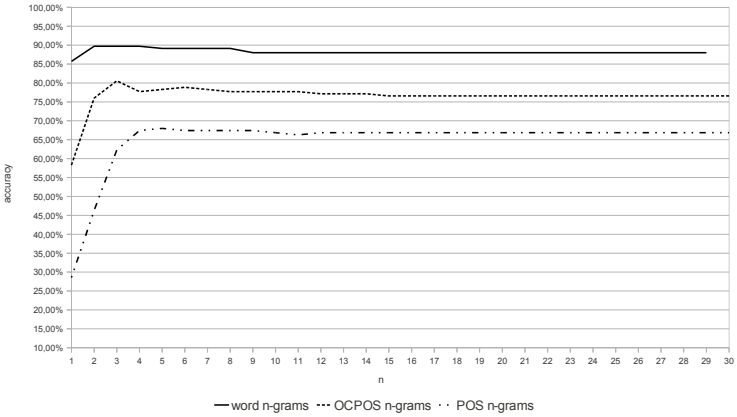


Figure 4: Results for $[1, n]$ n-gram settings for the single ICLE sample

Figure 3 shows the classification accuracy for all n values of the n-grams separately (i.e., for uni-grams, bi-grams, tri-grams, etc.), whereas Figure 4 depicts the classification accuracy for all $[1, n]$ intervals (i.e., for uni-grams alone, uni- and bi-grams together, uni-, bi-, tri-grams together, etc.). There are seven different native languages as classes, each represented by an equal number of essays, so 14.29% is the random baseline against which to interpret the results.

Best Accuracy Range The highest accuracy achieved by our recurring n-gram approach is 89,71% using word-based n-grams with intervals from $[1,2]$ to $[1,4]$. This is 16% higher than the best result reported by Wong and Dras (2009) and about 8% higher than that reported by Wong and Dras (2011) on a comparable data set. Brooke and Hirst (2011) reported a slightly better result, 93.8% for seven native languages, but as discussed in Section 2 they used more data and a different data split.

The confusion matrix in Table 1 shows the distribution of correctly classified as well as misclassified samples for each of the native languages. The performance on the different native languages is generally comparable; only the result for Russian is slightly below the others.

	BG	**CN**	**CZ**	**FR**	**JP**	**RU**	**SP**
BG	23	0	0	0	0	2	0
CN	0	24	0	0	1	0	0
CZ	0	0	23	1	0	1	0
FR	1	0	0	22	0	0	2
JP	0	0	1	0	24	0	0
RU	1	0	3	1	1	19	0
SP	1	1	0	1	0	0	22

Table 1: Confusion matrix for the best result for the single ICLE sample: 89,71%, word-based n-grams, $[1,2]$; BG: Bulgarian, CN: Chinese, CZ: Czech, FR: French, JP: Japanese, RU: Russian, SP: Spanish

However, there are clear differences in terms of accuracy between the n-gram classes utilized in this study. As mentioned above, the best result is obtained using pure surface forms, the word-based n-grams. The more different POS tags are incorporated, i.e., the bigger the step from the surface to the more general forms, the lower the accuracy. The information loss involved in the abstraction thus outweighs the broader applicability. The best results are presented in detail in Table 2.[2]

	n Intervals			**Single n**		
Features	$[1,n]$	**Accuracy**	**Feature #**	n	**Accuracy**	**Feature #**
word n-grams	2	89.71%	38,300	1	85.71%	7,446
OCPOS n-grams	3	80.57%	31,263	2	74.86%	7,176
POS n-grams	5	68.00%	69,139	4	65.14%	22,462

Table 2: Best results for the single ICLE sample

Table 2 shows that POS-based n-grams, i.e., features at the highest generalization level, yield about 13% lower accuracy than the Open-Class-POS-based n-grams, and the latter are performing about 9% worse than word-based n-grams. There is a gap of about 22% between the surface-based and the most generalized n-gram representation used in our study. However, even the most general POS-based n-grams still yield a result of 68%, which is reasonably high considering the baseline of 14.29%. The accuracy of 80.57% obtained using Open-Class-POS-based n-grams is in line with the best results published for a comparable data set.

[2] If more than one setting per feature class yields the same best accuracy, only the lowest n or $[1,n]$ interval is listed.

Different n Values Using intervals of n always leads to better results than using n-grams of a particular single n value alone (see Figures 3 and 4). One can also see that the more POS generalization is incorporated, the longer n-grams are needed to obtain the best results. In this study, the accuracy benefited from n-grams up to $n = 5$. Thus n-grams with $n > 3$, which are generally not considered in the related research, are not a priori useless.

The longer n-grams in the range of $6 \leq n \leq 10$ seem to be too sparse to improve on the accuracy obtained by intervals of shorter n-grams, at least in the data used in this study. There are a lot of different n-grams in that range, especially for n-grams with POS incorporated (see Figure 1), but the impact of lots of different features, with each occurring only in a few essays, seems to be very limited. Moreover, using them in intervals with n-grams of lower n values usually decreases the accuracy (see Figure 4). Thus they seem to introduce some noise into the feature set. However, increasing the size of the data set or incorporating more sophisticated generalizations may still allow such n-grams to become useful.

Finally, "very long" n-grams, i.e., n-grams with $n > 10$, usually encode a few, predetermined phrases, such as the wording of the topic the essay is about, or consist of some other copied passages. Hence, they are unlikely to be relevant for the given NLI task.

Reliability of the Findings Since the results described above are based on a single experiment, one may wonder, how generalizable those findings are. As mentioned in Section 3, we thus conducted nine further experiments. Summing up the results of the ten experiments, we computed the *mean accuracy* values along with the *Sample Standard Deviations (SSD)*. Given that the $max_n(d)$ value varies for the ten training sets, one cannot average over all n for all of the experiments. But as discussed in the previous paragraph, n-grams with $n > 10$ are unlikely to be useful for the purposes of the given task. Hence, we report the accuracy results for the $1 \leq n \leq 10$ range. Figure 5 shows the results for $[1, n]$. Overall, the means curves are very similar to the curves we presented in Figure 4.

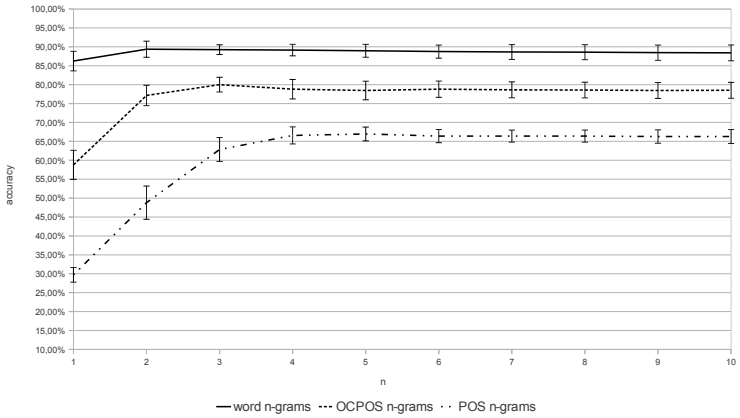


Figure 5: *Mean accuracy* and *SSD* for $[1, n]$ n-gram settings for the ten ICLE samples

The overall best outcomes are shown in Table 3. The best mean accuracy result of 89.37% is yielded by the same setting, namely by the word-based n-grams using the $[1, 2]$ interval.

	n **Intervals**			**Single** n		
Features	$[1, n]$	**Mean accuracy**	**SSD**	n	**Mean accuracy**	**SSD**
word n-grams	2	89.37%	2.12%	1	86.23%	2.59%
OCPOS n-grams	3	80.00%	1.94%	2	73.71%	2.68%
POS n-grams	5	66.97%	1.82%	4	60.91%	3.38%

Table 3: Best mean accuracy results for ten ICLE samples

This best mean accuracy over ten experiments is only 0.34% lower than the corresponding best result from the single experiment described in the *Best Accuracy Range* paragraph of the current section. The SSD with values around 2% for the best performing settings indicates that there is little variance among the experiments.

Discussion The ICLE contains essays from a range of topics, so one may wonder about the impact of the contents on the native language identification. Using only essays of the same topic would in principle be preferable, but it would significantly reduce the amount of data available. As mentioned in Section 2, Tsur and Rappoport (2007) argued that such a content bias is rather marginal for the subset of the ICLE they used. In contrast, the findings of Brooke and Hirst (2011) suggested a high topic bias in the ICLE data. In order to obtain more independence from the content of an essay, there is a clear need for some abstraction away from the surface encoding form and meaning together. Yet, the features in our study with the highest level of generalization and thus probably the lowest topic bias, recurring POS-based n-grams, provide results about 22% below those purely based on surface forms. A combination of surface and generalized forms may be a reasonable middle ground. In that light, the *Open-Class-POS-based n-grams* appear attractive since they replace many of the topic-specific meaning distinctions with POS-tags. They are less tied to the meaning than word-based n-grams, but still yield high accuracy with relatively low feature counts in the best performing n range. At the same time, Brooke and Hirst (2011) observe a comparable drop for word and POS-based features in cross-corpus evaluation with the Lang-8 corpus, and Golcher and Reznicek (2011, p. 31) show that POS n-grams still contain information relevant to topic classification for the German learner corpus FALKO. More research thus is needed to verify which features are sufficiently general and applicable across corpora. We address this issue in the next section.

7 Investigating the cross-corpus generalizability of the results

To address the question whether the models trained and evaluated on the ICLE corpus generalize to other learner corpora, we conducted a second set of experiments.

Data In this second study, we use four different learner corpora. Complementing the ICLE introduced above, we use the NOCE, USE and HKUST corpora compiled by independent research teams.

The *Non-Native Corpus of English / NOCE* (Díaz Negrillo, 2007, 2009) is an English learner corpus consisting of mainly argumentative essays on several topics written by Spanish native speakers. The data was collected at the University of Granada and the University of Jaén using texts by undergraduates pursuing an English degree. The corpus contains 1,022 essays.

The *Uppsala Student English Corpus / USE* (Axelsson, 2000, 2003) is a corpus of learner English consisting of texts written by Swedish students at the Department of English at Uppsala University. The texts contained in the corpus are essays written as part of the regular curriculum and cover several topics of different genres, e.g., argumentation, reflection, literature course assignment, etc. The corpus contains 1,489 essays. Since the essays from the other corpora used in this study are mostly argumentative, to obtain comparable data in terms of the text properties we use only the argumentative subset of the corpus (from the first term). This USE subcorpus consists of 344 essays.

The *Hong Kong University of Science and Technology English Examination Corpus / HKUST* (Milton and Chowdhury, 1994) is an English learner corpus containing texts written by Chinese native speakers. The version of the corpus we are using consists of 1,100 argumentative essays on different topics collected 1992 during the public matriculation examination, which is taken each year by students leaving secondary school. For the present work, we took a 8% random sample of the whole corpus, consisting of manually tagged 77 essays as described in Milton and Chowdhury (1994, p. 128).

As preprocessing, we removed all types of meta-information and annotation contained in the learner corpora (personal information about the author of the text such as the age or the native language, topic tags, error annotation, etc.) as well as all punctuation marks, special characters and capitalization, and we tokenized the essays. Hence, as in the first study each text is represented as an array of lower-case words.

Based on the data described above, we explore the NLI task using a setup with three native languages: Spanish, Swedish and Chinese. First, we compile *two separate test sets*. The first test set consists of randomly selected 70 essays per native language from ICLE. To compile the second test set, we randomly select 70 essays per native language correspondingly from HKUST and USE and 140 essays from the NOCE corpus. Since the NOCE essays tend to be shorter than the other ones, we merge the 140 essays pairwise to obtain 70 texts of a size comparable to the essay size from the other corpora. The texts on average contain 620 words. Second, we compile *ten separate training sets*. Each training set consists of randomly selected 140 essays per native language from the overall ICLE corpus (without the essays selected for the ICLE test). Thus we obtain ten separate training sets with 420 essays each, randomly selected from the ICLE corpus, and two separate test sets with 210 texts each, one compiled using ICLE alone and another compiled using NOCE, USE and HKUST.

This setup allows us to perform ten *single-corpus* evaluations (i.e., training and testing on the same corpus) on the ICLE data alone as well as ten *cross-corpus* evaluations (i.e., training on the one corpus and testing on another corpus) using ICLE data for training and NOCE, USE, HKUST data for testing. With ten separate ICLE training sets, we are able to build ten different classifier models and to observe the variance in the generalizability of the patterns learned on different ICLE subsets. We thus are able to observe the generalizability of the ICLE patterns to other corpora in direct comparison to ICLE itself.

Results Based on the ten different training sets, we conducted tests for each $[1, n]$ n-gram interval with $1 \leq n \leq 10$ using the two best performing n-gram classes (i.e., word- and OCPOS-based n-grams as features), and performed both a single-corpus evaluation and a cross-corpus evaluation. We thus obtained 400 separate accuracy values overall (10 training sets $\cdot$ 2 n-gram classes $\cdot$ 10 n-gram intervals $\cdot$ 2 evaluation types).

Figure 6 sums up the results by depicting the *mean accuracy* values on the two test sets obtained using ten different training sets for both n-gram classes and each of the ten n-gram intervals along with the random baseline. Since in this set of experiments there are three different native language classes, each represented by an equal number of essays, we obtain 33.33% as a random baseline against which to interpret the results.

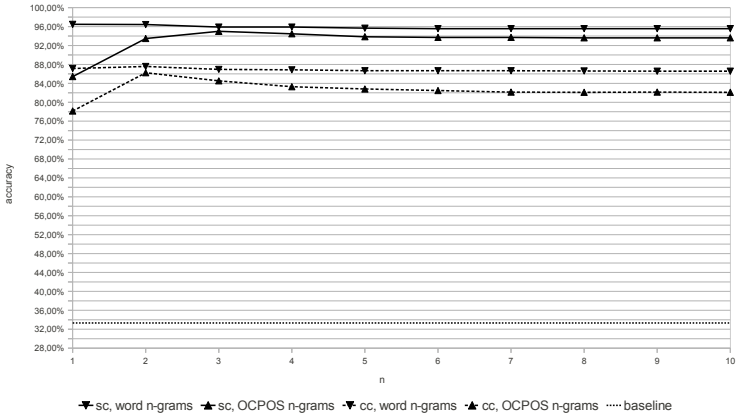


Figure 6: *Mean accuracy* for $[1, n]$ n-gram settings for the ten ICLE training sets (sc = single-corpus, cc = cross-corpus evaluation)

We left the SSD bars out of Figure 6 to keep it readable, but it naturally is interesting to consider the variance. Figure 7 shows the single- and cross-corpus accuracies for the word-based n-grams from Figure 6 together with the corresponding SSD. Figure 8 presents the same for the OCPOS-based n-grams. We see that in both figures the variance is low, with the cross-corpus evaluation showing slightly higher SSD values as expected.

Table 4 shows the best accuracies for both feature classes along with the corresponding SSD values obtained on the two different evaluation types as well as the corresponding n intervals. Though the best performing n-gram intervals differ for both feature classes on single-corpus evaluation, in the cross-corpus evaluation *recurrent bi-grams* perform best for both.

At the end of Section 6, we hypothesized that the more abstract OCPOS-based n-grams may perform better than the surface-near word-based ones in cross-corpus evaluation. However, the accuracies obtained using word-based n-grams are on average as good or better than the ones obtained using OCPOS-based n-grams (see Figure 6 and Table 4). Apparently people with different native language backgrounds make lexical choices which are indicative across a range of topics. A first qualitative analysis points to the use of predicates such as *get*, *take*, *choose*, *make use of*, *consider*, *be able to*, *understand*, or *suggest*. A precise characterization of the nature of this lexical material seems relevant to investigate in future work.

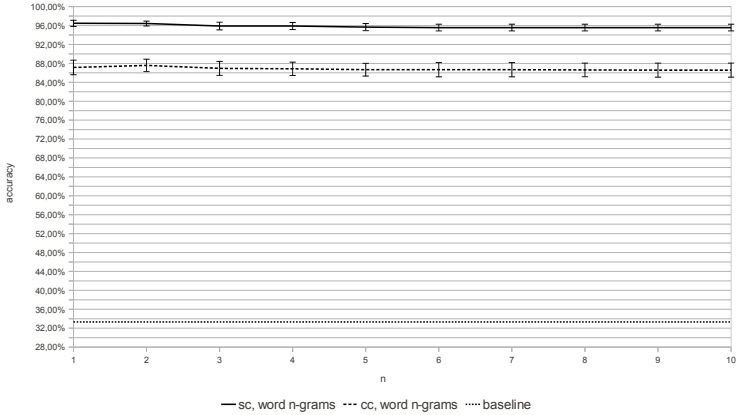


Figure 7: *Mean accuracy* and *SSD* for $[1, n]$ n-gram settings for the ten ICLE training sets, recurring *word-based n-grams* as features (sc = single-corpus, cc = cross-corpus evaluation)

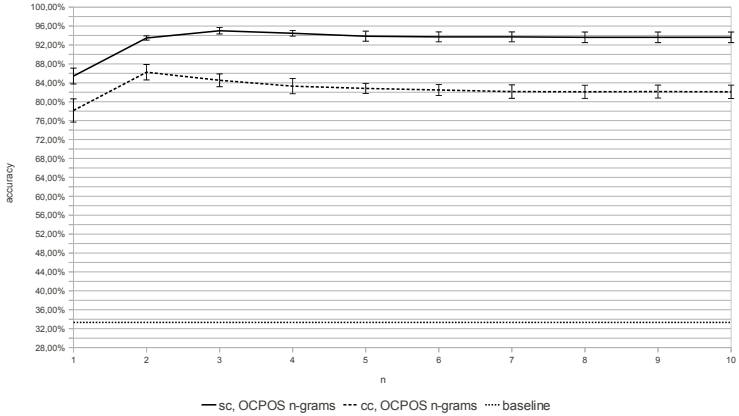


Figure 8: *Mean accuracy* and *SSD* for $[1, n]$ n-gram settings for the ten ICLE training sets, recurring *OCPOS-based n-grams* as features (sc = single-corpus, cc = cross-corpus evaluation)

Features	**Evaluation**	$[1, n]$	**Mean accuracy**	**SSD**
word n-grams	single-corpus	1	96.48%	0.64%
	cross-corpus	2	87.57%	1.32%
OCPOS n-grams	single-corpus	3	95.00%	0.68%
	cross-corpus	2	86.24%	1.63%

Table 4: Best results for ten ICLE training sets

Domain Dependence The experiments we ran with the NOCE, USE and HKUST corpora show far higher accuracies for the cross-corpus evaluation than what is reported by Brooke and Hirst (2011) for the Lang-8 corpus. In a setup with a random baseline of 14.2%, Brooke and Hirst (2011) report 70.1% – 93.8% (depending on the employed feature set) on single-corpus evaluation using ICLE, but only 15.7% – 17.0% for cross-corpus evaluation, training on ICLE and testing on Lang-8. In contrast, in a setup with a random baseline of 33.33% we obtained a best result of 95% – 96.48% (depending on the employed n-gram class) on single-corpus evaluation using ICLE, and 86.24% – 87.57% in a cross-corpus evaluation setup with training on ICLE and testing on NOCE/USE/HKUST (see Table 4 and Figure 6). Thus when using ICLE for training and another corpus instead of ICLE for testing, there is a drop of about 54% – 77% in Brooke and Hirst (2011) but only around 9% in our work. The dramatic drop Brooke and Hirst observed thus seems to be caused by some characteristic of the Lang-8 corpus and not by a general failure of the models learned on the ICLE corpus to generalize to other learner corpora.

The corpora we used for the cross-corpus evaluation were compiled by different research teams using their own essay topic lists. To investigate whether there still may be some topic overlap, we extracted the topics from our NOCE/USE/HKUST test set as well as from the ICLE training set yielding the best cross-corpus evaluation results. In both cases there were more than 100 different topics, and none of them matched between ICLE used for training and NOCE/USE/HKUST used for testing in the cross-corpus setup. Thus topic overlaps seem very unlikely to have notably skewed the results in our cross-corpus evaluation.

Conclusion

In this paper, we explored the task of *Native Language Identification (NLI)*. We derive three different classes of *recurring n-grams* as features, namely *word-*, *POS-* and *Open-Class-POS-based n-grams*. We use these features in a machine learning setup employing a Support Vector Machine (SVM) classifier on randomly selected data from the ICLE corpus incorporating seven different native languages. The best performing class are the word-based n-grams with an accuracy of 89.71%, which compares well to the 81.71% reported by Wong and Dras (2011) as the highest accuracy achieved thus far for a comparable data setup. To investigate the variance, we conducted nine further experiments based on random samples from ICLE. The mean accuracy values obtained from the overall ten experiments are very similar to those from the first experiment. The variance of the outcomes is moderate, with SSD being about 2% for the best performing settings. The bigger the step from the surface-based to more generalized features, the lower the accuracy. The recurring n-gram approach employing Open-Class-POS-based n-grams yields an accuracy of 80.57% and using POS-based n-grams we obtained 68%, which still is reasonably high considering the random baseline of 14.29% for this task.

We then investigated the claim in Brooke and Hirst (2011) that surface-based NLI classification models trained on the ICLE corpus do not generalize to other learner corpora. For this purpose we conducted a second set of experiments comparing *single-corpus* and *cross-corpus* results. In contrast to their cross-corpus findings using the Lang-8 corpus, our results show that the patterns learned on ICLE do generalize well to other learner corpora. More specifically, we showed that training on ICLE and testing on three independently collected corpora, NOCE, USE and HKUST, still yields reasonably high accuracy values of about 88% for a NLI classification task with three native languages. The low results for the Lang-8 corpus reported in Brooke and Hirst (2011) thus must have other reasons, possibly a lack of consistency in the Lang-8 pieces combined into documents, or the very different nature of the ICLE and the Lang-8 data.

References

Amaral, L. and Meurers, D. (2008). From Recording Linguistic Competence to Supporting Inferences about Language Acquisition in Context: Extending the Conceptualization of Student Models for Intelligent Computer-Assisted Language Learning. *Computer-Assisted Language Learning*, 21(4):323–338.

Axelsson, M. W. (2000). USE – The Uppsala Student English corpus: An instrument for needs analysis. *ICAME Journal*, 24:155–157.

Axelsson, M. W. (2003). *Manual: The Uppsala Student English Corpus (USE)*. Uppsala University, Department of English, Sweden. Available at `http://www.engelska.uu.se/Research/English_Language/Research_Areas/Electronic_Resource_Projects/USE-Corpus`.

Baroni, M. and Bernardini, S. (2006). A new approach to the study of translationese: Machine-learning the difference between original and translated text. *Literary and Linguistic Computing*, 21(3):259–274.

Boyd, A., Dickinson, M., and Meurers, D. (2008). On detecting errors in dependency treebanks. *Research on Language and Computation*, 6(2):113–137.

Brooke, J. and Hirst, G. (2011). Native language detection with 'cheap' learner corpora. In *Learner Corpus Research 2011 (LCR 2011)*, Louvain-la-Neuve.

Brooke, J. and Hirst, G. (2012). Measuring interlanguage: Native language identification with l1-influence metrics. In *Proceedings of the 8th ELRA Conference on Language Resources and Evaluation (LREC 2012)*, pages 779–784, Istanbul.

Chang, C.-C. and Lin, C.-J. (2011). LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2:27:1–27:27. Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.

Daelemans, W., Zavrel, J., van der Sloot, K., and van den Bosch, A. (2007). *TiMBL: Tilburg Memory-Based Learner Reference Guide, ILK Technical Report ILK 07-03*. Induction of Linguistic Knowledge Research Group Department of Communication and Information Sciences, Tilburg University, Tilburg, The Netherlands. Version 6.0.

Dickinson, M. and Meurers, W. D. (2003). Detecting errors in part-of-speech annotation. In *Proceedings of the 10th Conference of the European Chapter of the Association for Computational Linguistics (EACL-03)*, pages 107–114, Budapest, Hungary.

Dickinson, M. and Meurers, W. D. (2005). Detecting errors in discontinuous structural annotation. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL-05)*, pages 322–329.

Díaz Negrillo, A. (2007). *A Fine-Grained Error Tagger for Learner Corpora*. PhD thesis, University of Jaén, Spain.

Díaz Negrillo, A. (2009). *EARS: A User's Manual*. LINCOM Academic Reference Books, Munich, Germany.

Estival, D., Gaustad, T., Pham, S., Radford, W., and Hutchinson, B. (2007). Author profiling for English emails. In *Proceedings of the 10th Conference of the Pacific Association for Computational Linguistics*, pages 263–272.

Fan, R., Chang, K., Hsieh, C., Wang, X., and Lin, C. (2008). Liblinear: A library for large linear classification. *The Journal of Machine Learning Research*, 9:1871–1874. Software available at `http://www.csie.ntu.edu.tw/~cjlin/liblinear`.

Gass, S. and Selinker, L., editors (1983). *Language Transfer in Language Learning*. Newbury House, Rowley, MA.

Golcher, F. and Reznicek, M. (2011). Stylometry and the interplay of topic and L1 in the different annotation layers in the falko corpus. In *Proceedings of Quantitative Investigations in Theoretical Linguistics 4*, pages 29–34, Berlin.

Granger, S., Dagneaux, E., and Meunier, F. (2002). *International Corpus of Learner English*. Presses Universitaires de Louvain, Louvain-la-Neuve.

Granger, S., Dagneaux, E., Meunier, F., and Paquot, M. (2009). *International Corpus of Learner English, Version 2*. Presses Universitaires de Louvain, Louvain-la-Neuve.

Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., and Witten, I. H. (2009). The weka data mining software: An update. In *The SIGKDD Explorations*, volume 11, pages 10–18.

Koppel, M., Schler, J., and Zigdon, K. (2005). Determining an author's native language by mining a text for errors. In *Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining (KDD '05)*, pages 624–628, New York.

Milton, J. C. P. and Chowdhury, N. (1994). Tagging the interlanguage of Chinese learners of English. In *Proceedings joint seminar on corpus linguistics and lexicology, Guangzhou and Hong Kong, 19-22 June, 1993, Language Centre, HKUST*, pages 127–143, Hong Kong.

Odlin, T. (1989). *Language Transfer: Cross-linguistic influence in language learning*. Cambridge University Press, New York.

Platt, J. C. (1998). Sequential minimal optimization: A fast algorithm for training support vector machines. Technical Report MSR-TR-98-14, Microsoft Research.

Santorini, B. (1990). Part-of-speech tagging guidelines for the Penn Treebank, 3rd revision, 2nd printing. Technical report, Department of Computer Science, University of Pennsylvania.

Tsur, O. and Rappoport, A. (2007). Using classifier features for studying the effect of native language on the choice of written second language words. In *Proceedings of the Workshop on Cognitive Aspects of Computational Language Acquisition (CACLA '07)*, pages 9–16, Stroudsburg.

Wong, S.-M. J. and Dras, M. (2009). Contrastive analysis and native language identification. In *Australasian Language Technology Association Workshop 2009*, pages 53–61.

Wong, S.-M. J. and Dras, M. (2011). Exploiting parse structures for native language identification. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 1600–1610, Edinburgh, Scotland, UK.

Analysis and Enhancement of Wikification for Microblogs with Context Expansion

Taylor CASSIDY[1] *Heng JI*[1]
Lev RATINOV[2] *Arkaitz ZUBIAGA*[1] *Hongzhao HUANG*[1]
(1) DEPARTMENT OF LINGUISTICS & DEPARTMENT OF COMPUTER SCIENCE
THE GRADUATE CENTER & QUEENS COLLEGE
CITY UNIVERSITY OF NEW YORK
New York, NY, USA 10016
(2) GOOGLE INC.
New York, NY, USA 10011
taylorcassidy64@gmail.com, hengjicuny@gmail.com

ABSTRACT
Disambiguation to Wikipedia (D2W) is the task of linking mentions of concepts in text to their corresponding Wikipedia entries. Most previous work has focused on linking terms in formal texts (e.g. newswire) to Wikipedia. Linking terms in short informal texts (e.g. tweets) is difficult for systems and humans alike as they lack a rich disambiguation context. We first evaluate an existing Twitter dataset as well as the D2W task in general. We then test the effects of two tweet context expansion methods, based on tweet authorship and topic-based clustering, on a state-of-the-art D2W system and evaluate the results.

TITLE AND ABSTRACT IN BASQUE

Testuinguruaren Hedapenaren Analisia eta Hobekuntza Mikroblogak Wikifikatzeko

Esanahia Wikipediarekiko Argitzea (D2W) deritzo testuetan aurkitutako kontzeptuen aipamenak Wikipedian dagozkien sarrerei lotzeari. Aurreko lan gehienek testu formalak (newswire, esate baterako) lotu dituzte Wikipediarekin. Testu informalak (tweet-ak, esate baterako) lotzea, ordea, zaila da bai sistementzat eta baita gizakiontzat ere, argipena erraztuko luketen testuingururik ez dutelako. Lehenik eta behin, Twitter-en gainean sortutako datu-sorta bat, eta D2W ataza bera ebaluatzen ditugu. Ondoren, egungo D2W sistema baten gainean testuingurua hedatzeko bi teknika aztertu eta ebaluatzen ditugu. Bi teknika hauek tweet-aren egilean eta gaikako multzokatze metodo batean oinarritzen dira.

KEYWORDS: Disambiguation to Wikipedia (D2W), Twitter, disambiguation context.

KEYWORDS IN BASQUE: Wikipediarekiko Argitzea (D2W), Twitter, argipen testuingurua.

Proceedings of COLING 2012: Technical Papers, pages 441–456,
COLING 2012, Mumbai, December 2012.

1 Introduction

Determining the correct meaning of each word in a natural language text is a prerequisite for proper understanding. Disambiguation to Wikipedia (D2W) (Mihalcea and Csomai, 2007), the process of linking each *concept mention* in a text to a *concept* referent (i.e. a Wikipedia page), is a framework that supports the word sense disambiguation (WSD) task[1]. For example, consider the sentence, "BP said Halliburton destroyed Gulf Spill evidence". A D2W system should break the text into concept mentions and return a unique identifier (an article title, in the case of Wikipedia) for each concept. The intended meaning of each concept mention can be inferred in terms of its surface form and its context.

Mention	**Wikipedia title**
BP	*BP*
said	*Press Release*
Halliburton	*Halliburton*
destroyed	*Spoliation of Evidence*
Gulf Spill	*Deepwater Horizon oil spill*
evidence	*Evidence*

Table 1: Desired D2W output

D2W may benefit both human end-users and natural language processing (NLP) systems. When a document is *Wikified* a reader can more easily grasp its contents as information about related topics is readily accessible[2]. From a system-to-system perspective, a disambiguated corpus has the meanings of many of its terms grounded in a structurally rich ontology, and indeed there is evidence that D2W output (Ratinov and Roth, 2012; Vitale et al., 2012) can improve NLP systems. Given a concept mention in a source text, and Wikipedia, D2W operates over a representation of the following:

1. the content of the text, and how its elements are related to the concept mention.
2. the content of Wikipedia, and how its concepts are related to one another.
3. how individual elements of the text are related to elements of Wikipedia.
4. a method for generating candidate concepts for the concept mention.

Each of these items may be represented using the output of Natural Language Processing (NLP) techniques applied to the source text and Wikipedia, and/or an analysis of built-in structure (e.g. TF-IDF, Information Extraction techniques, relationships between documents, structural features of Wikipedia such as links, info boxes, and categories). Most successful D2W applications enumerate potential concept referents for a given concept mention based on the anchor text of already existing links within Wikipedia, as well as information from *redirects* and *disambiguation pages*. Context is extracted from throughout the document where a target concept mention occurs, which is then compared against Wikipedia content to narrow the hypothesis space of potential concepts. The task is therefore more challenging when concept mentions occur in short texts containing informal language.

[1]We use “concept” in both the usual sense and to refer to a Wikipedia page about a concept.
[2]http://en.wikipedia.org/wiki/Wikipedia:Glossary#Wikify.

Over 300 million Twitter users generate over 400 million tweets (posts) daily[3] [4]. The microblogging genre presents unique challenges for NLP tasks. Twitter posts (tweets) are limited to 140 characters and informal language is often used. Contextual evidence is important for accurate D2W, but for tweets it is scattered among various knowledge sources.

In this work we explore ways in which the disambiguation context of concept mentions in tweets can be enhanced. The novel contributions of the paper are as follows. First, we provide a qualitative analysis of a hand-annotated data set (Meij et al., 2012) and infer some properties of the contextual evidence most likely sought by annotators. Two sources of additional context useful for disambiguation are identified: tweets from the same author, and topically related tweets. In addition, we evaluate the contribution of these additional context types to the performance of GLOW, a state-of-the-art D2W system (Ratinov et al., 2011).

2 Related work

The task of linking expressions to Wikipedia concepts has received increased attention over the past several years, as the linking of all concept mentions in a single text (Mihalcea and Csomai, 2007; Milne and Witten, 2008a,b; Kulkarni et al., 2009; He et al., 2011; Ratinov et al., 2011), the linking of a cluster of co-referent named entity mentions spread throughout different documents (Entity Linking) (McNamee and Dang, 2009; Ji et al., 2010, 2011; Zhang et al., 2011; Han and Sun, 2011; Han et al., 2011), or the linking of a whole tweet to a single concept (Genc et al., 2011). Most D2W work has been performed on newswire collections, and most work on tweets has been limited to a particular type of concept mention. For example, the Online Reputation Management Task (Amigó et al., 2010) focused on filtering tweets containing company name to extract only those tweets that were actually related to the company.

For an n-gram deemed a concept mention, most D2W systems define candidate target concepts as a subset of those that were ever linked to using the n-gram in question as anchor text, from within Wikipedia itself (though (Zhou et al., 2010) expanded this set using search engine click results). The relative frequency with which a given n-gram links to each target concept is referred to as its *commonness* distribution[5]. Disambiguation is then couched as re-ranking, computed based on similarity between the concept mention along with its surrounding context, and a candidate concept, The systems of (Ferragina and Scaiella, 2010; Ratinov et al., 2011; Milne and Witten, 2008a; Cucerzan, 2007; Han and Zhao, 2009) take into account the coherence of all concepts linked to in a given document, based on concept similarity. (Meij et al., 2012) created the hand-labeled dataset that we use in our work. Their best performing system based on random forests outperforms commonness accord, though it does not ensure any global coherence over the concepts assigned to a given tweet.

Some TAC-KBP Entity Linking (Ji et al., 2011) systems utilized all entities in the context of a given query, disambiguating all entities simultaneously using a graph-based re-ranking algorithm (Fernandez et al., 2010; Radford et al., 2010; Cucerzan, 2011; Guo et al., 2011; Han and Sun, 2011; Han et al., 2011) or a collaborative/ensemble ranking algorithm (Pennacchiotti and Pantel, 2009; Chen and Ji, 2011; Kozareva et al., 2011) to ensure global consistency. (McNamee et al., 2011) demonstrated that co-occurring named entities are particularly helpful

[3] http://blog.twitter.com/2011/08/your-world-more-connected.html as of August 2011.

[4] http://www.mediabistro.com/alltwitter/twitter400milliontweets_b23744 as of August 2012.

[5] For an n-gram m, concept $t \in T$, $COMMONNESS(m,t) = \frac{c(m \to t)}{\sum_{t' \in T} c(m \to t')}$, where $c(m \to t)$ denotes the number of times m serves as a hyperlink to the concept t.

for Cross-lingual Entity Linking (CLEL). None of the TAC-KBP systems performed full-document D2W to include concept mentions of different types, including non-entities.

For a given concept mention, all-concept D2W work we are aware of makes use of context that is part of or derived from its containing document, whereas we explore ways to obtain supporting context in the form of additional (tweet) documents.

3 Motivation

3.1 Analysis of human annotation task

Although there is a consensus that WSD is best suited for evaluation *in vivo* (i.e. as a component of another system), a reliable gold standard data set for *in vitro* evaluation is desirable, even if the output is not intended for a human end-user (Navigli, 2009). While annotation reliability depends in part on robust guidelines designed to maximize inter-annotator agreement (IAA), IAA tends to degrade as the sense repository becomes more fine-grained (Navigli, 2009), as is the case in D2W. On one hand, if a D2W task is limited to named entities, and the set of mentions to be linked is given in advance, agreement can be rather high – e.g. 91.53%, 87.5%, and 92.98% was observed for Person, Geo-political, and Organization type entities in the TAC2010 data (Ji et al., 2010) – in spite of a sense repository which is *a priori* quite vast. In contrast, the task of linking whichever concept mentions appear important in a corpus of very small documents should prove difficult, as it is more demanding in spite of a dearth of contextual evidence. A D2W task may be characterized along two dimensions: whether concept mentions to be disambiguated are given in advance, and whether the target domain of concepts consists of all of Wikipedia or from a limited subset (e.g. only named entities). We refer to the task of linking whichever concept mentions appear important to a (largely) unrestricted domain of concepts (i.e. all Wikipedia pages) as *open-ended concept linking*.

Annotating every word without regard to its syntactic or semantic category, or its prominence in the discourse, is probably unnecessary for any application (Navigli, 2009). The criteria for determining which concept mentions to annotate must be specified in terms of (1) the properties of the target domain of concepts, (2) whether a concept exists in the target domain, and (3) the extent to which a mention is deemed ambiguous. A concept mention can be said to lack a (Wikipedia) concept referent in two distinct ways: it may be deemed *unlinkable* because the string in question, in the context in question, does not refer to a *valid concept* (i.e. one that could, in principle, appear in Wikipedia). On the other hand, the mention may refer to a valid concept, but there is not yet a corresponding Wikipedia page (see (Lin et al., 2012) for further discussion). Similarly, a concept mention can qualify as ambiguous in two ways: it may obviously refer to some valid concept, but even if each candidate has a corresponding Wikipedia page, the intended concept may be impossible to determine; on the other hand, the (Wikipedia-independent) concept being referred to may be clear, but there may be more than one (Wikipedia) concept that constitutes a correct answer in accordance with the annotation guidelines (e.g. concepts for which article mergers have been suggested might be considered equivalent, for annotation purposes; also c.f. "Gators" and "Pine nut" in section 3.2 regarding taxonomic granularity). Concept mentions that unambiguously refer to a Wikipedia concept may still present difficulties. Specification of which concepts constitute valid targets must be done in terms of the property space of all concepts, which is arguably quite complex. In the case of D2W a concept's content derives not only from explicit (e.g. infobox, category, and link structure) but implicit (article text) facts, and may be difficult to separate from personal knowledge and experience with the (Wikipedia-independent) concept in

question. Such a separation potentially limits annotation richness but may reduce inconsistency across annotators. Furthermore, determining which mentions to annotate depends not only on the properties of potential target concepts but on the prominence of the mention in question in the context in which it occurs. Perhaps a concept mentioned in passing, which does not pertain to the main point, should not be annotated. Finally, a concept might be relevant to an entire tweet though not denoted by any word or phrase therein. For example, *2011 Tohoku earthquake and tsunami* is clearly related to the tweet, "my thoughts and prayers go out to the Japanese people". We are aware of no annotation schemes that account for all of these variables, and leave a more precise formulation to future work.

3.2 Information potentially used by annotators

Annotators use information from different sources when annotating a concept mention. When short and informal texts such as tweets are analyzed in isolation, identifying the context necessary to disambiguate the concept mentions therein is non-trivial. Informative context for a given concept mention might be derived from the mention alone, within the tweet, or within the authors other tweets. Information about the author in general, his or her interests, recent events in the author's life, , and world knowledge may be informative as well. We inferred that annotators made use of several different sources of information, often simultaneously, and that world knowledge is supplemented by information acquired from Wikipedia during annotation . We aim to determine what sort of additional tweet context might have provided for an improved disambiguation context[6]. In what follows we give examples in which annotators either (1) appeared to use, or (2) failed to take advantage of, a given type of contextual support, along with analysis. Table 2 illustrates cases in which it appears that annotators *have* taken advantage of the type of supporting context in question. First, "St. Patrick's Day" is unambiguous

Type	**Tweet text**	**Mention**
Mention Alone	Are you a college kid who likes drinking, dressing up, and making irish immigrants roll in their graves? Then St. Patrick's Day is for you!	St. Patrick's Day
Within Tweet	Slump is over! Way to ball out Jeff and Damian. Much needed win. Go Hawks!!	Hawks
Within Author's Tweets	Go Gators!!! A1: Sweet 16! What a good feeling. Keep it going... Go Gators!!!' A2: What's good everyone, catching up on these Tourney games and already see some upsets... March Madness! Go Gators!	Gators

Table 2: Context type used by annotators

regardless of context. That "Hawks" refers to a sports team is implied by "Slump" and the pattern "Go ... !", but "Hawks" also may refer to the teams *Fukuoka Softbank Hawks* or *Chicago Blackhawks*, in addition to the correct referent *Atlanta Hawks*. However, only the Atlanta Hawks have players named Jeff (Teague) and Damien (Wilkins), and knowing this requires either being a member of a subculture that possesses enough knowledge to make this distinction, or having searched for this information, which can be done with a Wikipedia search and very few clicks. That "Gators" refers to a sports team is implied by "Go ... !". Whether the mention can be reliably linked to *Florida Gators men's basketball* may depend on mentions in other tweets written by the same author. In the first supporting tweet, "Sweet 16" refers to *NCAA Men's*

[6]Note that in general by *disambiguation context* we mean all information that is applicable to the disambiguation task. Later in our description of GLOW 4.1 we take a narrower definition of this term.

Division I Basketball Championship as opposed to *Sweet Sixteen (birthday)*, as evidenced by the sports context; the situation is analogous for "March Madness" in the second supporting tweet. A candidate target like *Sweet Sixteen (KHSAA State Basketball Championship)*, a less prominent basketball tournament, is ruled out by the presence of "March Madness" and "Gators" (as both are associated with only the NCAA tournament). In addition, time of publication and author attributes provide ample evidence, independent of these supporting tweets: the tweet date was March 18th, during the NCAA Division I Men's Basketball Tournament, and the author played basketball at the University of Florida. Commonness alone would not suffice as "Gators" links most commonly to *Florida Gators*, the Wikipedia page about the University of Florida's athletics in general, which is not specific enough[7]. Some additional source of information is required to link to *Florida Gators men's basketball*. Table 3 illustrates annotation errors; presumably, annotators did not take advantage of the type of context in question.

Type	**Tweet text**	**Mention**
Mention Alone	So excited to announce I'll be singing "God Bless America" during the 7th Inning Stretch at the Detroit Tigers..	Detroit Tigers
Within Tweet	Making pesto! I had to soak my nuts for 3 hours	nuts
Within Author's Tweets	It was a pool report typo. Here is exact Rhodes quote: "this is not gonna be a couple of weeks. It will be a period of days." A1: At a WH briefing here in Santiago, NSA spox Rhodes came with a litany of pushback on idea WH didn't consult with Congress. A2: Rhodes singled out a Senate resolution that passed on March 1st which denounced Khaddafy's atrocities. WH says UN rez incorporates it	Rhodes
URL Content	Awesome post from wolfblitzercnn: Behind the scenes on Clinton's Mideast trip - URL - #cnn	Clinton

Table 3: Context type *not* used by annotators

"Detroit Tigers" is unambiguously associated with *Detroit Tigers*. The given annotation for "nuts" is *Nut (fruit)*, which is reasonable, but *Pine nut* is more appropriate as it is the nut ingredient used in pesto according to Wikipedia[8]. Ben Rhodes was the deputy National Security Advisor (NSA) to Barack Obama in March of 2011. This is not clear from the tweet text, but supporting tweets each provide evidence in favor of the target *Ben Rhodes (speechwriter)*. The American Political context indicates the target concept for "Clinton" is either *Bill Clinton* or *Hillary Rodham Clinton*. To inter that Hillary Clinton went on such a trip at the time of publication requires either American political knowledge or access to the URL in the tweet.

We observe that world knowledge, including what can quickly be obtained by looking through Wikipedia, helps annotation. Many such on-the-fly inferences would be difficult to make automatically, thus additional textual context is needed in order to generate a more comprehensive disambiguation context. We consider two methods for providing such content: (1) disambiguating mentions in the context of all tweets in the dataset by the same author, and (2) disambiguating mentions in the context of all tweets in the same cluster (section 4.2.1)[9].

[7]We base this judgement on the Gricean maxim of quantity: "Be as informative as required" (c.f. http://plato.stanford.edu/entries/implicature/). We leave an analysis in this vane to future work.

[8]Pesto may be made with other nuts, but according to the article *Pesto* this does not correspond with the classic recipe. The existence of multiple correct options for candidate targets at varying taxonomic levels makes evaluation more difficult because some arbitrary choices about what constitutes "close enough" or "specific enough" must be made.

[9]Other dimensions in terms of which tweets could be clustered to filter out noise include hashtags, timestamps and the mention/retweet structure for the tweet in question. Unfortunately Twitter API restrictions render these extensions slightly less accessible for older tweets.

4 System

4.1 Global coherence

Some D2W systems aim to maximize the *global coherence* of their output, i.e., the concepts linked to in a given *source document*. Essentially, some measure of relatedness among these concepts informs the selection process for a given concept mention. A relatedness metric based on the Wikipedia link structure can leverage the co-occurrence of concept mentions in a document to the extent that the relationships expressed therein are captured in the links between their referent concepts. Concept mentions in microblog messages often lack explicit supporting context, therefore systems and annotators alike must look elsewhere for disambiguation context. We hypothesize that with the right additional context, given the resulting enriched disambiguation context, a D2W system that relies on optimizing its output for global coherence should perform better. In our experiments we do this in two ways: to a given tweet, we (1) append additional tweets by the same author, and (2) append tweets based on a clustering algorithm. We constrain the term *disambiguation context* in what follows to a set of concepts, each deemed a candidate referent of any concept mention in the source document. This definition is analogous to that used in previous sections; world knowledge, including that gained by reading tweets and examining Wikipedia, is represented approximately, via the extension of the disambiguation context that results from augmenting tweets with related tweets to create multi-tweet documents.

Enforcing constraints can be potentially harmful. The system of (Milne and Witten, 2008a) performs poorly on the tweet dataset because it relies on unambiguous concept mentions for disambiguation, the guaranteed existence of which is implausible for the microblog genre (Meij et al., 2012). TAGME (Ferragina and Scaiella, 2010) begins with commonness but enforces global coherence through a "voting" scheme in which the score associated with an n-gram m and a target concept t is derived from the vote of each other n-gram m' in the tweet. The vote of m' is the average of the relatedness scores (Milne and Witten, 2008b) between each of its candidate concepts t' with t, weighted according to $COMMONNESS(m', t')$, and though links may be pruned, this system performs poorly on the tweet dataset as well (Meij et al., 2012). GLOW (Ratinov et al., 2011), on the other hand, optimizes for global coherence using two supervised classifiers, and is conducive to a balanced disambiguation context, neither prohibitively small, nor large and noisy. Their notion of disambiguation context consists of the top candidates returned by a *local model* (described below) that for a given concept mention takes into account surrounding textual context while remaining agnostic to candidate concepts for surrounding mentions. A *global model* finalizes linking choices so as to optimize global coherence of the output. We chose to use GLOW because of its state-of-the-art performance on benchmark D2W datasets and its focus on a balanced disambiguation context.

4.2 Pipeline

The pipeline consists of three phases: first a *tweet document* is generated, then the document is fed to the D2W system, and finally results are extracted from the D2W system output.

4.2.1 Tweet document creation

The first phase consists of grouping individual tweets into documents. We create tweet documents for each experimental case, as described in Table 4.

Case	Tweet document content
By file	Each document consists of a single tweet
By author	Each document consists of all tweets by a given author
By cluster	Each document consists of all tweets in the same cluster

Table 4: Description of experimental cases

All tweets are pre-processed such that URLs are removed, and the @ and # characters are removed from user mentions and hashtags respectively. Tweets in documents are ordered chronologically by publication date, and those labeled ambiguous or non-referential are omitted.

A number of well-known probabilistic topic modeling approaches such as Probabilistic Latent Semantic Analysis (PLSA) (Hofmann, 1999) and Latent Dirichlet Allocation (LDA) (Blei et al., 2003), have been explored to discover topics from a set of documents. However, due to the shortness and lack of context, these topic modeling approaches may not work well with tweets. To overcome this difficulty, we explicitly smooth the topic distributions of tweets by building linkages between tweets, weighted by cosine similarity in terms of TF-IDF. A random walk-based approach is used to propagate the topic distribution probabilities across the linkages:

$$\begin{aligned} \hat{P}(z_k|x_i) &= \sum_{x_j \in X} w_{ji} P(z_k|x_j), \\ P(z_k|x_i) &= (1-\lambda)P(z_k|x_i) + \lambda \frac{\hat{P}(z_k|x_i)}{\sum_i \hat{P}(z_k|x_i)} \end{aligned} \quad (1)$$

where $P(z_k|x_i)$ is the probability of topic z_k for tweet x_i, w_{ij} is the similarity between x_i and x_j, and λ is a parameter that controls the balance between the previous topic distribution $P(z_k|x_i)$ and propagated topic distribution. We utilize PLSA to initialize the topic distributions. We cluster tweets using this PLSA+Random Walk-based Propagation (PRP) method by assigning a tweet x_i to the topic z_k that maximizes $P(z_k|x_i)$.

4.2.2 GLOW: a D2W system

In the second phase we use GLOW (Ratinov et al., 2011), a D2W system that disambiguates terms by attempting to optimize the global coherence of its output. Given a document d consisting of mentions $M = \{m_1, \ldots, m_N\}$, the system output consists of an N-tuple of target concepts, $\Gamma =< t_1, \ldots, t_N >$, a subset of all available concepts $T = \{t_1, \ldots, t_{|T|}\}$. Formally, one element of T is a null concept $t_\emptyset$, such that linking m to $t_\emptyset$ is akin to not linking m at all. *Local* feature functions ϕ assign $< m, t >$ pairs a high score to the extent that the context surrounding m is similar to t, and are meant to measure the likelihood that m links to t irrespective of the concepts referred to by m's surrounding mentions. *Global* feature functions ψ assign a high score to Γ to the extent that its contents are coherent. Coherence is calculated on a pairwise basis. Each global feature is either the Pointwise mutual information (PMI) or normalized Google distance (NGD) of a pair of concepts in the set, calculated in terms of the sets of concepts that either (1) link to each concept in the pair, (2) are linked to from each concept in the pair, or (3) are in the intersection of the sets defined in (1) and (2), for each concept in the pair[10]. Thus, GLOW attempts to solve the following optimization problem for a given document d:

$$\Gamma^* = \arg\max_{\Gamma} [\sum_{i=1}^{N} \phi(m_i, t_i) + \psi(\Gamma)] \quad (2)$$

[10]See (Ratinov et al., 2011) for a detailed explanation including the adaptations of PMI and NGD used.

Where Γ^* is the optimal output. This problem is NP hard, so inter-concept relatedness is calculated pairwise to reduce complexity, reformulating the problem as:

$$\Gamma^* \approx \arg\max_{\Gamma} \sum_{i-1}^{N} [\phi(m_i, t_i)] + \sum_{t_j \in \Gamma'} [\psi(t_i, t_j)] \quad (3)$$

The optimization is performed in two stages. First, in the *ranker* stage, Γ^* is found but without allowing any mention to be linked to $t_\emptyset$. Next, in the *linker* stage, whether each mention's top candidate should be replaced by $t_\emptyset$ is determined. In the system output, mentions linked to $t_\emptyset$ have a negative linker score while others have a positive linker score.

4.2.3 Extracting output

For a given case, each tweet document d is fed to the D2W system separately, the output of which consists of mentions that were linked (including those ultimately linked to $t_\emptyset$ and their associated target concepts). Each mention is associated with a *linker score* - the confidence associated with the choice to link that term - while each of its candidate target concepts is associated with a *ranker score* - the confidence associated with that particular concept. Thus for each linked mention m_{di} we have its *result tuple*, $R(m_{di})$ which consists of a linker score and a list of k targets, ordered according to their ranker score.

$$R(m_{di}) = < ls(m_{di}), (< t^1_{m_{di}}, rs(t^1_{m_{di}}) >, \ldots, < t^k_{m_{di}}, rs(t^k_{m_{di}}) >) > \quad (4)$$

We abbreviate the first and second elements of $R(m_{di})$ as $R(m_{di})_{ls}$ and $R(m_{di})_{rs}$. The output for each set of surface-identical mentions in d is then aggregated into one result tuple as follows. For a surface string s_d associated with one or more mentions in d, the set of associated result tuples is denoted R_{s_d}. Then $R(s_d)$, the result tuple for s_d, is defined by:

$$R(s_d) = < max_{R(m_{di}) \in R_{s_d}} R(m_{di})_{ls}, \bigcup_{R(m_{di}) \in R_{s_d}} R(m_{di})_{rs} > \quad (5)$$

In other words for any surface string, we consider all target concepts and associated ranker scores, and associate the string with the highest linker score of any matching mention.

Output aggregation is informed by two parameters: *longest-n-gram*, a binary parameter indicating whether or not the "longest n-gram heuristic" is used (as opposed to "all terms"), and a *linker score threshold* λ. If the longest n-gram heuristic is used, then if both "Houston Rockets" and "Rockets" are disambiguated, for example, "Rockets" will be ignored. Finally, $R(s_d)$ will only be included in the final output if $R(s_d)_{ls} > \lambda$.

5 Data and scoring metric

In this section we describe the dataset, provide a critical evaluation, and explain how system output is evaluated.

5.1 Construction, content, and annotation

We use the dataset described in (Meij et al., 2012), which we refer to as *gold1*. A random sample of verified twitter accounts were selected, and up to their 20 most recent tweets were extracted. The original dataset had 562 tweets, but due to tweets having been deleted, the dataset consists of 502 tweets from 28 authors. Annotators used an interface enabling them to read and annotate tweets, searching Wikipedia as needed, and were instructed to, where

possible, indicate which concepts were "contained in, meant by, or relevant" to a particular tweet. Alternatively they were permitted to label tweets as ambiguous or as having referents outside of Wikipedia; 127 tweets were labeled as such and discarded[11]. The gold standard consists of the union of annotations from two annotators which amounts to 812 annotations (not including discarded tweets). URLs were removed entirely while mentions and hashtags were edited to remove leading @ and # characters respectively[12].

5.2 System false positives

Some system errors are the result of human annotation omissions (Meij et al., 2012) . There were 229 false positives when applying the GLOW to single tweets, using the longest n-gram heuristic, with the linker score threshold at -0.04. We looked at each one and rated it incorrect (110), partially correct (49), or correct (70). False positives deemed correct (FPDC) were labeled as follows: "@" (2), "#" (13), "lol" (5), "replace" (6), "new" (35), "equivalent" (9).

The *gold2* dataset is the result of adding all FPDC to *gold1*. For each FPDC type we provide representative system results followed by analysis. Table 5 gives some examples of each FPDC type, along with from the system output or *gold1*.

FPDC labeled *new* consist of a mention that annotators previously did not link and a target concept deemed correct. Table 5 gives three examples; "support", in this case, is an example of an analogous annotation in *gold1*. In the first a song was omitted in one tweet whereas in another a song was linked, and similarly so for the dates in the second example and its counterpart. In the third, a governmental acronym and an associated term are omitted, whereas in its counterpart they are annotated.

FPDC labeled *replace* consist of mentions that were originally annotated, but we believe the annotation provided by GLOW was significantly better. Table 5 contains three examples; "support", in this case, illustrates the change made by the system. In the first example some evidence was available in the tweet itself (though more conclusive evidence is available in the author's other tweets, as alluded to in section 3.2). In the second example note that *Grammy Nominees* is an album containing Grammy-nominated songs for a given year, but the URL in the tweet links to a page where only the album "Infinite Arms" can be purchased, revealing that the original annotation is incorrect (note that annotators did not have access to URLs in tweets). In the third the original annotation is too general. Note that the vast majority of false positives deemed *partially correct* are of this type.

FPDC labeled *eq* are instances where GLOW's target was deemed equivalent to the target in the original annotation. Table 5 lists three such examples followed by justification. FPDC labeled @ were user mentions that were not annotated, even though the user is identifiable and is prominent enough to have a Wikipedia page. FPDC labeled # were hash marked mentions that were not annotated. FPDC labeled *lol* were mentions expressing that the user laughed, e.g. "lol", "ROFL", "LMAO", etc. Annotating such mentions depends on whether we want to annotate actions the user indicates he or she performs in conjunction with the tweet.

[11] We acknowledge that ignoring non-referential tweets makes the task easier. Work that focuses on a system's ability to ignore irrelevant content is needed. Tweets were deemed ambiguous if annotators identified more than one correct answer, a case our system did not accommodate.

[12] @ and # characters were visible to human annotators, who were asked to ignore hash tagged terms unless their meaning is obvious; they were stripped during pre-processing. For further details and access to the dataset: http://ilps.science.uva.nl/resources/wsdm2012-adding-semantics-to-microblog-posts/ (Meij et al., 2012).

Note that these omissions and errors drawn from a subset of those mentions whose annotation was corrected by GLOW; however, other errors and omissions exist (e.g. when both humans and GLOW made mistakes). The purpose of this analysis is not to discredit the dataset. Classification of annotations or omissions as erroneous is highly subjective in that it depends on both the user's interpretation of the annotation guidelines, which in this case were rather open-ended, along with their own world knowledge. We believe the formation of guidelines and annotation methods that are more robust to such discrepancies is an important avenue of research.

Type	**False Positives Deemed Correct (FPDC)**	**Support from system output or *gold1***
New	So excited to announce I'll be singing "God Bless America" during the 7th Inning Stretch at the Detroit Tigers... URL	#NP "Crazy"- The Boys - *heeeeey*
New	Enter to win FREE tickets to my Houston show March 29th! URL	Ben has announced a benefit show in Charleston on December 10th for the family of Andy Kotowitcz. Details here: URL
New	DOE approves $102 million loan aid for Maine wind farm - URL	So nothing has changed from last night. Timetable for handover of no-fly zone enforcement is still "days" according to WH
Replace	Sweet 16! What a good feeling. Keep it going... Go Gators!!!	Sweet sixteen (birthday) → NCAA Men's Division I Basketball Championship
Replace	The deluxe version of the Grammy Nominated, Infinite Arms, is available for a special holiday price. Get it here URL	Grammy Nominees → Grammy Award
Replace	RT @user: @user I always spend my summer here! An old-growth forest within the Sipalay island in the Philippines!	Forest → Old-growth forest
Eq	Photos are great for engaging with your audiences. Upload images to Flickr.com and create slideshows with URL	Slideshow redirects to Slide show
Eq	Jalen said "How did Santa make my presents and it says Made in China?! Santa ain't Chinese!" lmao	People's Republic of China was merged with China
Eq	The Devil is a liar! Thank God for giving you to chance to see this beautiful morning. I'm thankful and very blessed.	Satan was deemed conceptually equivalent to Devil
Eq	Which childhood story would you miss the most? Peter Pan and Mary Plain for me. URL #IdMiss	Peter and Mary is the fairy tale whose main character is Peter Pan
#	#NATO to enforce arms embargo against #Libya - URL #Gaddafi	The situation in Libya is of great concern. NATO can act as an enabler and coordinator if and when member states will take action
@	RT @user: Tweets to 6.5 million followers in the name of #girlseducation: Thanks @Shakira, @user and @user! URL	Obama set to deliver a response on #Libya soon

Table 5: A mention is underlined to indicate it was annotated.

6 Experiments

In this section we present and discuss experimental results. For each case we generate tweet documents (see section 4.2.1), each of which is fed to the D2W system, and final output is extracted from system output (see section 4.2.3). We calculate precision, recall, and MRR.

6.1 Evaluation metric

Output is evaluated against *gold1* and *gold2* (see section 5). Final output for a tweet document distinguishes identical mentions allowing each tweet to be associated with a list of targets.

Table of f-measures for different experimental parameters						
	By file		**By cluster**		**By author**	
	gold1	***gold2***	***gold1***	***gold2***	***gold1***	***gold2***
all terms	44.19%	51.00%	46.35%	51.82%	47.00%	52.56%
longest ngram	45.58%	52.79%	47.50%	53.13%	48.07%	53.92%
Link Threshold for F-Measures Shown Above						
	gold1	***gold2***	***gold1***	***gold2***	***gold1***	***gold2***
all terms	-0.2	-0.2	0	0.1	0.1	0
longest ngram	-0.4	-0.4	0	-0.2	0	-0.2
Cluster Size for F-Measures Shown Above						
	gold1	***gold2***	***gold1***	***gold2***	***gold1***	***gold2***
all terms	n/a	n/a	28	28	n/a	n/a
longest ngram	n/a	n/a	28	50	n/a	n/a

Table 6: Overview of different methods

Precision (P), recall (R), and F-measure (F1) are calculated on a by-tweet basis as follows:

$$P = \frac{\sum_i^{N_S} |T(x_i) \cap G(x_i)|}{N_S} \tag{6}$$

$$R = \frac{\sum_i^{N_G} |T(x_i) \cap G(x_i)|}{N_G} \tag{7}$$

$$F_1 = \frac{2PR}{P+R} \tag{8}$$

Where N_S is the number of $\langle m, t \rangle$ pairs in the system output, each x_i is a tweet, $T(x)$ contains the top target concept from each mention in tweet x, $G(x)$ contains each concept associated with x by an annotator, and N_G is the total number of gold standard annotations. Mean Reciprocal Rank (MRR) is calculated over all gold annotation tuples $\langle x, t \rangle \in G$ as follows:

$$MRR = \frac{1}{|G|} \sum_{i=1}^{|G|} \frac{1}{rank_i} \tag{9}$$

Where $rank_i$ is r if $< t_i^r, rs(t_i^r) >$ is in $R(s_d)_{rs}$, where t_i is the target of the ith gold annotation $< x_i, t_i >$, and d is the document that contains x_i. Otherwise, $1/rank_i = 0$.

6.2 Results

In order to investigate the most effective way to extend tweet context to improve D2W, we augmented single tweets using either the *by author* or *by cluster* methods (see Table 4)

For the case of single tweets, each tweet was input one at a time into GLOW. For cases where tweets were aggregated, a document containing the tweets, delimited by a line break and in chronological order by publication date, was input into GLOW.

Table 6 presents the results of applying these different methods to augment tweets. *By author* outperforms *by cluster*. Table 7 shows details for the top performing systems of each type. The systems that achieve the top Mean Reciprocal Rank (MRR), as well as MRR for the systems with the top F measure, are shown in Table 8.

The by file system performs the worst in each category. By author improves recall while by cluster improves precision. The Wilcoxon matched pairs signed rank text shows that improvement in f-measure from by file to by author method was significant ($p < .01$); improvement from by file

Statistics for top performing systems of each type							
System	**Correct**	**Missed**	**False Positives**	**Total Output**	**Precision**	**Recall**	**F1**
by file	307	505	228	535	0.5738	0.3781	0.4558
by author	318	494	193	511	0.6223	0.3916	0.4807
by cluster	309	503	180	489	0.6319	0.3805	0.4750

Table 7: Detailed results by system type using the optimal parameters for each

		MRR1		**MRR2**	
		Best Params	**Best F Params**	**Best Params**	**Best F Params**
by File	All terms	44.20%	41.62%	43.77%	41.29%
	Longest ngram	40.75%	39.70%	40.50%	39.53%
by Author	All terms	45.82%	42.27%	45.44%	42.03%
	Longest ngram	42.23%	40.21%	42.06%	40.05%
by Cluster	All terms	44.89%	41.86%	44.42%	41.56%
	Longest ngram	41.52%	39.35%	41.32%	39.25%

Table 8: Best MRR & MRR for parameters yielding best F1

to by cluster was significant as well ($p < .013$)[13]. The Adjusted Rand Index (ARI) is a measure of cluster similarity, corrected for chance. The ARI between the top author based and cluster based methods is low (.0128), indicating that there is very little overlap.

Detailed results for the highest performing systems are shown in Table 7. The differences in output moving from by file to by author systems consisted of 23 gains and 12 losses. Gains resulted for the following reasons: (i) because the top candidate was correct in both cases but in the by author case the linker score exceeded 0.0, but in the by file case it did not exceed -0.4; (ii) the top candidate was incorrect in the by file case but correct in the by author case; (iii) a surface-identical mention in another tweet either had a better linker score and/or it was linked to the correct target[14]. Some gains were deemed neutral (4) or bad (1), meaning that we deemed the change made incorrect, contrary to *gold1*. Examples of changes are illustrated in Table 9 and explained below. Losses were categorized in an analogous way.

Tweet	**By file**	**By author**	**Type**
Japan is one of NATOs global partners. On behalf of our Allies I want to extend our heartfelt condolences to those who have lost loved ones	Empire of Japan	Japan	Good change
Ejoying myself in Whistler!	Whistler, British Columbia	Whistler, British Columbia	Greater LS for identical mention
RT @kmoxnews: Section of I-55 Closed Until Monday: I-55 will be closed in both directions between Carondelet and the 4500 block of...	Carondelet, St. Louis	Carondelet, St. Louis	Context
Obama says he doesn't expect harmful levels of radiation to hit the U.S. ... public health experts say no precautionary measures needed	Ionizing radiation	Radiation	Neutral Change
Making pesto! I had to soak my nuts for 3 hours!	Pine nut	Nut (fruit)	Bad change

Table 9: Gains from by file to by author system

[13]We randomly split tweets into 17 groups, yielding 17 lists of annotations. We calculated F-measure for each group using both methods and the resulting F-measure pairs served as input to the test.

[14]Gains are determined with respect to the gold standard. The best by file and by author systems had linker score thresholds of -0.4 and 0.0, respectively.

The first change is due to additional supporting context in the author's other tweets, which include entities from modern politics (e.g. politician names and organizations). This additional context alleviates the noisy mention "Allies" which is strongly associated with *World War II* and hence *Empire of Japan*. In the second case the author had later mentioned "Whistler", a popular winter sports destination, near mentions of "slopes", "snowboarding", and "jet lag". In the third case, the author frequently mentions "St. Louis" in other tweets.

7 Conclusions and future work

D2W systems that attempt to maximize the global coherence of output have been successful in formal genres, but the required supporting concept mentions are hidden in the Twitter domain. Our approach to this apparent data sparsity is orthogonal to that taken by (Meij et al., 2012), who designed features in terms of individual n-grams and candidate concepts, rarely dependent on the entire tweet (5 out of 33), never attempting to achieve global coherence. We showed that for a given tweet, adding tweets based on both authorship and topical similarity provided GLOW sufficient information to enhance the disambiguation context for concept mentions therein, yielding statistically significant gains over the *by file* base.

We have provided a qualitative analysis of an existing hand-labeled dataset, which raised questions about both definition and evaluation of the D2W task, elucidating various sources of difficulty. In future work we plan to generate comprehensive annotation and evaluation guidelines for D2W. Second, it is clear that sometimes there is more than one appropriate target concept for a given concept mention. In some cases two concepts are equally plausible targets (*Devil* vs. *Satan* for the n-gram "the devil"), while in other cases returning a concept slightly higher up in the *is-a* taxonomic structure would plausibly still be useful for downstream applications (e.g. returning *Florida Gators* instead of the more accurate *Florida Gators men's basketball*, given only "go Gators!!"). We plan to explore principled criteria for Wikipedia concept equivalence that go beyond the provided redirects, as well as evaluation methods that do not penalize such "not so bad" deviation from human annotation. Finally, we plan to evaluate the effects of expanding tweet context based on Twitter-centric features such as the mention/retweet structure and hashtags, as well as websites linked to from within tweets.

Acknowledgments

This work was supported by the U.S. Army Research Laboratory under Cooperative Agreement No. W911NF-09-2-0053, the U.S. NSF Grants IIS-0953149 and IIS-1144111 and the U.S. DARPA BOLT program. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation here on.

References

Amigó, E., Artiles, J., Gonzalo, J., Spina, D., Liu, B., and Corujo, A. (2010). Weps-3 evaluation campaign: Overview of the online reputation management task. In *CLEF 2010 (Notebook Papers/LABs/Workshops)*.

Blei, D. M., Ng, A. Y., and Jordan, M. I. (2003). Latent dirichlet allocation. *J. Mach. Learn. Res.*, 3.

Chen, Z. and Ji, H. (2011). Collaborative ranking: A case study on entity linking. In *Proc. EMNLP2011*.

Cucerzan, S. (2007). Large-scale named entity disambiguation based on wikipedia data. In *EMNLP-CoNLL 2007*.

Cucerzan, S. (2011). Tac entity linking by performing full-document entity extraction and disambiguation. In *Proc. TAC 2011 Workshop*.

Fernandez, N., Fisteus, J. A., Sanchez, L., and Martin, E. (2010). Webtlab: A cooccurence-based approach to kbp 2010 entity-linking task. In *Proc. TAC 2010 Workshop*.

Ferragina, P. and Scaiella, U. (2010). Tagme: on-the-fly annotation of short text fragments (by wikipedia entities). In *Proceedings of the 19th ACM international conference on Information and knowledge management*, pages 1625–1628. ACM.

Genc, Y., Sakamoto, Y., and Nickerson, J. V. (2011). Discovering context: classifying tweets through a semantic transform based on wikipedia. In *Proceedings of the 6th international conference on Foundations of augmented cognition: directing the future of adaptive systems*, FAC'11, pages 484–492.

Guo, Y., Che, W., Liu, T., and Li, S. (2011). A graph-based method for entity linking. In *Proc. IJCNLP2011*.

Han, X. and Sun, L. (2011). A generative entity-mention model for linking entities with knowledge base. In *Proc. ACL2011*.

Han, X., Sun, L., and Zhao, J. (2011). Collective entity linking in web text: A graph-based method. In *Proc. SIGIR2011*.

Han, X. and Zhao, J. (2009). Named entity disambiguation by leveraging wikipedia semantic knowledge. In *Proceedings of the 18th ACM conference on Information and knowledge management*, CIKM 2009.

He, J., de Rijke, M., Sevenster, M., van Ommering, R., and Qian, Y. (2011). Generating links to background knowledge: A case study using narrative radiology reports. In *Proceedings of the 20th ACM international conference on Information and knowledge management*, pages 1867–1876. ACM.

Hofmann, T. (1999). Probabilistic latent semantic indexing. In *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '99.

Ji, H., Grishman, R., and Dang, H. (2011). Overview of the tac 2011 knowledge base population track. In *Text Analysis Conference (TAC) 2011*.

Ji, H., Grishman, R., Dang, H., Griffitt, K., and Ellis, J. (2010). Overview of the tac 2010 knowledge base population track. In *Text Analysis Conference (TAC) 2010*.

Kozareva, Z., Voevodski, K., and Teng, S. (2011). Class label enhancement via related instances. In *Proc. EMNLP2011*.

Kulkarni, S., Singh, A., Ramakrishnan, G., and Chakrabarti, S. (2009). Collective annotation of wikipedia entities in web text. In *KDD*, pages 457–466.

Lin, T., Mausam, and Etzioni, O. (2012). No noun phrase left behind: Detecting and typing unlinkable entities. In *EMNLP-CoNLL*, pages 893–903.

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

McNamee, P., Mayfield, J., Lawrie, D., Oard, D. W., and Doermann, D. (2011). Cross-language entity linking. In *Proc. IJCNLP2011*.

Meij, E., Weerkamp, W., and de Rijke, M. (2012). Adding semantics to microblog posts. In *Proceedings of the fifth ACM international conference on Web search and data mining*, WSDM '12, New York, NY, USA. ACM.

Mihalcea, R. and Csomai, A. (2007). Wikify!: linking documents to encyclopedic knowledge. In *CIKM*, volume 7, pages 233–242.

Milne, D. and Witten, I. (2008a). Learning to link with wikipedia. In *Proceeding of the 17th ACM conference on Information and knowledge management*, pages 509–518. ACM.

Milne, D. and Witten, I. (2008b). Learning to link with wikipedia. In *An effective, low-cost measure of semantic relatedness obtained from wikipedia links*. the Wikipedia and AI Workshop of AAAI.

Navigli, R. (2009). Word Sense Disambiguation: a survey. *ACM Computing Surveys*, 41(2):1–69.

Pennacchiotti, M. and Pantel, P. (2009). Entity extraction via ensemble semantics. In *Proc. EMNLP2009*.

Radford, W., Hachey, B., Nothman, J., Honnibal, M., and Curran, J. R. (2010). Cmcrc at tac10: Document-level entity linking with graph-based re-ranking. In *Proc. TAC 2010 Workshop*.

Ratinov, L. and Roth, D. (2012). Learning-based multi-sieve co-reference resolution with knowledge. In *Proc. EMNLP*.

Ratinov, L., Roth, D., Downey, D., and Anderson, M. (2011). Local and global algorithms for disambiguation to wikipedia. In *Proc. of the Annual Meeting of the Association of Computational Linguistics (ACL)*.

Vitale, D., Ferragina, P., and Scaiella, U. (2012). Classification of short texts by deploying topical annotations. In *ECIR*, pages 376–387.

Zhang, W., Su, J., and Tan, C. L. (2011). A wikipedia-lda model for entity linking with batch size changing. In *Proc. IJCNLP2011*.

Zhou, Y., Nie, L., Rouhani-Kalleh, O., Vasile, F., and Gaffney, S. (2010). Resolving surface forms to wikipedia topics. In *Proceedings of the 23rd International Conference on Computational Linguistics*, COLING '10, pages 1335–1343.

On the Effectiveness of Using Sentence Compression Models for Query-Focused Multi-Document Summarization

Yllias Chali *Sadid A. Hasan*
University of Lethbridge, Lethbridge, AB, Canada
chali@cs.uleth.ca, hasan@cs.uleth.ca

ABSTRACT
This paper applies sentence compression models for the task of query-focused multi-document summarization in order to investigate if sentence compression improves the overall summarization performance. Both compression and summarization are considered as global optimization problems and solved using integer linear programming (ILP). Three different models are built depending on the order in which compression and summarization are performed: 1) *ComFirst* (where compression is performed first), 2) *SumFirst* (where important sentence extraction is performed first), and 3) *Combined* (where compression and extraction are performed jointly via optimizing a combined objective function). Sentence compression models include lexical, syntactic and semantic constraints while summarization models include relevance, redundancy and length constraints. A comprehensive set of query-related and importance-oriented measures are used to define the relevance constraint whereas four alternative redundancy constraints are employed based on different sentence similarity measures using a) cosine similarity, b) syntactic similarity, c) semantic similarity, and d) extended string subsequence kernel (ESSK). Empirical evaluation on the DUC benchmark datasets demonstrates that the overall summary quality can be improved significantly using global optimization with semantically motivated models.

KEYWORDS: Sentence compression, query-focused multi-document summarization, integer linear programming (ILP).

Proceedings of COLING 2012: Technical Papers, pages 457–474,
COLING 2012, Mumbai, December 2012.

1 Introduction and Related Work

Text summarization is a good way to compress large amount of information into a concise form by selecting the most important information and discarding redundant information (Mani and Maybury, 1999). Query-focused multi-document summarization aims to create a summary from the available source documents that can answer the requested information need (Chali and Hasan, 2012). Extraction-based automatic summarization has been a common practice over the years for its simplicity (Edmundson, 1969; Kupiec et al., 1995; Carbonell and Goldstein, 1998; Lin, 2003; Martins and Smith, 2009; Berg-Kirkpatrick et al., 2011). Extraction of the most important sentences to form a summary can degrade the summary quality if there exists a longer sentence with partly relevant information to prevent inclusion of other important sentences (due to summary length constraint) (Martins and Smith, 2009). Sentence compression can be a good remedy for this problem where the task can be viewed as a single-sentence summarization (Jing, 2000; Clarke and Lapata, 2008). Sentence compression[1] aims to retain the most important information of a sentence in the shortest form whilst being grammatical at the same time (Knight and Marcu, 2000, 2002; Lin, 2003). Previous researches have shown that sentence compression can be used effectively in automatic summarization systems to produce more informative summaries by reducing the redundancy in the summary sentences (Jing, 2000; Knight and Marcu, 2002; Lin, 2003; Daumé III and Marcu, 2005; Zajic et al., 2007; Madnani et al., 2007; Martins and Smith, 2009; Berg-Kirkpatrick et al., 2011). However, most of these researches either focused on the task of single document summarization and generic summarization or did not consider global properties of the sentence compression problem (Clarke and Lapata, 2008). Due to the vast increase in both the amount of online data and the demand for access to different types of information in recent years, attention has shifted from single document and generic summarization[2] toward query-based multi-document summarization. On the other hand, sentence compression can achieve superior performance if it can be treated as an optimization problem and solved using integer linear programming (ILP) to infer globally optimal compressions (Gillick and Favre, 2009; Clarke and Lapata, 2008). ILP has recently attracted much attention in the natural language processing (NLP) community (Roth and Yih, 2004; Clarke and Lapata, 2008; Punyakanok et al., 2004; Riedel and Clarke, 2006; Denis and Baldridge, 2007). Gillick and Favre (2009) proposed to extend their ILP formulation for a concept-based model of summarization by incorporating additional constraints for sentence compression. However, to the best of our knowledge, there has not been a single research that deeply investigates the potential of using ILP-based sentence compression models for the task of query-focused multi-document summarization. In this paper, we accomplish this task by considering both compression and summarization as global optimization problems.

The sentence compression models used in the existing automatic summarization systems mostly exploit various lexical and syntactic properties of the sentences (Knight and Marcu, 2002; Mcdonald, 2006; Clarke and Lapata, 2008; Cohn and Lapata, 2008; Galanis and Androutsopoulos, 2010). A recent work has shown that discourse segmentation could be incorporated in a sentence compression system which can aid automatic summarization (Molina et al., 2011).

[1] Although most of the works on sentence compression are mainly related to the English language, researchers have also worked on sentence compression related to languages other than English (Molina et al., 2011; Filippova, 2010; Bouayad-Agha et al., 2006). Our work is applied to the English language. However, we believe that the proposed techniques can be applicable to other languages provided that the lexical, syntactical and semantic properties of the corresponding language are considered.

[2] A generic summary includes information which is central to the source documents whereas a query-oriented summary should formulate an answer to the user query (Goldstein et al., 1999).

Lin (2003) showed that pure syntactic-based compression does not improve a generic summarization system. A most recent work has shown that sentence compression can achieve better performance if semantic role information can be incorporated into the model (Yoshikawa et al., 2012). Inspired by their work, we recast their formulation as an ILP for sentence compression with semantic role constraints. We build three different ILP-based sentence compression models: 1) a bigram language model with lexical and syntactic constraints (derived from Clarke and Lapata (2008)), 2) the bigram language model with a topic signature modeling function (Lin and Hovy, 2000), and 3) the bigram language model with semantic role constraints (Yoshikawa et al., 2012). We choose to build them since the variation of these models were shown to achieve better results comparable to the state-of-the-art techniques (Clarke and Lapata, 2008; Yoshikawa et al., 2012). We perform a rigorous study to analyze the effectiveness of using these sentence compression models to generate query-focused summaries. For this study, we compose three different models depending on the order to perform sentence compression and extraction: 1) *ComFirst*, 2) *SumFirst*, and 3) *Combined*. The main motivation behind building these models is that we intend to study if the order of performing compression and extraction can affect the overall performance of the query-focused multi-document summarization. Martins and Smith (2009) argued that the two-step "pipeline" approaches such as *ComFirst* and *SumFirst* might often fail to select global optimal summaries.

Query-focused extractive multi-document summarization generally needs three essential criteria to be satisfied (McDonald, 2007): 1) Relevance: to contain informative sentences relevant to the given query, 2) Redundancy: to not contain multiple similar sentences, and 3) Length: should follow a fixed length constraint. We define a global optimization model that uses ILP to infer optimal summaries. The existing ILP formulations to the summarization task mostly rely on relevance and redundancy functions (such as word-level cosine similarity measure, word bigrams) that are primitive in nature (McDonald, 2007; Gillick and Favre, 2009; Martins and Smith, 2009). The major limitation of these approaches is that they do not consider the sequence of words (i.e. word ordering). They ignore the syntactic and semantic structure of the sentences and thus, cannot distinguish between "The police shot the gunman" and "The gunman shot the police". The researchers speculate that the better the relevance and redundancy functions could be, the more the solutions would be efficient (Gillick and Favre, 2009). In the proposed optimization framework, we incorporate a comprehensive set of query-related and importance-oriented measures to define the relevance function. We employ four alternative redundancy constraints based on different sentence similarity measures using a) cosine similarity, b) syntactic similarity, c) semantic similarity, and d) extended string subsequence kernel (ESSK). We propose the use of syntactic tree kernel (Moschitti and Basili, 2006), shallow semantic tree kernel (Moschitti et al., 2007), and a variation of the extended string subsequence kernel (ESSK) (Hirao et al., 2003) to accomplish the task. Our empirical evaluation on the DUC benchmark datasets demonstrate the effectiveness of applying sentence compression for the task of query-focused multi-document summarization. The results also show that the quality of the generated summaries vary based on the use of alternative redundancy constraints in the optimization framework.

2 ILP-based Sentence Compression Models

An ILP is a constrained optimization problem, where both the cost function and constraints are linear in a set of integer variables (McDonald, 2007; Clarke and Lapata, 2008). In this section we describe three ILP-based sentence compression models which we apply for the task of query-focused multi-document summarization. Our first model is a bigram language model

derived from the work of Knight and Marcu (2002); Clarke and Lapata (2008). Our second model is close in spirit rather different in content to Clarke and Lapata (2008). In this model, we combine the bigram language model with a corpus-based topic signature modeling approach of Lin and Hovy (2000). Our first two models include various lexical and syntactical constraints based on the work of Clarke and Lapata (2008). In the third model, we add a set of semantically motivated constraints into the bigram language model based on the work of Yoshikawa et al. (2012).

2.1 Bigram Language Model

According to Clarke and Lapata (2008), the sentence compression problem can be formally defined as follows. Let $S = w_1, w_2, \cdots, w_n$ is an original sentence in a document. To represent the words to be included in the compressed version of this sentence, we define a set of indicator variables δ_i that are set to 1 if i-th word is selected into the compression, and 0 otherwise. To make decisions based on word sequences (rather than individual words), we define additional indicator variables a_i (that are set to 1 if i-th word starts the compression, and 0 otherwise), b_i (that are set to 1 if i-th word ends the compression, and 0 otherwise), and c_{ij} (that are set to 1 if sequence w_i, w_j is present in the compression, and 0 otherwise). Now the inference task is solved by maximizing the following objective function (that includes the overall sum of the decision variables multiplied by their log-transformed corpus bigram probabilities) (Clarke and Lapata, 2008):

$$Maximize \sum_i a_i \cdot P(w_i|start) + \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} c_{ij} \cdot P(w_j|w_i) + \sum_i b_i \cdot P(end|w_i) \quad (1)$$

such that $\forall i, j \in \{1 \cdots n\}$:

$$\delta_i, a_i, b_i, c_{ij} \in \{0, 1\} \quad (2)$$

$$\sum_i a_i = 1 \quad (3)$$

$$\delta_j - a_j - \sum_{i=1}^{j} c_{ij} = 0 \quad (4)$$

$$\delta_i - \sum_{j=i+1}^{n} c_{ij} - b_i = 0 \quad (5)$$

$$\sum_i b_i = 1 \quad (6)$$

$$\sum_i \delta_i \geq l \quad (7)$$

$$\sum_{i:w_i \in verbs} \delta_i \geq 1 \quad (8)$$

$$\delta_i = 1 \quad (9)$$
$$\forall i : w_i \in personal\ pronouns$$

$$\delta_i = 0 \quad (10)$$
$$\forall i : w_i \in words\ in\ parentheses$$

$$\delta_i - \delta_j = 0 \quad (11)$$
$$\forall i, j : w_j \in possessive\ mods\ of\ w_i$$

The objective function in Equation 1 is maximized to find the optimal target compression where "start" and "end" denote w_0 and w_n, respectively. The above ILP formulation incorporates

various constraints. The first constraint states that the variables are binary. The later constraints are defined to disallow invalid bigram sequences in the compression. Constraint 3 states that exactly one word can start a compression. Constraint 4 and Constraint 5 are responsible to ensure correct bigram sequences, whereas Constraint 6 denotes that exactly one word can end the compression. On the other hand, Constraint 7 forces the compression to have at least l words. We add some additional constraints (Constraint 8 to Constraint 11) from Clarke and Lapata (2008) to ensure that the target compressions are lexically and syntactically acceptable. To accomplish this purpose, we use the Oak system[3] (Sekine, 2002) and the Charniak parser[4] (Charniak, 1999) to obtain information regarding parts-of-speech and grammatical relations in a sentence.

2.2 Topic Signature Model

We use a topic signature modeling approach (Lin and Hovy, 2000) to identify the important content words from the original source sentence. The important words are considered to have significantly greater probability of occurring in a given text compared to that in a large background corpus. We incorporate this importance score into the objective function of the bigram language model (Section 2.1) to ensure that the target compression prefers to keep important content words. We use a topic signature computation tool[5] for this purpose. The background corpus that is used in this tool contains 5000 documents from the English GigaWord Corpus. Our modified objective function becomes:

$$Maximize \sum_i \delta_i \cdot I(w_i) + \sum_i a_i \cdot P(w_i|start) + \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} c_{ij} \cdot P(w_j|w_i) + \sum_i b_i \cdot P(end|w_i) \quad (12)$$

where $I(w_i)$ denotes the importance score of the i-th word.

2.3 Bigram Language Model with Semantic Constraints

Yoshikawa et al. (2012) have proposed a set of formulas called Markov Logic Network (MLN) to build a semantically motivated sentence compression model and showed that their model achieves improved performance. We recast their formulas as constraints of our ILP model and incorporate them into the bigram language model. The main idea is to utilize the predicate-argument relations of a sentence and define constraints based on semantic roles to improve the weaknesses of the lexical and syntactical constraints. In this manner, we can ensure that the target compression contains meaningful information. For this purpose, we parse the source sentence semantically using a Semantic Role Labeling (SRL) system (Kingsbury and Palmer, 2002; Hacioglu et al., 2003), ASSERT[6]. When presented with a sentence, ASSERT performs a full syntactic analysis of the sentence, automatically identifies all the verb predicates in that sentence, extracts features for all constituents in the parse tree relative to the predicate, and identifies and tags the constituents with the appropriate semantic arguments. We add the following additional constraints as the semantic constraints to our bigram language model (Section 2.1):

$$\delta_i = 1 \quad (13)$$
$$\forall i : w_i \; is \; a \; predicate$$

[3] http://nlp.cs.nyu.edu/oak/
[4] Available at ftp://ftp.cs.brown.edu/pub/nlparser/
[5] Available at http://www.cis.upenn.edu/ lannie/topicS.html
[6] Available at http://cemantix.org/assert.html

$$\delta_i - \delta_j = 0 \quad (14)$$
$$\forall i, j : w_j \text{ is an argument of predicate } w_i$$

$$\delta_i = 1 \quad (15)$$
$$\forall i : w_i \in [ARG0...ARG5]$$

$$\delta_i = 0 \quad (16)$$
$$\forall i : w_i \in \text{optional arguments}$$

Here, Constraint 13 guarantees that if a word is a predicate, it is included in the compression. Constraint 14 states that if a predicate is in compression, then its argument is also kept in the compression. In Constraint 15, we define that if a word denotes any of the possible semantic roles (i.e. *[ARG0...ARG5]* which are called *mandatory arguments*), it is included in the compression. On the other hand, we use Constraint 16 to restrict the inclusion of optional arguments[7] in the compression.

3 ILP for Query-focused Multi-document Summarization

The query-focused multi-document summarization inference problem can be formulated in terms of ILP. To represent the sentences included in the summary we define a set of indicator variables α_i that are set to 1 if i-th sentence is selected into the summary, and 0 otherwise. Let $Rel(i)$ be the relevance function that returns the relevance score of the i-th sentence. The score of a summary is the sum of the relevance scores of the sentences present in the summary. The inference task is solved by maximizing the overall score of a summary:

$$Maximize \sum_i Rel(i) * \alpha_i$$

such that $\forall i, j$:

$$\alpha_i \in \{0, 1\} \quad (17)$$

$$Sim(i, j) * \left(\alpha_i + \alpha_j\right) \leq K \quad (18)$$

$$\sum_i Len(i) * \alpha_i \leq L \quad (19)$$

We incorporate three constraints into our formulation. The first constraint states that the variables are binary. The second constraint is the redundancy constraint that ensures that only one of the two similar sentences is chosen into the summary. $Sim(i, j)$ function returns a similarity score between the i-th and j-th sentences. Higher scores correspond to higher similarity between a pair of sentences. We assume a threshold K, that sets a tolerance limit to the acceptable similarity score between any two sentences. This value is empirically determined during experiments. The third constraint controls the length of the summary up to a maximum limit, L. $Len(i)$ denotes the length of the i-th sentence in words.

3.1 $Rel(i)$ Function

For each sentence, the $Rel(i)$ function returns a relevance score by combining a set of query-related and importance-oriented measures. The query-related measures calculate the similarity between each sentence and the given query while the importance-oriented measures denote the importance of a sentence in a given document (Chali and Hasan, 2012; Edmundson, 1969; Sekine and Nobata, 2001). For query-related measures, we consider n–gram overlap,

[7] There are some additional arguments or semantic roles that can be tagged by ASSERT. They are called *optional arguments* and they start with the prefix *ARGM*. These are defined by the annotation guidelines set in (Palmer et al., 2005).

longest common subsequence (LCS), weighted LCS, skip-bigram, exact word, synonym, hypernym/hyponym, gloss and basic elements (BE) overlap (Lin, 2004; Zhou et al., 2005) using WordNet (Fellbaum, 1998), and syntactic similarity (Collins and Duffy, 2001; Moschitti and Basili, 2006). To measure the importance of a sentence, we consider its position, length, similarity with topic title, and presence of certain named entities and cue words. The mean of these scores denote the relevance of a sentence.

3.1.1 Query-related Measures

n-gram Overlap n–gram overlap measures the overlapping word sequences between the candidate document sentence and the query sentence (Lin, 2004).

LCS Given two sequences S_1 *and* S_2, the longest common subsequence (LCS) of S_1 *and* S_2 is a common subsequence with maximum length. We use this feature to calculate the longest common subsequence between a candidate sentence and the query.

WLCS Weighted Longest Common Subsequence (WLCS) improves the basic LCS method by remembering the length of consecutive matches encountered so far. Given two sentences X and Y, the WLCS score of X and Y can be computed using the similar dynamic programming procedure as stated in Lin (2004).

Skip-Bigram Skip-bigram measures the overlap of skip-bigrams between a candidate sentence and a query sentence. Skip-bigram counts all in-order matching word pairs while LCS only counts one longest common subsequence.

Exact-word Overlap This is a measure that counts the number of words matching exactly between the candidate sentence and the query sentence.

Synonym Overlap This is the overlap between the list of synonyms of the content words (i.e. nouns, verbs and adjectives) extracted from the candidate sentence and *query related words*[8].

Hypernym/Hyponym Overlap This is the overlap between the list of hypernyms (up to depth 2 in WordNet's hierarchy) and hyponyms (depth 3) of the nouns extracted from the sentence in consideration and *query related words*.

Gloss Overlap This is the overlap between the list of content words that are extracted from the gloss definition of the nouns in the sentence in consideration and *query related words*.

Syntactic Feature The syntactic similarity between the *query* and the *sentence* is calculated using a similar procedure discussed in Section 3.2.2, which gives the similarity score based on syntactic structures.

Basic Element (BE) Overlap We extract BEs (Hovy et al., 2006) for the sentences (or query) by using the BE package distributed by ISI[9]. We compute the Likelihood Ratio (LR) for each BE according to Zhou et al. (2005). We sort the BEs based on LR scores to produce a BE-ranked list. The ranked list contains important BEs at the top which may or may not be relevant to the

[8]To establish the query related words, we took a query and created a set of related queries by replacing its content words by their first-sense synonyms using WordNet.

[9]BE website:http://www.isi.edu/ cyl/BE

complex question. We filter out the BEs that are not related to the query and get the BE overlap score.

3.1.2 Importance-oriented Measures

Position of Sentences Sentences that reside at the start and at the end of a document often tend to include the most valuable information. We manually inspected[10] the given document collection and found that the first and the last 3 sentences of a document often qualify to be considered for this feature. We assign the score 1 to them and 0 to the rest.

Length of Sentences Longer sentences contain more words and have a greater probability of containing valuable information. Therefore, a longer sentence has a better chance of inclusion in a summary[11]. We give the score 1 to a longer sentence and assign the score 0 otherwise. We manually investigated the document collection and set a threshold that a longer sentence should contain at least 11 words.

Title Match If we find a match such as exact word overlap, synonym overlap and hyponym overlap between the title and a sentence, we give it the score 1, otherwise 0.

Named Entity The score 1 is given to a sentence that contains a Named Entity class among: PERSON, LOCATION, ORGANIZATION, GPE (Geo-Political Entity), FACILITY, DATE, MONEY, PERCENT, TIME. We believe that the presence of a Named Entity increases the importance of a sentence. For example, the sentence *"Washington, D.C. is the capital of the United States"* has two named entities (i.e. locations) which denote that the sentence is important. We use the *OAK* System (Sekine, 2002), from New York University for Named Entity recognition.

Cue Word Match The probable relevance of a sentence is affected by the presence of pragmatic words such as "significant", "impossible", "in conclusion", "finally" etc. We use a cue word list of 228 words. We give the score 1 to a sentence having any of the cue words and 0 otherwise.

3.2 $Sim(i, j)$ Function

We employ four alternative redundancy constraints based on different sentence similarity functions (i.e. $Sim(i, j)$) using a) cosine similarity, b) syntactic similarity, c) semantic similarity, and d) extended string subsequence kernel (ESSK).

3.2.1 Cosine Similarity Measure (COS)

The cosine similarity between the respective pair of sentences can be calculated by representing each sentence as a vector of term specific weights (Erkan and Radev, 2004). The term specific weights in the sentence vectors are products of local and global parameters. This is known as term frequency-inverse document frequency (tf-idf) model. The weight vector for a sentence s is $\vec{v_s} = [w_{1,s}, w_{2,s}, \ldots, w_{N,s}]^T$, where,

$$w_{t,s} = tf_t \times \log \frac{|S|}{|\{t \in s\}|}$$

[10] We randomly investigated few newspaper articles and observed that sentences that reside at the start and at the end of a document often tend to include the most valuable information. The "Position of sentences" feature could be tuned to fit other genres of texts as well.

[11] The "Length of sentences" feature was exploited for summarization by extraction in general, which was our motivation to apply different compression models for the task.

Here, tf_t is the term frequency (*tf*) of the term t in a sentence s (a local parameter). $\log \frac{|S|}{|\{t \in s\}|}$ is the inverse document frequency (idf) (a global parameter). $|S|$ is the total number of sentences in the corpus, and $|\{t \in s\}|$ is the number of sentences containing the term t.

3.2.2 Syntactic Similarity Measure (SYN)

Pasca and Harabagiu (2001) demonstrated that with the syntactic form one can see which words depend on other words. Syntactic features have been used successfully so far in *question answering* (Zhang and Lee, 2003; Moschitti et al., 2007; Moschitti and Basili, 2006). Inspired by the potential significance of using syntactic measures for finding similar texts, we get a strong motivation to use it as a redundancy measure in our optimization framework. The first step to calculate the syntactic similarity between two sentences is to parse the corresponding sentences into syntactic trees using the Charniak parser (Charniak, 1999). Once we build the syntactic trees, our next task is to measure the similarity between the trees. For this, every tree T is represented by an m dimensional vector $v(T) = \big(v_1(T), v_2(T), \cdots v_m(T)\big)$, where the i-th element $v_i(T)$ is the number of occurrences of the i-th tree fragment in tree T. The tree fragments of a tree are all of its sub-trees which include at least one production with the restriction that no production rules can be broken into incomplete parts. The tree kernel of two trees T_1 and T_2 is actually the inner product of $v(T_1)$ and $v(T_2)$ (Collins and Duffy, 2001):

$$TK(T_1, T_2) = v(T_1).v(T_2) \tag{20}$$

We define the indicator function $I_i(n)$ to be 1 if the sub-tree i is seen rooted at node n and 0 otherwise. It follows:

$$v_i(T_1) = \sum_{n_1 \in N_1} I_i(n_1)$$

$$v_i(T_2) = \sum_{n_2 \in N_2} I_i(n_2)$$

where, N_1 and N_2 are the set of nodes in T_1 and T_2 respectively. The *TK* (tree kernel) function gives the similarity score between a pair of sentences based on the syntactic structure.

3.2.3 Semantic Similarity Measure (SEM)

Shallow semantic representations can prevent the sparseness of deep structural approaches and the weakness of cosine similarity based models (Moschitti et al., 2007). As an example, PropBank (PB) (Kingsbury and Palmer, 2002) made it possible to design accurate automatic Semantic Role Labeling (SRL) systems (Hacioglu et al., 2003). Therefore, we get the feeling that an application of SRL as a redundancy measure might suit well, since the textual similarity between a pair of sentences relies on a deep understanding of the semantics of both. So, applying semantic similarity measurement as a $Sim(i, j)$ function is another noticeable contribution of this paper. To calculate the semantic similarity between two sentences, we first parse the corresponding sentences semantically using the Semantic Role Labeling (SRL) system, ASSERT. ASSERT is an automatic statistical semantic role tagger, that can annotate naturally occurring text with semantic arguments. We represent the annotated sentences using tree structures that are called semantic trees (ST). In the semantic tree, arguments are replaced with the most important word, often referred to as the semantic head. We look for noun, then verb, then adjective, then adverb to find the semantic head in the argument. If none of these is present, we take the first word of the argument as the semantic head. As in tree kernels (Section 3.2.2), common substructures cannot be composed by a node with only some of its

children as an effective ST representation would require, Moschitti et al. (2007) solved this problem by designing the Shallow Semantic Tree Kernel (SSTK) which allows to match portions of a ST. The SSTK function yields the similarity score between a pair of sentences based on their semantic structures.

3.2.4 Extended String Subsequence Kernel (ESSK)

The ESSK is a simple extension of the Word Sequence Kernel (WSK) (Cancedda et al., 2003) and String Subsequence Kernel (SSK) (Lodhi et al., 2002) that can incorporate semantic information with the use of word senses. In original ESSK, each "alphabet" in SSK is replaced by a disjunction of an "alphabet" and its alternative (word senses) (Hirao et al., 2003). Here, all possible senses of a word are used as the alternatives. However, in our ESSK formulation, we consider each word in a sentence as an "alphabet", and the alternative as its disambiguated sense found through a dictionary based disambiguation approach. We use WordNet to find the semantic relations among the words in a text. We calculate the similarity score $\mathrm{Sim}(T_i, U_j)$ using ESSK where T_i and U_j are the two sentences. Formally, ESSK is defined as follows[12]:

$$K_{essk}(T,U) = \sum_{m=1}^{d} \sum_{t_i \in T} \sum_{u_j \in U} K_m(t_i, u_j)$$

$$K_m(t_i,u_j) = \begin{cases} val(t_i,u_j) & \text{if } m=1 \\ K'_{m-1}(t_i,u_j) \cdot val(t_i,u_j) & \end{cases}$$

Here, $K'_m(t_i,u_j)$ is defined below. t_i and u_j are the nodes of T and U, respectively. The function $val(t,u)$ returns the number of attributes (i.e. words) common to the given nodes t and u.

$$K'_m(t_i,u_j) = \begin{cases} 0 & \text{if } j=1 \\ \lambda K'_m(t_i,u_{j-1}) + K''_m(t_i,u_{j-1}) & \end{cases}$$

Here λ is the decay parameter for the number of skipped words. We choose $\lambda = 0.5$ for this research. $K''_m(t_i,u_j)$ is defined as:

$$K''_m(t_i,u_j) = \begin{cases} 0 & \text{if } i=1 \\ \lambda K''_m(t_{i-1},u_j) + K_m(t_{i-1},u_j) & \end{cases}$$

Finally, the similarity measure is defined after normalization as below:

$$sim_{essk}(T,U) = \frac{K_{essk}(T,U)}{\sqrt{K_{essk}(T,T) K_{essk}(U,U)}}$$

4 Experiments

4.1 Task Description

We consider the query-focused multi-document summarization task defined in the Document Understanding Conference (DUC[13]), 2007. The task is: *"Given a complex question and a collection of relevant documents, the task is to synthesize a fluent, well-organized 250-word summary of the documents that answers the question(s) in the topic"*. We generate 250-word extract summaries for the topics of DUC-2007 using different combinations of sentence compression models (defined in Section 2) and alternative redundancy constraints (Section 3.2). DUC-2007 provided 45 document clusters each containing 25 news articles that came from the

[12]The formulae denotes a dynamic programming technique to compute the ESSK similarity score (Hirao et al., 2004) where d is the vector space dimension i.e. the number of all possible subsequences of up to length d.

[13]http://duc.nist.gov/

AQUAINT corpus, which is comprised of newswire articles from the Associated Press and New York Times (1998-2000) and Xinhua News Agency (1996-2000). As we intend to study if the order of performing compression and extraction can affect the overall performance of the query-focused multi-document summarization, we compose three different models depending on the order to perform sentence compression and extraction: **(1) ComFirst:** In this approach, document sentences are compressed first (using different models as described in Section 2) and then the most relevant compressions are selected to form the summaries (according to Section 3), **(2) SumFirst:** In this approach, we extract the most important sentences first from the source documents (according to Section 3) and then compress them (using different models as described in Section 2) to form the summaries, and **(3) Combined:** Here, we perform compression and extraction jointly by combining the objective functions of Section 2 and Section 3 according to Martins and Smith (2009). Then we optimize the combined objective function to select a small number of most important sentences (from the source documents) whose compressions should be used to form a summary.

4.2 Solving the ILPs

To solve the proposed ILP formulations, we use *lp_solve*[14], a widely used Integer Linear Programming solver that implements Branch-and-Bound algorithm. For summarization, we solve an ILP for each topic in consideration and generate the corresponding query-focused summary. For a document cluster of average size (approximately 510 sentences), the solving process takes under 20 seconds on an Intel Pentium 4, 3.20 GHz desktop machine. For a larger document cluster (of size around 1000 sentences), it takes $90-120$ seconds to solve the ILP. For a smaller document set, the ILP is solved in a few seconds. For compression, we solve an ILP for each sentence in consideration. The solving process takes less than a second per sentence on average for all the compression models. For the joint extraction and compression model, we solve an ILP for each topic in consideration. The solving process is generally slower than solving the ILPs for only sentence extraction or compression as it takes $300-1200$ seconds depending on the document cluster size.

4.3 Evaluation Results and Discussion

4.3.1 Automatic Evaluation

The multiple "reference summaries" given by DUC-2007 are used in the evaluation of our summary content. We carried out the automatic evaluation of our summaries using the ROUGE (Lin, 2004) toolkit. Among different scores reported by ROUGE, unigram-based ROUGE score (ROUGE-1) has been shown to agree with human judgment most (Lin, 2003). We report the widely adopted important ROUGE metrics in the results: ROUGE-1 (unigram),and ROUGE-2 (bigram). The comparison between the systems in terms of their F-scores is given in Table 1. We also include the results of the official baseline systems, the best system (Pingali et al., 2007), and the average ROUGE scores of all the participating systems of DUC-2007. Baseline-1 returns all the leading sentences (up to 250 words) of the most recent document whereas baseline-2's main idea is to ignore the topic narrative while generating summaries using an HMM model[15].

The columns in Table 1 denote the use of alternative redundancy constraints in the optimization

[14] http://lpsolve.sourceforge.net/5.5/
[15] http://duc.nist.gov/pubs/2004papers/ida.conroy.ps

	COS		SYN		SEM		ESSK		No Red.		Comp.	
Model	R1	R2	R1	R2	R1	R2	R1	R2	R1	R2	R1	R2
						ComFirst						
bi	0.359	0.074	0.369	0.078	0.371	0.077	0.368	0.072	0.355	0.060		
topicS	0.372	0.080	0.366	0.081	0.378	0.079	0.373	0.076	0.360	0.071		
bi+sem	0.385	0.093	0.376	0.085	0.389	0.092	0.384	0.088	0.367	0.075		
						SumFirst						
bi	0.368	0.076	0.365	0.079	0.388	0.096	0.370	0.088	0.362	0.071		
topicS	0.374	0.083	0.371	0.084	0.392	0.101	0.378	0.091	0.365	0.074		
bi+sem	0.388	0.096	0.382	0.091	0.405	0.113	0.391	0.101	0.374	0.083		
						Combined						
bi	0.384	0.102	0.371	0.087	0.385	0.091	0.371	0.081	0.356	0.082		
topicS	0.389	0.105	0.374	0.089	0.398	0.103	0.368	0.084	0.364	0.078		
bi+sem	0.412	0.115	0.390	0.092	0.424	0.119	0.395	0.094	0.372	0.086		
No compr.	0.400	0.108	0.399	0.109	0.412	0.111	0.396	0.105	0.381	0.091		
Baseline1											0.334	0.060
Baseline2											0.400	0.093
AverageDUC											0.400	0.095
Best System											0.438	0.122

Table 1: Automatic Evaluation Results: Average ROUGE F-scores

framework whereas the rows stand for the use of different compression models[16]. From these results, we can clearly see the impact of using different sentence compression models on the overall summarization performance. In the **ComFirst** approach, we can see that the bigram model with semantic constraints outperforms all the other alternative models by a clear margin. We can also see the impact of different redundancy constraints on the overall performance. We observe that the use of semantic measure as the redundancy constraint yields the best performance. On the other hand, we see a clear improvement in almost all the scores when we follow the **SumFirst** approach. This phenomenon suggests that compressing the document sentences at the beginning often tend to reduce relevant information in the sentences for which we get lesser similarity matching when we calculate the relevance scores according to Section 3.1. In the **Combined** approach, we achieve better summarization performance than the other two approaches which denotes that the overall summary quality can be improved if a global optimization framework is utilized having a joint compression and extraction model. Again, we see that the bigram language model with semantic constraints along with the semantic redundancy constraint (used in the summarization model) yields the best performance. We also report the results of a *"No compression"* and a *"No redundancy"* baseline. Comparisons with these baselines also suggest that our bigram compression model with semantic constraints can improve the overall summarization performance if a **Combined** optimization framework is used in presence of **COS** or **SEM** redundancy constraints. These results also demonstrate that the absence of a redundancy constraint in the ILP framework for summarization really hurts the overall quality of the summaries. We also compare the scores of our model with the state-of-the-art systems of DUC-2007. From the results, we see that our semantically motivated models can mostly outperform the **DUC baselines** and the **AverageDUC** scores to show a clear improvement in the overall summarization performance while achieving a comparable performance with respect to the DUC-2007 best system. The differences between the models are computed

[16]The last few rows and columns are used to accommodate the scores of the baselines and the state-of-the-art systems.

to be statistically significant at $p < 0.05$ (using Student's t-test) except for the differences between **topicSig+SYN** and **bigram+SYN**, and **topicSig+ESSK** and **bigram+ESSK** in all the three approaches, between **topicSig+COS** and **bigram+COS** in the **Combined** approach, and between **"bigram+sem"+SEM** and **DUC Best System** in the **Combined** approach.

4.3.2 Manual Evaluation

One of the important demerits of using sentence compression models is that they can degrade the linguistic quality of a summary by showing poor compression performance. ROUGE is not reliable to some researchers as there might be some linguistically bad summaries that get state-of-the-art ROUGE scores (Sjöbergh, 2007). So, we conduct an extensive manual evaluation in order to analyze the effectiveness of our approaches. Two self reported native English-speaking university graduate students judge the summaries for linguistic quality and overall responsiveness according to the DUC-2007 evaluation guidelines[17]. The given linguistic quality score is an integer between 1 (very poor) and 5 (very good) and is guided by consideration of the following factors: 1. Grammaticality, 2. Non-redundancy, 3. Referential clarity, 4. Focus, and 5. Structure and Coherence. The responsiveness score is also an integer between 1 (very poor) and 5 (very good) and is based on the amount of information in the summary that helps to satisfy the information need. The carried out user evaluation was subjective in nature specially while judging referential clarity, focus, coherence and overall responsiveness of the summaries. The inter-annotator agreement of Cohen's $\kappa = 0.43$ (Cohen, 1960) was computed that denotes a moderate degree of agreement (Landis and Koch, 1977) between the raters. Table 2 presents the average linguistic quality and overall responsive scores of all the systems. From these results, we can see that the use of different sentence compression models has a negative impact on the overall linguistic quality of the summaries. The reason behind this is that our bigram compression models were less aware of the underlying context in a sentence and hence, some word deletions resulted a loss in focus and coherence of the overall summaries. However, we observe that the semantically motivated models are showing an improved summarization performance; also, their overall responsiveness scores are comparable to the state-of-the-art systems. This suggests that the manual evaluation results are corresponding well to the automatic evaluation results. Considering the work of Gillick and Favre (2009) for a relative comparison, we find that both our automatic and manual evaluation results are corresponding fairly well to their results obtained on the TAC[18]-2008 data. Their ILP model with additional constraints to include sentence compression achieved an improvement in ROUGE-2 score over the "no compression" alternative while having reductions in manual evaluation scores. We perform a statistical significance test on our manual evaluation results at $p < 0.05$ using Student's t-test. The differences between the models are statistically significant except for the differences between **topicSig+COS** and **bigram+COS**, and **topicSig+SYN** and **bigram+SYN** in all the three approaches. The manual evaluation results also demonstrate that the use of different redundancy constraints certainly affects the overall performance of the proposed optimization framework for summarization[19]. From these experiments we can conclude that the semantic similarity measure can be used effectively as the $Sim(i, j)$ function to improve the performance of the traditional cosine similarity based approaches. We plan to make our created resources available to the scientific community.

[17] http://www-nlpir.nist.gov/projects/duc/duc2007/quality-questions.txt

[18] Text Analysis Conference, http://www.nist.gov/tac/

[19] The selection of sentences in the optimal summaries varied due to different redundancy measures, hence, the linguistic quality scores also varied to reflect the differences in coherence, redundancy etc.

	COS		SYN		SEM		ESSK		No Redundancy		Comparison	
Models	LQ	Res.	LQ	Res.	LQ	Res.	LQ	Res.	LQ	Res.	LQ	Res.
ComFirst												
bigram	2.10	2.12	2.28	2.20	2.44	2.21	2.32	2.25	1.94	2.10		
topicSig	2.14	2.30	2.45	2.27	2.48	2.78	2.39	2.46	2.08	2.26		
bigram+sem	2.42	2.56	2.55	2.61	2.74	3.05	2.54	2.80	2.25	2.58		
SumFirst												
bigram	2.43	2.44	2.54	2.50	2.60	2.45	2.25	2.34	2.16	2.56		
topicSig	2.48	2.56	2.65	2.69	2.72	2.66	2.48	2.55	2.27	2.68		
bigram+sem	2.61	2.76	2.88	2.78	3.20	3.56	2.75	2.93	2.42	2.62		
Combined												
bigram	2.54	2.62	2.52	2.31	2.76	2.55	2.36	2.50	1.98	2.20		
topicSig	2.62	2.75	2.68	2.38	2.80	2.62	2.45	2.64	2.14	2.31		
bigram+sem	2.85	3.08	2.91	2.93	3.18	3.61	2.77	2.88	2.32	2.42		
No compression	3.30	3.38	3.42	3.15	3.64	3.50	3.38	3.21	2.28	2.15		
Baseline1											4.24	1.86
Baseline2											4.48	2.71
Best System											4.11	3.40

Table 2: Average linguistic quality (LQ) and responsiveness scores (Res.)

Conclusion and Future Work

We have analyzed the effectiveness of using different ILP-based sentence compression models for the task of query-focused multi-document summarization. Our empirical evaluation suggested that the semantically motivated sentence compression models can enhance the overall summarization performance in presence of the semantic redundancy constraint in the summarization model and this can be achieved irrespective of the compression and extraction order followed during the process. Our results also demonstrated that a combined optimization framework of compression and extraction can achieve better performance than the other two considered approaches effectively. We also found that the *SumFirst* approach shows superior performance to that of the *ComFirst* approach suggesting the fact that extracting the most important sentences before compression is a more effective way of summarization. We have also used different textual similarity measurement techniques as the redundancy constraints of the ILP-based summarization framework and performed an extensive experimental evaluation to show their impact on the overall summarization performance. Experimental results showed that the use of semantic similarity measure as the $Sim(i, j)$ function in the redundancy constraint yields the best performance. Overall, our global optimization frameworks showed promising performance with respect to the state-of-the-art systems. We look forward to apply our approach to other available datasets of DUC-2005 and DUC-2006. The findings should hold for these datasets as well as for other genres of datasets since we believe that our ILP-based compression and summarization models could be tuned to fit them. We also plan to use other automatic measures (Saggion et al., 2010; Pitler et al., 2010) to evaluate our approach.

Acknowledgments

The research reported in this paper was supported by the Natural Sciences and Engineering Research Council (NSERC) of Canada – discovery grant and the University of Lethbridge.

References

Berg-Kirkpatrick, T., Gillick, D., and Klein, D. (2011). Jointly Learning to Extract and Compress. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1*, HLT '11, pages 481–490. ACL.

Bouayad-Agha, N., Gil, A., Valentin, O., and Pascual, V. (2006). A Sentence Compression Module for Machine-Assisted Subtitling. In *Computational Linguistics and Intelligent Text Processing*, pages 490–501. Springer Berlin Heidelberg.

Cancedda, N., Gaussier, E., Goutte, C., and Renders, J. M. (2003). Word Sequence Kernels. *Journal of Machine Learning Research*, 3:1059–1082.

Carbonell, J. and Goldstein, J. (1998). The Use of MMR, Diversity-based Reranking for Reordering Documents and Producing Summaries. In *Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 1998)*, pages 335–336, Melbourne, Australia.

Chali, Y. and Hasan, S. A. (2012). Query-focused Multi-document Summarization: Automatic Data Annotations and Supervised Learning Approaches. *Natural Language Engineering*, 18(1):109–145.

Charniak, E. (1999). A Maximum-Entropy-Inspired Parser. In *Technical Report CS-99-12*, Brown University, Computer Science Department.

Clarke, J. and Lapata, M. (2008). Global Inference for Sentence Compression An Integer Linear Programming Approach. *Journal of Artificial Intelligence Research*, 31(1):399–429.

Cohen, J. (1960). A Coefficient of Agreement for Nominal Scales. *Educational and Psychological Measurement*, 20(1):37–46.

Cohn, T. and Lapata, M. (2008). Sentence Compression Beyond Word Deletion. In *Proceedings of the 22nd International Conference on Computational Linguistics (Coling 2008)*, pages 137–144, Manchester, UK.

Collins, M. and Duffy, N. (2001). Convolution Kernels for Natural Language. In *Proceedings of Neural Information Processing Systems*, pages 625–632, Vancouver, Canada.

Daumé III, H. and Marcu, D. (2005). Bayesian Multi-Document Summarization at MSE. In *Proceedings of the Workshop on Multilingual Summarization Evaluation (MSE)*.

Denis, P. and Baldridge, J. (2007). Joint Determination of Anaphoricity and Coreference Resolution using Integer Programming. In *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics;*, pages 236–243. ACL.

Edmundson, H. P. (1969). New Methods in Automatic Extracting. *Journal of the Association for Computing Machinery (ACM)*, 16(2):264–285.

Erkan, G. and Radev, D. R. (2004). LexRank: Graph-based Lexical Centrality as Salience in Text Summarization. *Journal of Artificial Intelligence Research*, 22:457–479.

Fellbaum, C. (1998). WordNet - An Electronic Lexical Database. Cambridge, MA. MIT Press.

Filippova, K. (2010). Multi-Sentence Compression: Finding Shortest Paths in Word Graphs. In *Proceedings of the 23rd International Conference on Computational Linguistics*, pages 322–330. ACL.

Galanis, D. and Androutsopoulos, I. (2010). An Extractive Supervised Two-Stage Method for Sentence Compression. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, HLT '10, pages 885–893. ACL.

Gillick, D. and Favre, B. (2009). A Scalable Global Model for Summarization. In *Proceedings of the Workshop on Integer Linear Programming for Natural Langauge Processing*, ILP '09, pages 10–18. ACL.

Goldstein, J., Kantrowitz, M., Mittal, V., and Carbonell, J. (1999). Summarizing Text Documents: Sentence Selection and Evaluation Metrics. In *Proceedings of the 22nd International ACM Conference on Research and Development in Information Retrieval, SIGIR*, pages 121–128, Berkeley, CA.

Hacioglu, K., Pradhan, S., Ward, W., Martin, J. H., and Jurafsky, D. (2003). Shallow Semantic Parsing Using Support Vector Machines. In *Technical Report TR-CSLR-2003-03*, University of Colorado.

Hirao, T., , Suzuki, J., Isozaki, H., and Maeda, E. (2004). Dependency-based Sentence Alignment for Multiple Document Summarization. In *Proceedings of COLING 2004*, pages 446–452, Geneva, Switzerland. COLING.

Hirao, T., Suzuki, J., Isozaki, H., and Maeda, E. (2003). NTT's Multiple Document Summarization System for DUC2003. In *Proceedings of the Document Understanding Conference*.

Hovy, E., Lin, C. Y., Zhou, L., and Fukumoto, J. (2006). Automated Summarization Evaluation with Basic Elements. In *Proceedings of the Fifth Conference on Language Resources and Evaluation*, Genoa, Italy.

Jing, H. (2000). Sentence Reduction for Automatic Text Summarization. In *Proceedings of the Sixth Conference on Applied Natural Language Processing*, pages 310–315. ACL.

Kingsbury, P. and Palmer, M. (2002). From Treebank to PropBank. In *Proceedings of the International Conference on Language Resources and Evaluation*, Las Palmas, Spain.

Knight, K. and Marcu, D. (2000). Statistics-Based Summarization - Step One: Sentence Compression. In *Proceedings of the Seventeenth National Conference on Artificial Intelligence and Twelfth Conference on Innovative Applications of Artificial Intelligence*, pages 703–710. AAAI Press.

Knight, K. and Marcu, D. (2002). Summarization Beyond Sentence Extraction: A Probabilistic Approach to Sentence Compression. *Artificial Intelligence*, 139(1):91–107.

Kupiec, J., Pedersen, J., and Chen, F. (1995). A Trainable Document Summarizer. In *Proceedings of the 18th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 1995)*, pages 68–73, Seattle, Washington, USA.

Landis, J. R. and Koch, G. G. (1977). The Measurement of Observer Agreement for Categorical Data. *Biometrics*, 33(1):159–174.

Lin, C. Y. (2003). Improving Summarization Performance by Sentence compression: A Pilot Study. In *Proceedings of the sixth international workshop on Information retrieval with Asian languages - Volume 11*, pages 1–8. ACL.

Lin, C. Y. (2004). ROUGE: A Package for Automatic Evaluation of Summaries. In *Proceedings of Workshop on Text Summarization Branches Out, Post-Conference Workshop of Association for Computational Linguistics*, pages 74–81, Barcelona, Spain.

Lin, C.-Y. and Hovy, E. H. (2000). The Automated Acquisition of Topic Signatures for Text Summarization. In *Proceedings of the 18th conference on Computational linguistics*, pages 495–501.

Lodhi, H., Saunders, C., Shawe-Taylor, J., Cristianini, N., and Watkins, C. (2002). Text Classification using String Kernels. *Journal of Machine Learning Research*, 2:419–444.

Madnani, N., Zajic, D., Dorr, B., Ayan, N. F., and Lin, J. (2007). Multiple Alternative Sentence Compressions for Automatic Text Summarization. In *In Proceedings of the 2007 Document Understanding Conference (DUC-2007) at NLT/NAACL 2007*.

Mani, I. and Maybury, M. (1999). *Advances in Automatic Text Summarization*. MIT Press.

Martins, A. F. T. and Smith, N. A. (2009). Summarization with a Joint Model for Sentence Extraction and Compression. In *Proceedings of the Workshop on Integer Linear Programming for Natural Langauge Processing*, ILP '09, pages 1–9. ACL.

Mcdonald, R. (2006). Discriminative Sentence Compression with Soft Syntactic Constraints. In *In Proceedings of the 11th Conference of the EACL*.

McDonald, R. (2007). A Study of Global Inference Algorithms in Multi-document Summarization. In *Proceedings of the 29th European conference on IR research*, ECIR'07, pages 557–564. Springer-Verlag.

Molina, A., Torres-Moreno, J., SanJuan, E., da Cunha, I., Sierra, G., and Velázquez-Morales, P. (2011). Discourse Segmentation for Sentence Compression. In *Proceedings of the 10th Mexican international conference on Advances in Artificial Intelligence - Volume Part I*, pages 316–327. Springer-Verlag.

Moschitti, A. and Basili, R. (2006). A Tree Kernel Approach to Question and Answer Classification in Question Answering Systems. In *Proceedings of the 5th International Conference on Language Resources and Evaluation*, Genoa, Italy.

Moschitti, A., Quarteroni, S., Basili, R., and Manandhar, S. (2007). Exploiting Syntactic and Shallow Semantic Kernels for Question/Answer Classificaion. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 776–783, Prague, Czech Republic. ACL.

Palmer, M., Gildea, D., and Kingsbury, P. (2005). The Proposition Bank: An Annotated Corpus of Semantic Roles. *Computational Linguistics*, 31:71–106.

Pasca, M. and Harabagiu, S. M. (2001). Answer Mining from On-Line Documents. In *Proceedings of the Association for Computational Linguistics 39th Annual Meeting and 10th Conference of the European Chapter Workshop on Open-Domain Question Answering*, pages 38–45, Toulouse, France.

Pingali, P., K., R., and Varma, V. (2007). IIIT Hyderabad at DUC 2007. In *Proceedings of the Document Understanding Conference*, Rochester. NIST.

Pitler, E., Louis, A., and Nenkova, A. (2010). Automatic Evaluation of Linguistic Quality in Multi-document Summarization. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 544–554. ACL.

Punyakanok, V., Roth, D., Yih, W., and Zimak, D. (2004). Semantic Role Labeling via Integer Linear Programming Inference. In *Proceedings of the 20th international conference on Computational Linguistics*, COLING '04. ACL.

Riedel, S. and Clarke, J. (2006). Incremental Integer Linear Programming for Non-projective Dependency Parsing. In *In EMNLP*, pages 129–137.

Roth, D. and Yih, W. (2004). A Linear Programming Formulation for Global Inference in Natural Language Tasks. In *In Proceedings of CoNLL-2004*, pages 1–8.

Saggion, H., Torres-Moreno, J., Cunha, I., and SanJuan, E. (2010). Multilingual Summarization Evaluation without Human Models. In *Proceedings of the 23rd International Conference on Computational Linguistics: Posters*, pages 1059–1067. ACL.

Sekine, S. (2002). Proteus Project OAK System (English Sentence Analyzer), http://nlp.nyu.edu/oak.

Sekine, S. and Nobata, C. A. (2001). Sentence Extraction with Information Extraction Technique. In *Proceedings of the Document Understanding Conference (DUC 2001)*, New Orleans, Louisiana, USA.

Sjöbergh, J. (2007). Older Versions of the ROUGEeval Summarization Evaluation System Were Easier to Fool. *Information Processing and Management*, 43:1500–1505.

Yoshikawa, K., Iida, R., Hirao, T., and Okumura, M. (2012). Sentence Compression with Semantic Role Constraints. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 349–353, Jeju Island, Korea. ACL.

Zajic, D., Dorr, B. J., Lin, J., and Schwartz, R. (2007). Multi-candidate Reduction: Sentence Compression as a Tool for Document Summarization Tasks. *Information Processing and Management*, 43(6):1549–1570.

Zhang, A. and Lee, W. (2003). Question Classification using Support Vector Machines. In *Proceedings of the Special Interest Group on Information Retrieval*, pages 26–32, Toronto, Canada. ACM.

Zhou, L., Lin, C. Y., and Hovy, E. (2005). A BE-based Multi-dccument Summarizer with Query Interpretation. In *Proceedings of Document Understanding Conference*, Vancouver, B.C., Canada.

Towards Automatic Topical Question Generation

Yllias Chali *Sadid A. Hasan*
University of Lethbridge, Lethbridge, AB, Canada
chali@cs.uleth.ca, hasan@cs.uleth.ca

ABSTRACT
We address the challenge of automatically generating questions from topics. We consider that each topic is associated with a body of texts containing useful information about the topic. Questions are generated by exploiting the named entity information and the predicate argument structures of the sentences present in the body of texts. To measure the importance of the generated questions, we use Latent Dirichlet Allocation (LDA) to identify the sub-topics (which are closely related to the original topic) in the given body of texts and apply the Extended String Subsequence Kernel (ESSK) to calculate their similarity with the questions. We also propose the use of syntactic tree kernels for computing the syntactic correctness of the questions. The questions are ranked by considering their importance (in the context of the given body of texts) and syntactic correctness. To the best of our knowledge, no other study has accomplished this task in our setting before. Experiments show that our approach can significantly outperform the state-of-the-art results.

KEYWORDS: Question generation, named entity information, predicate argument structures, latent dirichlet allocation (LDA), extended string subsequence kernel (ESSK), syntactic tree kernel.

Proceedings of COLING 2012: Technical Papers, pages 475–492,
COLING 2012, Mumbai, December 2012.

1 Introduction

When a user is served with a ranked list of relevant documents by the standard document retrieval systems (i.e. search engines), his/her search task is usually not over (Chali et al., 2009b). The next step for him/her is to look into the documents themselves and search for the precise piece of information he/she was looking for. This method is time consuming, and a correct answer could easily be missed, by either an incorrect query resulting in missing documents or by careless reading. This is why, Question Answering (QA) has received immense attention from the information retrieval, information extraction, machine learning, and natural language processing communities (Kotov and Zhai, 2010). One of the main requirements of a QA system is that it must receive a well-formed question as input in order to come up with the best possible correct answer as output. Available studies revealed that humans are not very skilled in asking good questions about a topic of their interest. They are forgetful in nature which often restricts them to properly express whatever that is peeking in their mind. Therefore, they would benefit from automated Question Generation (QG) systems that can assist in meeting their inquiry needs (Olney et al., 2012; Ali et al., 2010; Kotov and Zhai, 2010; Rus and Graesser, 2009; Lauer et al., 1992; Graesser et al., 2001). Question asking and Question Generation are important components in advanced learning technologies such as intelligent tutoring systems, and inquiry-based environments (Graesser et al., 2001). A QG system would be useful for building better question asking facilities in intelligent tutoring systems. Another benefit of QG is that it can be a good tool to help improve the quality of the Question Answering (QA) systems (Graesser et al., 2001; Rus and Graesser, 2009).

The main motivation of this work is to generate all possible questions about a given topic. For example, given the topic *"Apple Inc. Logos"*, we can generate questions such as *"What is Apple Inc.?"*, *"Where is Apple Inc. located?"*, *"Who designed Apple's Logo?"* etc. We consider this task of automatically generating questions from topics and assume that each topic is associated with a body of texts having useful information about the topic. Our main goal is to generate fact-based questions[1] about a given topic from its associated content information. We generate questions by exploiting the named entity information and the predicate argument structures of the sentences (along with semantic roles) present in the given body of texts. The named entities and the semantic role labels are used to identify relevant parts of a sentence in order to form relevant questions over them. The importance of the generated questions is measured in two steps. In the first step, we identify whether the question is asking something about the topic or something that is very closely related to the topic. We call this the measure of *topic relevance*. For this purpose, we use Latent Dirichlet Allocation (LDA) (Blei et al., 2003) to identify the sub-topics (which are closely related to the original topic) in the given body of texts and apply the Extended String Subsequence Kernel (ESSK) (Hirao et al., 2003) to calculate their similarity with the questions. In the second step, we judge the syntactic correctness of each generated question. We apply the tree kernel functions (Collins and Duffy, 2001) and re-implement the syntactic tree kernel model according to Moschitti et al. (2007) for computing the syntactic similarity of each question with the associated content information. We rank the questions by considering their topic relevance and syntactic correctness scores. Experimental results show the effectiveness of our approach for automatically generating topical questions. The remainder of the paper is organized as follows. Section 2 describes the related work and motivation followed by Section 3 that presents the description of our QG system. Section 4 explains the experiments and shows evaluation results. We conclude the paper in the next section.

[1] We mainly focus on generating *Who, What, Where, Which, When, Why and How* questions in this research.

2 Related Work and Motivation

Recently, question generation has got immense attention from the researchers and hence, different methods have been proposed to accomplish the task in different relevant fields (Andrenucci and Sneiders, 2005). McGough et al. (2001) proposed an approach to build a web-based testing system with the facility of dynamic question generation. Wang et al. (2008) showed a method to automatically generate questions based on question templates (which are created from training on medical articles). Brown et al. (2005) described an approach to automatically generate questions to assess the user's vocabulary knowledge. To mimic the reader's self-questioning strategy during reading, Chen et al. (2009) developed a method to generate questions automatically from informational text. On the other hand, Agarwal et al. (2011) considered the question generation problem beyond sentence level and proposed an approach that uses discourse connectives to generate questions from a given text. Several other QG models have been proposed over the years that deal with transforming answers to questions and utilizing question generation as an intermediate step in the question answering process (Echihabi and Marcu, 2003; Hickl et al., 2005). There are some other researchers who have approached the task of generating questions for educational purposes (Mitkov and Ha, 2003; Heilman and Smith, 2010b).

The Natural Language Processing (NLP), Natural Language Generation (NLG), Intelligent Tutoring System, and Information Retrieval (IR) communities have currently identified the Text-to-Question generation task as promising candidates for shared tasks[2] (Rus and Graesser, 2009; Boyer and Piwek, 2010). In the Text-to-Question generation task, a QG system is given a text, and the goal is to generate a set of questions for which the text contains answers. The task of generating a question about a given text can be typically decomposed into three subtasks. First, given the source text, a content selection step is necessary to select a target to ask about, such as the desired answer. Second, given a target answer, an appropriate question type is selected, i.e., the form of question to ask is determined. Third, given the content, and question type, the actual question is constructed. Based on this principle, several approaches have been described in Boyer and Piwek (2010) that use named entity information, syntactic knowledge and semantic structures of the sentences to perform the task of generating questions from sentences and paragraphs (Heilman and Smith, 2010a; Mannem et al., 2010). Inspired by these works, we perform the task of topic to question generation using named entity information and semantic structures of the sentences. A task that is similar to ours is the task of keywords to question generation that has been addressed recently in Zheng et al. (2011). They propose a user model for jointly generating keywords and questions. However, their approach is based on generating question templates from existing questions which requires a large set of English questions as training data. In recent years, some other related researches have proposed the tasks of high quality question generation (Ignatova et al., 2008) and generating questions from queries (Lin, 2008). Fact-based question generation has been accomplished previously by Rus et al. (2007); Heilman and Smith (2010b). We also focus on generating fact-based questions in this research.

Besides grammaticality, an effective QG system should focus deeply on the importance of the generated questions (Vanderwende, 2008). This motivates the use of a question ranking module in a typical QG system. Over-generated questions can be ranked using different approaches such as statistical ranking methods, dependency parsing, identifying the presence of pronouns and

[2] http://www.questiongeneration.org/QGSTEC2010

named entities, and topic scoring (Heilman and Smith, 2010a; Mannem et al., 2010; McConnell et al., 2011). However, most of these automatic ranking approaches ignore the aspects of complex paraphrasing by not considering lexical semantic variations (e.g. synonymy) while measuring the importance of the questions. In our work, we use Latent Dirichlet Allocation (LDA) (Blei et al., 2003) to identify the sub-topics (which are closely related to the original topic) in the given body of texts. In recent years, LDA has become one of the most popular topic modeling techniques and has been shown to be effective in several text-related tasks such as document classification, information retrieval, and question answering (Misra et al., 2008; Wei and Croft, 2006; Celikyilmaz et al., 2010). Hirao et al. (2003) introduced ESSK considering all possible senses to each word to perform their summarization task. Their method is effective. However, the fact that they do not disambiguate word senses cannot be disregarded. In our task, we apply ESSK to calculate the similarity between important topics (discovered using LDA) and the generated questions in order to measure the importance of each question. We use disambiguated word senses for this purpose.

Syntactic information has been used successfully in *question answering* previously (Chali et al., 2009a, 2011; Zhang and Lee, 2003; Moschitti et al., 2007; Moschitti and Basili, 2006). Pasca and Harabagiu (2001) argued that with the syntactic form of a sentence one can see which words depend on other words. We also feel that there should be a similarity between the words which are dependent in the sentences present in the associated body of texts and the dependency between words of the generated question. This motivates us to propose the use of syntactic kernels in judging the syntactic correctness of the generated questions automatically.

The main goal of our work is to generate as many questions as possible related to the topic. We use NE information and the predicate argument structures of the sentences to accomplish this goal. Our approach is different from the setup in shared tasks (Rus and Graesser, 2009; Boyer and Piwek, 2010) as we generate a set of basic questions which are useful to add variety in the question space. A paragraph associated with each topic is used as the source of relevant information about the topic. We evaluate our systems in terms of topic relevance which is different from the prior works (Heilman and Smith, 2010a; Mannem et al., 2010). Syntactic correctness is also an important property of a good question. For this reason, we evaluate our system in terms of syntactic correctness as well. The proposed system will be useful to generate topic related questions from the associated content information which can be used to incorporate a "question suggestions for a certain topic" facility in the search systems. For example, if a user searches for some information related to a certain topic, the search system could generate all possible topic-relevant questions from a preexistent related body of texts to provide suggestions. Kotov and Zhai (2010) approached a similar task by proposing a technique to augment the standard ranked list presentation of search results with a question based interface to refine user given queries.

The major contributions of our work can be summarized as follows:

- We perform the task of topic to question generation which can help users in expressing their information needs. Questions are generated using a set of general-purpose rules based on named entity information and the predicate argument structures of the sentences (along with semantic roles) present in the associated body of texts.
- We use LDA to identify the sub-topics (which are closely related to the original topic) in the given body of texts and apply ESSK (with disambiguated word senses) to calculate

their similarity with the questions. This helps us to measure the importance of each question.

- We apply the tree kernel functions and re-implement the syntactic tree kernel model for computing the syntactic similarity of each question with the associated content information. In this way, we judge the syntactic correctness of each generated question automatically.
- The ESSK similarity scores and the syntactic similarity scores are used to rank the generated questions. In doing so, we show that the use of ESSK and syntactic kernels improve the relevance and the syntactic correctness of the top-ranked questions, respectively.
- We also run experiments by narrowing down the topic focus. Experiments with the topics about persons (biographical focus) reveal improvements in the overall results.

3 Topic to Question Generation

Our QG approach mainly builds on four steps. In the first step, complex sentences (from the given body of texts) related to a topic are simplified as it is easier to generate questions from simple sentences. In the next step, named entity information and predicate argument structures of the sentences are extracted and then, questions are generated using them. In the third step, LDA is used to identify important sub-topics from the given body of texts and then ESSK is applied to find their similarity with the generated questions. In the final step, syntactic tree kernel is employed and syntactic similarity between the generated questions and the sentences present in the body of texts determines the syntactic correctness of the questions. Questions are then ranked by considering the ESSK similarity scores and the syntactic similarity scores. We describe the overall procedure in the following subsections.

3.1 Sentence Simplification

Sentences may have complex grammatical structure with multiple embedded clauses. Therefore, we simplify the complex sentences with the intention to generate more accurate questions. We use the simplified factual statement extractor model[3] of Heilman and Smith (2010a). Their model extracts the simpler forms of the complex source sentence by altering lexical items, syntactic structure, and semantics and by removing phrase types such as leading conjunctions, sentence-level modifying phrases, and appositives. For example, given a complex sentence *s*, we get a corresponding simple sentence as follows:

Complex Sentence (s): Apple's first logo, designed by Jobs and Wayne, depicts Sir Isaac Newton sitting under an apple tree.

Simple Sentence: Apple's first logo is designed by Jobs and Wayne.

3.2 Named Entity (NE) Information and Semantic Role Labeling (SRL) for QG

We use the Illinois Named Entity Tagger[4], a state of the art NE tagger that tags a plain text with named entities (people, organizations, locations, miscellaneous) (Ratinov and Roth, 2009).

[3] Available at http://www.ark.cs.cmu.edu/mheilman/
[4] Available at http://cogcomp.cs.illinois.edu/

Once we tag the topic in consideration and its associated body of texts, we use some general purpose rules to create some basic questions even though the answer is not present in the body of texts. For example, "Apple Inc." is tagged as an organization, so we generate a question: "Where is Apple Inc. located?". The main motivation behind generating such questions is to add variety to the generated question space. Table 1 shows some example rules for basic questions generated in this work.

Tag	Example Question
person	Who is *person*?
organization	Where is *organization* located?
location	Where is *location*?
misc.	What do you know about *misc.*?

Table 1: Example basic question rules

Our next task is to generate specific questions from the sentences present in the given body of texts. For this purpose, we parse the sentences semantically using a Semantic Role Labeling (SRL) system (Kingsbury and Palmer, 2002; Hacioglu et al., 2003), ASSERT[5]. ASSERT is an automatic statistical semantic role tagger, that can annotate naturally occuring text with semantic arguments. When presented with a sentence, it performs a full syntactic analysis of the sentence, automatically identifies all the verb predicates in that sentence, extracts features for all constituents in the parse tree relative to the predicate, and identifies and tags the constituents with the appropriate semantic arguments. For example, the output of the SRL system for the sentence "Apple's first logo is designed by Jobs and Wayne." is: [ARG1 Apple 's first logo] is [TARGET designed] [ARG0 by Jobs and Wayne]. The output contains one verb (predicate) with its arguments (i.e. semantic roles). These arguments are used to generate specific questions from the sentences. For example, we can replace *[ARG1 ..]* with *What* and generate a question as: "What is designed by Jobs and Wayne?". Similarly, *[ARG0 ..]* can be replaced and the question: "Who designed Apple's first logo?" can be generated. The semantic roles *ARG0...ARG5* are called *mandatory arguments*. There are some additional arguments or semantic roles that can be tagged by ASSERT. They are called *optional arguments* and they start with the prefix *ARGM*. These are defined by the annotation guidelines set in (Palmer et al., 2005). A set of about 350 general purpose rules are used to transform the semantic-role labeled sentences into the questions. The rules were set up in a way that we could use the semantic role information to find the potential answer words in a sentence which would be replaced by suitable question words. In case of a mandatory argument, the choice of question word depends on the argument's named entity tag (e.g. "Who" for a person, "Where" for a location etc.). Table 2 shows how different semantic roles can be replaced by possible question words in order to generate a question.

3.3 Importance of Generated Questions

3.3.1 Latent Dirichlet Allocation (LDA)

To measure the importance of the generated questions, we use LDA (Blei et al., 2003) to identify the important sub-topics from the given body of texts. LDA is a probabilistic topic modeling technique where the main principle is to view each document as a mixture of various topics.

[5]Available at http://cemantix.org/assert.html

Arguments	Question Words
ARG0...ARG5	Who, Where, What, Which
ARGM-ADV	In what circumstances
ARGM-CAU	Why
ARGM-DIS	How
AGRM-EXT	To what extent
ARGM-LOC	Where
ARGM-MNR	How
ARGM-PNC	Why
ARGM-TMP	When

Table 2: Semantic roles with possible question words

Here each topic is a probability distribution over words. LDA assumes that documents are made up of words and word ordering is not important ("bag-of-words" assumption) (Misra et al., 2008). The main idea is to choose a distribution over topics while generating a new document. For each word in the new document, a topic is randomly chosen according to this distribution and a word is drawn from that topic. LDA uses a generative topic modeling approach to specify the following distribution over words within a document:

$$P(w_i) = \sum_{j=1}^{K} P(w_i|z_i = j)P(z_i = j) \tag{1}$$

where K is the number of topics, $P(w_i|z_i = j)$ is the probability of word w_i under topic j and $P(z_i = j)$ is the sampling probability of topic j for the i^{th} word. The multinomial distributions $\phi^{(j)} = P(w|z_i = j)$ and $\theta^{(d)} = P(z)$ are termed as topic-word distribution and document-topic distribution, respectively (Blei et al., 2003). A Dirichlet (α) prior is placed on θ and a Dirichlet (β) prior is set on ϕ to refine this basic model (Blei et al., 2003; Griffiths and Steyvers, 2002). Now the main goal is to estimate the two parameters: θ and ϕ. We apply this framework directly to solve our problem by considering each topic-related body of texts as a document. We use a GUI-based toolkit for topic modeling[6] that uses the popular MALLET (McCallum, 2002) toolkit for the back-end. The process starts by removing a list of "stop words" from the document and runs 200 iterations of Gibbs sampling (Geman and Geman, 1984) to estimate the parameters: θ and ϕ. From each body of texts, we discover K topics and choose the most frequent words from the most likely unigrams as the desired sub-topics. For example, from the associated body of texts of the topic *Apple Inc. Logos*, we get these sub-topics: *janoff, themes, logo, color, apple*.

3.3.2 Extended String Subsequence Kernel (ESSK)

Once we identify the sub-topics, we apply ESSK to measure their similarity with the generated questions. ESSK is the simple extension of the Word Sequence Kernel (WSK) (Cancedda et al., 2003) and String Subsequence Kernel (SSK) (Lodhi et al., 2002). WSK receives two sequences of words as input and maps each of them into a high-dimensional vector space. WSK's value is just the inner product of the two vectors. But, WSK disregards synonyms, hyponyms, and hypernyms. On the other hand, SSK measures the similarity between two sequences of

[6]Available at http://code.google.com/p/topic-modeling-tool/

"alphabets". In ESSK, each "alphabet" in SSK is replaced by a disjunction of an "alphabet" and its alternative (Hirao et al., 2003). In ESSK, each word in a sentence is considered an "alphabet", and the alternative is its all possible senses. However, our ESSK implementation considers the alternative of each word as its disambiguated sense. We use a dictionary based Word Sense Disambiguation (WSD) System assuming one sense per discourse. We use WordNet (Fellbaum, 1998) to find the semantic relations (such as repetition, synonym, hypernym and hyponym, holonym and meronym, and gloss) for all the words in a text. We assign a weight to each semantic relation and used all of them. Our WSD technique is decomposed into two steps: (1) building a representation of all possible senses of the words and (2) disambiguating the words based on the highest score. To be specific, each candidate word from the context is expanded to all of its senses. A disambiguation graph is constructed as the intermediate representation where the nodes denote word instances with their WordNet senses and the weighted edges (connecting the senses of two different words) represent semantic relations. This graph is exploited to perform the WSD. We sum the weights of all edges leaving the nodes under their different senses. The sense with the highest score is considered to be the most probable sense. In case of a tie between two or more senses, we select the sense that comes first in WordNet, since WordNet orders the senses of a word by decreasing order of their frequency.

ESSK is used to measure the similarity between all possible subsequences of the question words/senses and topic words/senses. We calculate the similarity score $\mathrm{Sim}(T_i, Q_j)$ using ESSK where T_i denotes a topic/sub-topic word sequence and Q_j stands for a generated question. Formally, ESSK is defined as follows[7]:

$$K_{essk}(T,Q) = \sum_{m=1}^{d} \sum_{t_i \in T} \sum_{q_j \in Q} K_m(t_i, q_j)$$

$$K_m(t_i, q_j) = \begin{cases} val(t_i, q_j) & \text{if } m = 1 \\ K'_{m-1}(t_i, q_j) \cdot val(t_i, q_j) & \end{cases}$$

Here, $K'_m(t_i, q_j)$ is defined below. t_i and q_j are nodes of T and Q, respectively. The function $val(t, q)$ returns the number of attributes common (i.e. the number of common words/senses) to the given nodes t and q.

$$K'_m(t_i, q_j) = \begin{cases} 0 & \text{if } j = 1 \\ \lambda K'_m(t_i, q_{j-1}) + K''_m(t_i, q_{j-1}) & \end{cases}$$

Here λ is the decay parameter for the number of skipped words. $K''_m(t_i, q_j)$ is defined as:

$$K''_m(t_i, q_j) = \begin{cases} 0 & \text{if } i = 1 \\ \lambda K''_m(t_{i-1}, q_j) + K_m(t_{i-1}, q_j) & \end{cases}$$

Finally, the similarity measure is defined after normalization as below:

$$sim_{essk}(T,Q) = \frac{K_{essk}(T,Q)}{\sqrt{K_{essk}(T,T) K_{essk}(Q,Q)}}$$

[7]The formulae denotes a dynamic programming technique to compute the ESSK similarity score where d is the vector space dimension i.e. the number of all possible subsequences of up to length d. More information about these formulae can be obtained from Hirao et al. (2003, 2004)

3.4 Judging Syntactic Correctness

The generated questions might be syntactically incorrect due to the process of automatic question generation. It is time consuming and a lot of human intervention is necessary to check for the syntactically incorrect questions manually. We strongly believe that a question should have a similar syntactic structure to a sentence from which it is generated. For example, the sentence "Apple's first logo is designed by Jobs and Wayne.", and the generated question "What is designed by Jobs and Wayne?" are syntactically similar. Hence, to judge the syntactic correctness of each generated question automatically, we apply the tree kernel functions and re-implement the syntactic tree kernel model for computing the syntactic similarity of each question with the associated content information. We first parse the sentences and the questions into syntactic trees using the Charniak parser[8] (Charniak, 1999). Then we calculate the similarity between the two corresponding trees using the *tree kernel* method (Collins and Duffy, 2001). We convert each parenthetic representation generated by the Charniak parser into its corresponding tree and give the trees as input to the tree kernel functions for measuring the syntactic similarity.

Each tree T is represented by an m dimensional vector $v(T) = \big(v_1(T), v_2(T), \cdots v_m(T)\big)$, where the i-th element $v_i(T)$ is the number of occurrences of the i-th tree fragment in tree T. The tree fragments of a tree are all of its sub-trees which include at least one production with the restriction that no production rules can be broken into incomplete parts. Figure 1 shows an example tree and a portion of its subtrees.

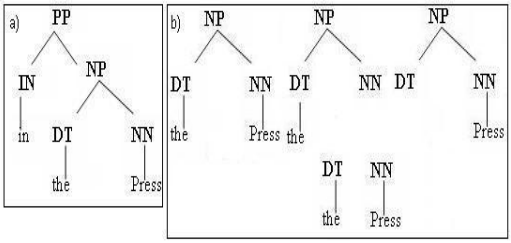


Figure 1: (a) An example tree (b) The sub-trees of the NP covering "the press".

Implicitly we enumerate all the possible tree fragments $1, 2, \cdots, m$. These fragments are the axis of this m-dimensional space. Note that this could be done only implicitly, since the number m is extremely large. Because of this, Collins and Duffy Collins and Duffy (2001) defined the tree kernel algorithm whose computational complexity does not depend on m. The tree kernel of two syntactic trees T_1 and T_2 is actually the inner product of $v(T_1)$ and $v(T_2)$:

$$TK(T_1, T_2) = v(T_1).v(T_2) \tag{2}$$

We define the indicator function $I_i(n)$ to be 1 if the sub-tree i is seen rooted at node n and 0 otherwise. It follows:

$$v_i(T_1) = \sum_{n_1 \in N_1} I_i(n_1)$$

[8] Available at ftp://ftp.cs.brown.edu/pub/nlparser/

$$v_i(T_2) = \sum_{n_2 \in N_2} I_i(n_2)$$

where, N_1 and N_2 are the set of nodes in T_1 and T_2 respectively. So, we can derive:

$$\begin{aligned} TK(T_1, T_2) &= v(T_1).v(T_2) \\ &= \sum_i v_i(T_1) v_i(T_2) \\ &= \sum_{n_1 \in N_1} \sum_{n_2 \in N_2} \sum_i I_i(n_1) I_i(n_2) \\ &= \sum_{n_1 \in N_1} \sum_{n_2 \in N_2} C(n_1, n_2) \end{aligned} \tag{3}$$

where, we define $C(n_1, n_2) = \sum_i I_i(n_1) I_i(n_2)$. Next, we note that $C(n_1, n_2)$ can be computed in polynomial time, due to the following recursive definition:

1. If the productions at n_1 and n_2 are different then $C(n_1, n_2) = 0$
2. If the productions at n_1 and n_2 are the same, and n_1 and n_2 are pre-terminals, then $C(n_1, n_2) = 1$
3. Else if the productions at n_1 and n_2 are not pre-terminals,

$$C(n_1, n_2) = \prod_{j=1}^{nc(n_1)} (1 + C(ch(n_1, j), ch(n_2, j))) \tag{4}$$

where, $nc(n_1)$ is the number of children of n_1 in the tree; because the productions at n_1 and n_2 are the same, we have $nc(n_1) = nc(n_2)$. The i-th child-node of n_1 is $ch(n_1, i)$.

Note that, the tree kernel (TK) function computes the number of common subtrees between two trees. Such subtrees are subject to the constraint that their nodes are taken with all or none of the children they have in the original tree. The *TK* (tree kernel) function gives the similarity score between each sentence in the given body of texts and the generated question based on the syntactic structure. Each sentence[9] contributes a score to the questions and then the questions are ranked by considering the average of similarity scores.

4 Experiments

4.1 System Description

We consider the task of automatically generating questions from topics where each topic is associated with a body of texts having a useful description about the topic. The proposed QG system ranks the questions by combining the topic relevance scores and the syntactic similarity scores of Section 3.3 and Section 3.4 using the formula as follows:

$$w * ESSK_{score} + (1 - w) * SYN_{score} \tag{5}$$

Here w is the importance parameter which holds the value in $[0, 1]$. We kept $w = 0.5$ to give equal importance[10] to topic relevance and syntactic correctness.

[9] We consider that a question is syntactically fluent as well as relevant to the topic if it has similar syntactic sub-trees as those of the most sentences in the body of texts.

[10] A syntactically incorrect question is not useful even if it is relevant to the topic. This motivated us to give equal

4.2 Corpus

To run our experiments, we use the dataset provided in the Question Generation Shared Task and Evaluation Challenge[11] (QGSTEC, 2010) for the task of question generation from paragraphs. This dataset consists of 60 paragraphs about 60 topics that were originally collected from several Wikipedia, OpenLearn, and Yahoo!Answers articles. The paragraphs contain around $5-7$ sentences for a total of $100-200$ tokens (including punctuation). This dataset includes a diversity of topics of general interest. We consider these topics and treat the paragraphs as their associated useful content information in order to generate a set of questions using our proposed QG approach. We use 10 topics and their associated paragraphs as the development data[12]. A total of 2186 questions are generated from the remaining 50 topics (test data) to be ranked.

4.3 Evaluation Setup

4.3.1 Methodology

We use a methodology derived from Boyer and Piwek (2010); Heilman and Smith (2010b) to evaluate the performance of our QG systems. Three native English-speaking university graduate students judge[13] the quality of the top-ranked 20% questions using two criteria: topic relevance and syntactic correctness. For topic relevance, the given score is an integer between 1 (very poor) and 5 (very good) and is guided by the consideration of the following aspects: 1. Semantic correctness (i.e. the question is meaningful and related to the topic), 2. Correctness of question type (i.e. a correct question word is used), and 3. Referential clarity (i.e. it is clearly possible to understand what the question refers to). For syntactic correctness, the assigned score is also an integer between 1 (very poor) and 5 (very good). Whether a question is grammatically correct or not is checked here. For each question, we calculate the average of the judges' scores.

4.3.2 Systems for Comparison

We report the performance of the following systems in order to do a meaningful comparison with our proposed QG system:

(1) Baseline1: This is our QG system without any question-ranking method applied to it. Here, we randomly select 20% questions and rate them.

(2) Baseline2: For our second baseline, we build a QG system using an alternative topic modeling approach. Here we use a topic signature model (instead of using LDA as discussed in Section 3.3.1) (Lin and Hovy, 2000) to identify the important sub-topics from the sentences present in the body of texts. The sub-topics are the important words in the context which are closely related to the topic and have significantly greater probability of occurring in the given text compared to that in a large background corpus. We use a topic signature computation tool[14] for this purpose. The background corpus that is used in this tool contains 5000 documents from the English GigaWord Corpus. For example, from the given body of texts of the topic *Apple Inc.*

importance to topic relevance and syntactic correctness. The parameter w can be tuned to investigate its impact on the system performance.

[11] http://www.questiongeneration.org/mediawiki

[12] We use this data to build necessary general purpose rules for our QG model.

[13] The inter-annotator agreement of Fleiss' $\kappa = 0.41, 0.45, 0.62$, and 0.33 are computed for the three judges for the results in Table 3 to Table 6, indicating moderate (for the first two tables), substantial and fair agreement (Landis and Koch, 1977) between the raters, respectively.

[14] Available at http://www.cis.upenn.edu/ lannie/topicS.html

Logos, we get these sub-topics: *jobs, logo, themes, rainbow, monochromatic*. Then we use the same steps of Section 3.3.2 and Section 3.4, and use equation 5 to combine the scores. We evaluate the top-ranked 20% questions and show the results.

(3) State-of-the-art: We choose a publicly available state-of-the-art QG system[15] to generate questions from the sentences in the body of texts. This system was shown to achieve good performance in generating fact-based questions about the content of a given article (Heilman and Smith, 2010b). Their method ranks the questions automatically using a logistic regression model. Given a paragraph as input, this system processes each sentence and generates a set of ranked questions for the entire paragraph. We evaluate the top-ranked 20% questions[16] and report the results.

4.3.3 Results and Discussion

Table 3 shows the average topic relevance and syntactic correctness scores for all the systems. From these results we can see that the *proposed QG* system improves the topic relevance and syntactic correctness scores over the *Baseline1* system by 61.86%, and 34.98%, respectively, and improves the topic relevance and syntactic correctness scores over the *Baseline2* system by 7.40%, and 7.57%, respectively. On the other hand, the *proposed QG* system improves the topic relevance and syntactic correctness scores over the *state-of-the-art* system by 3.88%, and 2.89%, respectively. From these results, we can clearly observe the effectiveness of our proposed QG system. The improvements in the results are statistically significant[17] ($p < 0.05$).

The main goal of this work was to generate as many questions as possible related to the topic. For this reason, we considered generating the basic questions. These questions were also useful to provide variety in the question space. We generated these questions using the NE information. As the performance of the NE-taggers were not that great, we had a few of these questions generated. In most cases, these questions were outranked by other important questions that included a combination of topics and sub-topics to show higher topic relevance score measured by ESSK. Therefore, they do not have a considerable impact on the evaluation statistics. We claim that the overall performance of our systems could be further improved if the accuracy of the NE-tagger and the semantic role labeler could be increased.

Systems	Topic Relevance	Syntactic Correctness
Baseline1 (No Ranking)	2.15	2.63
Baseline2 (Topic Signature)	3.24	3.30
State-of-the-art (Heilman and Smith, 2010b)	3.35	3.45
Proposed QG System	3.48	3.55

Table 3: Topic relevance and syntactic correctness scores

Acceptability Test In another evaluation setting, the three annotators judge the questions for their overall acceptability as a good question. If a question shows no deficiency in terms of the criteria considered for topic relevance and syntactic correctness, it is termed as *acceptable*. We evaluate the top 15% and top 30% questions separately for each QG system and report

[15] Available at http://www.ark.cs.cmu.edu/mheilman/questions/

[16] We ignore the yes-no questions for our task.

[17] We tested statistical significance using Student's t-test.

the results indicating the percentage of questions rated as acceptable in Table 4. The results indicate that the percentage of the questions rated acceptable is reduced when we evaluate more number of questions which proves the effectiveness of our QG system.

Systems	Top 15%	Top 30%
Baseline1 (No Ranking)	35.2	32.6
Baseline2 (Topic Signature)	45.9	33.8
State-of-the-art (Heilman and Smith, 2010b)	44.7	38.5
Proposed QG System	46.5	40.6

Table 4: Acceptability of the questions (in %)

Systems	Topic Relevance	Syntactic Correctness
Baseline1 (No Ranking)	3.20	3.54
Baseline2 (Topic Signature)	3.80	3.92
State-of-the-art (Heilman and Smith, 2010b)	4.01	4.15
Proposed QG System	4.12	4.25

Table 5: Topic relevance and syntactic correctness scores (narrowed focus)

Systems	Top 15%	Top 30%
Baseline1 (No Ranking)	41.3	37.1
Baseline2 (Topic Signature)	53.5	43.6
State-of-the-art (Heilman and Smith, 2010b)	57.5	43.2
Proposed QG System	58.4	44.5

Table 6: Acceptability of the questions in % (narrowed focus)

Narrowing Down the Focus We run further experiments by narrowing down the topic focus. We consider only the topics about persons (biographical focus). We choose 10 persons as our topics from the list of the 20th century's 100 most influential people, published in Time magazine in 1999 and obtained the paragraphs containing their biographical information from Wikipedia articles[18]. We generate a total of 390 questions from the considered 10 topics and rank them using different ranking schemes as discussed before. We evaluate the top 20% questions using the similar evaluation methodologies and report the results in Table 5. Again, we evaluate the top 15% and top 30% questions separately for each QG system and report the results indicating the percentage of questions rated as acceptable in Table 6. From these tables, we can clearly see the improvements in all the scores for all the QG approaches. This is reasonable because the accuracy of the NE tagger and the semantic role labeler is increased for the biographical data. These results further demonstrate that the proposed system is significantly better (at $p < 0.05$) than the other considered systems. We plan to make our created resources available to other researchers.

[18] http://en.wikipedia.org/wiki/Time_100

Systems	Top-ranked questions
Baseline2	Who presented Jobs with several different monochromatic themes for the bitten logo? What were conceived to make the logo more accessible? Who liked the logo?
State-of-the-art	Whose first logo depicts Sir Isaac Newton sitting under an apple tree? What depicts Sir Isaac Newton sitting under an apple tree? What did Janoff present Jobs with?
Proposed QG System	Who designed Apple's first logo? What was replaced by Rob Janoff 's "rainbow Apple"? What were conceived to make the logo more accessible?

Table 7: System output

4.3.4 An Input-Output Example

An input to our systems is for instance, the topic *"Apple Inc. Logos"* with the associated content information (body of texts): *"Apple's first logo, designed by Jobs and Wayne, depicts Sir Isaac Newton sitting under an apple tree. Almost immediately, though, this was replaced by Rob Janoff's "rainbow Apple", the now-familiar rainbow-colored silhouette of an apple with a bite taken out of it. Janoff presented Jobs with several different monochromatic themes for the "bitten" logo, and Jobs immediately took a liking to it. While Jobs liked the logo, he insisted it be in color to humanize the company. The Apple logo was designed with a bite so that it would be recognized as an apple rather than a cherry. The colored stripes were conceived to make the logo more accessible, and to represent the fact the monitor could reproduce images in color. In 1998, with the roll-out of the new iMac, Apple discontinued the rainbow theme and began to use monochromatic themes, nearly identical in shape to its previous rainbow incarnation."* The output of our systems is the ranked lists of questions. We show an example output in Table 7.

Conclusion and Future Work

In this paper, we have considered the task of automatically generating questions from topics where each topic is associated with a body of texts containing useful information. We have exploited the named entity and semantic role labeling information to accomplish the task. A key aspect of our approach is the use of LDA to automatically discover the hidden sub-topics from the sentences. We have proposed a method to rank the generated questions by considering: 1) sub-topical similarity determined using ESSK algorithm in combination with word sense disambiguation, and 2) syntactic similarity determined using the syntactic tree kernel based method. We have compared the proposed QG system with two baseline systems and one state-of-the-art system. The evaluation results have shown that the proposed QG system significantly outperforms all other considered systems as our system generated top-ranked questions are found to be better in topic-relevance and syntactic correctness than those of the other systems. We have conducted another experiment by narrowing down the topic focus. In this experiment, we have considered *persons* as topics. Our experiments have demonstrated the effectiveness of the proposed topic to question generation approach. We hope to carry on this ideas and develop further mechanisms to question generation based on the dependency features of the answers and answer finding (Li and Roth, 2006; Pinchak and Lin, 2006).

Acknowledgments

The research reported in this paper was supported by the Natural Sciences and Engineering Research Council (NSERC) of Canada – discovery grant and the University of Lethbridge.

References

Agarwal, M., Shah, R., and Mannem, P. (2011). Automatic Question Generation Using Discourse Cues. In *Proceedings of the 6th Workshop on Innovative Use of NLP for Building Educational Applications*, pages 1–9. ACL.

Ali, H., Chali, Y., and Hasan, S. A. (2010). Automation of Question Generation from Sentences. In *Proceedings of QG2010: The Third Workshop on Question Generation*, Pittsburgh, USA.

Andrenucci, A. and Sneiders, E. (2005). Automated Question Answering: Review of the Main Approaches. In *Proceedings of the 3rd International Conference on Information Technology and Applications (ICITA'05)*, Sydney, Australia.

Blei, D. M., Ng, A. Y., and Jordan, M. I. (2003). Latent Dirichlet Allocation. *Journal of Machine Learning Research*, 3:993–1022.

Boyer, K. E. and Piwek, P. (2010). Proceedings of QG2010: The Third Workshop on Question Generation. Pittsburgh: questiongeneration.org.

Brown, J. C., Frishkoff, G. A., and Eskenazi, M. (2005). Automatic Question Generation for Vocabulary Assessment. In *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, Vancouver, British Columbia, Canada.

Cancedda, N., Gaussier, E., Goutte, C., and Renders, J. M. (2003). Word Sequence Kernels. *Journal of Machine Learning Research*, 3:1059–1082.

Celikyilmaz, A., Hakkani-Tur, D., and Tur, G. (2010). LDA based Similarity Modeling for Question Answering. In *Proceedings of the NAACL HLT 2010 Workshop on Semantic Search*, SS '10, pages 1–9. ACL.

Chali, Y., Hasan, S. A., and Joty, S. R. (2009a). Do Automatic Annotation Techniques Have Any Impact on Supervised Complex Question Answering? In *Proceedings of the Joint conference of the 47th Annual Meeting of the Association for Computational Linguistics (ACL-IJCNLP 2009)*, pages 329–332, Suntec, Singapore.

Chali, Y., Hasan, S. A., and Joty, S. R. (2011). Improving Graph-based Random Walks for Complex Question Answering using Syntactic, Shallow Semantic and Extended String Subsequence Kernels. *Information Processing & Management*, 47(6):843–855.

Chali, Y., Joty, S. R., and Hasan, S. A. (2009b). Complex Question Answering: Unsupervised Learning Approaches and Experiments. *Journal of Artificial Intelligence Research*, 35:1–47.

Charniak, E. (1999). A Maximum-Entropy-Inspired Parser. In *Technical Report CS-99-12*, Brown University, Computer Science Department.

Chen, W., Aist, G., and Mostow, J. (2009). Generating Questions Automatically from Informational Text. In *Proceedings of the 2nd Workshop on Question Generation (AIED 2009)*, pages 17–24.

Collins, M. and Duffy, N. (2001). Convolution Kernels for Natural Language. In *Proceedings of Neural Information Processing Systems*, pages 625–632, Vancouver, Canada.

Echihabi, A. and Marcu, D. (2003). A Noisy-channel Approach to Question Answering. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics - Volume 1*, pages 16–23. ACL.

Fellbaum, C. (1998). WordNet - An Electronic Lexical Database. Cambridge, MA. MIT Press.

Geman, S. and Geman, D. (1984). Stochastic Relaxation, Gibbs Distributions, and the Bayesian Restoration of Images. *IEEE Trans. Pattern Anal. Mach. Intell.*, 6:721–741.

Graesser, A. C., VanLehn, K., Rose, C. P., Jordan, P. W., and Harter, D. (2001). Intelligent Tutoring Systems with Conversational Dialogue. *AI Magazine*, 22(4):39–52.

Griffiths, T. L. and Steyvers, M. (2002). Prediction and Semantic Association. In *NIPS'02*, pages 11–18.

Hacioglu, K., Pradhan, S., Ward, W., Martin, J. H., and Jurafsky, D. (2003). Shallow Semantic Parsing Using Support Vector Machines. In *Technical Report TR-CSLR-2003-03*, University of Colorado.

Heilman, M. and Smith, N. A. (2010a). Extracting Simplified Statements for Factual Question Generation. In *Proceedings of the Third Workshop on Question Generation*.

Heilman, M. and Smith, N. A. (2010b). Good Question! Statistical Ranking for Question Generation. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 609–617.

Hickl, A., Lehmann, J., Williams, J., and Harabagiu, A. (2005). Experiments with Interactive Question-Answering. In *In Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 60–69.

Hirao, T., Suzuki, J., Isozaki, H., and Maeda, E. (2003). NTT's Multiple Document Summarization System for DUC2003. In *Proceedings of the Document Understanding Conference*.

Hirao, T., Suzuki, J., Isozaki, H., and Maeda, E. (2004). Dependency-based Sentence Alignment for Multiple Document Summarization. In *Proceedings of COLING 2004*, pages 446–452, Geneva, Switzerland. COLING.

Ignatova, K., Bernhard, D., and Gurevych, I. (2008). Generating High Quality Questions from Low Quality Questions. In *Proceedings of the Workshop on the Question Generation Shared Task and Evaluation Challenge*, Arlington, VA. NSF.

Kingsbury, P. and Palmer, M. (2002). From Treebank to PropBank. In *Proceedings of the International Conference on Language Resources and Evaluation*, Las Palmas, Spain.

Kotov, A. and Zhai, C. (2010). Towards Natural Question Guided Search. In *Proceedings of the 19th international conference on World wide web*, WWW '10, pages 541–550. ACM.

Landis, J. R. and Koch, G. G. (1977). The Measurement of Observer Agreement for Categorical Data. *Biometrics*, 33(1):159–174.

Lauer, T. W., Peacock, E., and Graesser, A. C. (1992). Questions and Information Systems.

Li, X. and Roth, D. (2006). Learning Question Classifiers: The Role of Semantic Information. *Journal of Natural Language Engineering*, 12(3):229–249.

Lin, C. Y. (2008). Automatic Question Generation from Queries. In *Proceedings of the Workshop on the Question Generation Shared Task and Evaluation Challenge*, Arlington, VA. NSF.

Lin, C. Y. and Hovy, E. H. (2000). The Automated Acquisition of Topic Signatures for Text Summarization. In *Proceedings of the 18th conference on Computational linguistics*, pages 495–501.

Lodhi, H., Saunders, C., Shawe-Taylor, J., Cristianini, N., and Watkins, C. (2002). Text Classification using String Kernels. *Journal of Machine Learning Research*, 2:419–444.

Mannem, P., Prasad, R., and Joshi., A. (2010). Question Generation from Paragraphs at Upenn. In *Proceedings of the Third Workshop on Question Generation*.

McCallum, A. K. (2002). MALLET: A Machine Learning for Language Toolkit.

McConnell, C. C., Mannem, P., Prasad, R., and Joshi, A. (2011). A New Approach to Ranking Over-Generated Questions. In *Proceedings of the AAAI Fall Symposium on Question Generation*.

McGough, J., Mortensen, J., Johnson, J., and Fadali, S. (2001). A Web-based Testing System with Dynamic Question Generation. In *ASEE/IEEE Frontiers in Education Conference*.

Misra, H., Cappé, O., and Yvon, F. (2008). Using LDA to Detect Semantically Incoherent Documents. In *Proceedings of the Twelfth Conference on Computational Natural Language Learning*, CoNLL '08, pages 41–48. ACL.

Mitkov, R. and Ha, L. A. (2003). Computer-aided Generation of Multiple-Choice Tests. In *Proceedings of the HLT-NAACL 03 workshop on Building educational applications using natural language processing - Volume 2*, pages 17–22.

Moschitti, A. and Basili, R. (2006). A Tree Kernel Approach to Question and Answer Classification in Question Answering Systems. In *Proceedings of the 5th International Conference on Language Resources and Evaluation*, Genoa, Italy.

Moschitti, A., Quarteroni, S., Basili, R., and Manandhar, S. (2007). Exploiting Syntactic and Shallow Semantic Kernels for Question/Answer Classificaion. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 776–783, Prague, Czech Republic. ACL.

Olney, A. M., Graesser, A. C., and Person, N. K. (2012). Question Generation from Concept Maps. *Dialogue and Discourse*, 3(2):75–99.

Palmer, M., Gildea, D., and Kingsbury, P. (2005). The Proposition Bank: An Annotated Corpus of Semantic Roles. *Computational Linguistics*, 31(1):71–106.

Pasca, M. and Harabagiu, S. M. (2001). Answer Mining from On-Line Documents. In *Proceedings of the Association for Computational Linguistics 39th Annual Meeting and 10th Conference of the European Chapter Workshop on Open-Domain Question Answering*, pages 38–45, Toulouse, France.

Pinchak, C. and Lin, D. (2006). A Probabilistic Answer Type Model. In *Proceedings of the 11th Conference of the European Chapter of the Association for Computational Linguistics*, pages 393–400.

Ratinov, L. and Roth, D. (2009). Design Challenges and Misconceptions in Named Entity Recognition. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning*, pages 147–155. ACL.

Rus, V., Cai, Z., and Graesser, A. C. (2007). Experiments on Generating Questions About Facts. In *Proceedings of the 8th International Conference on Computational Linguistics and Intelligent Text Processing*, pages 444–455. Springer-Verlag.

Rus, V. and Graesser, A. C. (2009). The Question Generation Shared Task and Evaluation Challenge. In *Workshop on the Question Generation Shared Task and Evaluation Challenge, Final Report*, The University of Memphis. National Science Foundation.

Vanderwende, L. (2008). The Importance of Being Important: Question Generation. In *Proceedings of the Workshop on the Question Generation Shared Task and Evaluation Challenge*, Arlington, VA. NSF.

Wang, W., Tianyong, H., and Wenyin, L. (2008). Automatic Question Generation for Learning Evaluation in Medicine. In *LNCS Volume 4823*.

Wei, X. and Croft, W. B. (2006). LDA-based Document Models for Ad-hoc Retrieval. In *Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '06, pages 178–185. ACM.

Zhang, A. and Lee, W. (2003). Question Classification using Support Vector Machines. In *Proceedings of the Special Interest Group on Information Retrieval*, pages 26–32, Toronto, Canada. ACM.

Zheng, Z., Si, X., Chang, E. Y., and Zhu, X. (2011). K2Q: Generating Natural Language Questions from Keywords with User Refinements. In *Proceedings of the 5th International Joint Conference on Natural Language Processing*, pages 947–955.

Adjective Deletion for Linguistic Steganography and Secret Sharing

Ching – Yun Chang[1] *Stephen Clark*[1]
(1) University of Cambridge, Computer Laboratory, 15 JJ Thomson Avenue, Cambridge, UK
Ching-Yun.Chang@cl.cam.ac.uk, Stephen.Clark@cl.cam.ac.uk

ABSTRACT
This paper describes two methods for checking the acceptability of adjective deletion in noun phrases. The first method uses the Google n-gram corpus to check the fluency of the remaining context after an adjective is removed. The second method trains an SVM model using n-gram counts and other measures to classify deletable and undeletable adjectives in context. Both methods are evaluated against human judgements of sentence naturalness. The application motivating our interest in adjective deletion is data hiding, in particular linguistic steganography. We demonstrate the proposed adjective deletion technique can be integrated into an existing stegosystem, and in addition we propose a novel secret sharing scheme based on adjective deletion.

KEYWORDS: linguistic steganography, adjective deletion.

Proceedings of COLING 2012: Technical Papers, pages 493–510,
COLING 2012, Mumbai, December 2012.

1 Introduction

Linguistic steganography is a form of covert communication in which information is embedded in a seemly innocent cover text so that the presence of the information is imperceptible to an outside observer (human or computer) (Fridrich, 2009). An ideal linguistic stegosystem should fulfil two fundamental requirements: *high imperceptibility* and *high payload capacity*. The former aims at imposing minimum embedding distortion to the cover text so that the resulting stegotext in which a message is camouflaged is inconspicuous. The latter aims at providing sufficient embedding capacity in order to achieve efficient information transmission. There is a trade-off between imperceptibility and payload, since any attempt to embed additional information via changes to the cover text increases the chance of introducing anomalies into the text and thus raising the suspicion of an observer (Chang and Clark, 2010a).

Another cryptographic method is secret sharing. Secret sharing (Blakley, 1979; Shamir, 1979) refers to methods for distributing a secret amongst a group of n people, each of whom is allocated a *share* of the secret. Individual shares are of no use on their own; only when any group of t (for *threshold*) or more shares are combined together can the secret be reconstructed. Such a system is called a (t, n)-threshold scheme. For example, a simple (3,3)-threshold scheme for a secret number s can be achieved by splitting s into three numerical shares s_1, s_2 and s_3 such that $s = s_1 + s_2 + s_3$. Note that there is no way to recover the secret number by only using one or two of the shares; all shares are required for effective recovery.

There are some proposed (t, n)-threshold schemes where $t \neq n$. For example, Shamir's scheme (Shamir, 1979) allows that any t out of n shares may be used to recover the secret. This scheme relies on the idea that it takes t points to define a polynomial of degree t-1 (e.g. it takes two points to define a straight line, three points to define a quadratic, four points to define a cubic curve). The method first randomly creates a polynomial of degree t-1 with the secret number as the first coefficient. Next each of the n people is given a distinct point on the curve. Therefore, any t out of the n people can fit a (t-1)th degree polynomial using their points, where the first coefficient is the secret. For example, any three of the five points (1, 1494), (2, 1942), (3, 2578), (4, 3402) and (5, 4414) can fit the polynomial of degree two $f(x) = 1234 + 166x + 94x^2$ and reveal the secret as 1234.[1] From the above two secret sharing schemes we can see that the share can be in different forms, such as numbers and points, depending on the methods used.

In this paper, we propose a novel (2, 2)-threshold secret sharing method where the shares are presented as two comparable texts, as explained in Section 8. The proposed method exploits the adjective deletion technique to embed secret bitstrings of 0s and 1s in two texts. These two texts can then be combined to reveal the secret bitstring; but neither text by itself can reveal the bitstring. Hence the proposed method is a novel combination of secret sharing and linguistic steganography. In addition, we demonstrate the adjective deletion technique can be integrated into an existing linguistic stegosystem (Chang and Clark, 2010a).

We have identified adjectives as a potentially large source of deletable words, in the sense that adjectives can often be removed without significantly affecting the meaning or naturalness of the resulting text. For example, "he spent only his *own* money" and "he spent only his money" express the same meaning. In the extreme case, there are adjective-noun pairs in which the adjective is somewhat redundant, for example *unfair prejudice*, *horrible crime* and *fragile glass*.

We explore the identification of redundant adjectives in context for the applications of linguistic

[1] http://en.wikipedia.org/wiki/Shamir's_Secret_Sharing.

steganography and secret sharing. In order to generate unsuspicious stegotext and textual shares after adjective deletion, we propose two checking methods using the Google n-gram corpus and an SVM to certify the naturalness of the generated sentences. The methods are evaluated using human judgements of naturalness. Note that the evaluation is based on the sentence-level naturalness rather than the coherence of the whole document. Modeling the document-level coherence of modified text would be useful but is outside the scope of our study. The resulting precision can be seen as an indirect measure of the imperceptibility of the stegosystem since quality deletions are less likely to be seen as suspicious by the observer, whereas the recall can be seen as an indirect measure of the payload since deletable adjectives are detected where possible and therefore as much information as possible is embedded.

There are various practical security issues in the application of linguistic steganography and secret sharing systems that we have chosen to ignore or simplify in order to focus on the underlying NLP technology. For example, we assume the adversary is a human acting passively rather than actively. In other words, we have ignored the possibility of computational steganalysis and steganographic attacks, such as detecting, extracting and destroying the hidden message (Fridrich, 2009).

2 Related Work

2.1 Linguistic Transformations for Steganography

Existing studies have exploited different linguistic transformations for the application of steganography, such as lexical substitution (Chapman and Davida, 1997; Bolshakov, 2004; Taskiran et al., 2006; Topkara et al., 2006c; Chang and Clark, 2010b), phrase paraphrasing (Chang and Clark, 2010a), sentence structure manipulations (Atallah et al., 2001a,b; Liu et al., 2005; Meral et al., 2007; Murphy, 2001; Murphy and Vogel, 2007b; Topkara et al., 2006b) and semantic transformations (Atallah et al., 2002; Vybornova and Macq, 2007). For details of the transformations mentioned above, readers can refer to our previous papers: Chang and Clark (2010a) and Chang and Clark (2010b).

Another group of studies aim to embed information into translated text. Stutsman et al. (2006) use multiple translation systems to provide alternative candidates for a sentence. The secret information is then embedded into the choice of translation. Another recent work proposed by Venugopal et al. (2011) introduces a watermark as a parameter in the machine translation algorithm and probabilistically identifies the watermarked translation. The motivation of watermarking machine translation outputs is to distinguish machine and human generated translations so a machine translation system is unlikely to learn from self-generated data.

These transformations often rely on sophisticated NLP tools and resources. For example, a lexical substitution-based stegosystem may require synonym dictionaries, POS taggers, word sense disambiguation tools and language models; a syntactic transformation-based stegosystem may require syntactic or semantic parsers and language generation tools. However, given the current state-of-the-art, such NLP techniques cannot guarantee the transformation's imperceptibility. Hence it is important to evaluate the security level of a stegosystem.

2.2 Stegosystem Evaluations

A stegosystem can be evaluated from two aspects: the security level and the embedding capacity. The security assessment methods used so far can be classified into two categories: automatic evaluation and human evaluation. Topkara et al. (2006b) and Topkara et al. (2006a) used

machine translation evaluation metrics BLEU and NIST, automatically measuring how close a stego sentence is to the original. Topkara et al. (2006b) admitted that machine translation evaluation metrics are not sufficient for evaluating stegosystems; for example, BLEU relies on word sequences in the stego sentence matching those in the cover sentence and thus is not suitable for evaluating transformations that change the word order significantly.

The other widely adopted evaluation method is based on human judgements. Meral et al. (2007), Kim (2008), Kim (2009) and Meral et al. (2009) asked subjects to edit stegotext for improving intelligibility and style. The fewer edit-hits a transformed text received, the higher the reported security level. Murphy and Vogel (2007b) and Murphy and Vogel (2007a) first asked subjects to rate the acceptability (in terms of plausibility, grammaticality and style) of the stego sentences on a seven-point scale. Then subjects were provided with the originals and asked to judge to what extent meaning was preserved on a seven-point scale. Chang and Clark (2010a) asked subjects to judge whether a paraphrased sentence is grammatical and whether the paraphrasing retains the meaning of the original.

The other aspect of the stegosystem evaluation is to calculate the amount of data capable of being embedded in a stego text, which can be quantified in terms of bits per language unit, for example per word or per sentence. Payload measurements can be theoretical or empirical. The theoretical payload measurement only depends on an encoding method and is independent of the quality of a stego text; whereas the empirical measurement takes the applicability of a linguistic transformation, namely the security of a stego text, into consideration and measures the payload capacity while a certain security level is achieved. Most of the payload rates reported in existing work are based on empirical measurements, with typical payload rates between 0.5 and 1.0 bits per sentence.

Not only the linguistic transformation and the encoding method, but also the choice of cover text can affect the security level and the payload capacity of a stegosystem. For example, if a newspaper article were chosen as the cover text, then any changes could be easily found in practice by comparing the stego text with the original article, which is likely to be readily available. In addition, an anomaly introduced by a linguistic transformation may be more noticeable in a newspaper article than in a blog article. In terms of payload capacity, a synonym substitution-based stegosystem may find more words that can be substituted in a fairy tale than in a medical paper since there are usually many terminologies in a medical paper which cannot be changed or even cannot be found in a standard dictionary. To the best of our knowledge, there is no study on the practical issue of using different types of cover text for the steganography application.

2.3 Sentence Compression

Sentence compression, text simplification and text summarisation usually involve removing unimportant words in a sentence in order to make the text more concise. For example, Knight and Marcu (2002), Cohn and Lapata (2008), Filippova and Strube (2008) and Zhu et al. (2010) have used the word deletion operation in their systems. However, to our knowledge, there is no work looking at redundant adjectives in text in particular. The proposed adjective deletion methods can be applied before and/or after a sentence compression system. Deleting unnecessary adjectives before can help the system focus on other content of a sentence. Deleting unnecessary adjectives after can generate an even more concise sentence.

Sentence	Those awaiting execution spent their *last* days alone .
Supertags before deletion	NP[nb]/N N/N N (S[dcl]\NP)/NP NP[nb]/N *N/N* N NP\NP .
Supertags after deletion	NP[nb]/N N/N N (S[dcl]\NP)/NP NP[nb]/N N N NP\NP .
Sentence	We met in UK *last* time .
Supertags before deletion	NP S[dcl]\NP ((S\NP)\(S\NP))/NP N *((S\NP)\(S\NP))/((S\NP)\(S\NP))* (S\NP)\(S\NP) .
Supertags after deletion	NP S[dcl]\NP ((S\NP)\(S\NP))/NP N/N N .

Table 1: Comparing supertags before and after adjective deletion

3 Deletable Adjective Classification

In order for an adjective deletion to be acceptable according to our method, we use two checks: grammaticality and naturalness checks. In order to prevent an ungrammatical adjective deletion, we use the syntactic filter proposed in Chang and Clark (2010a) to certify the deletion grammaticality. This is only a preliminary grammaticality check and does not guarantee sentence fluency. For generating the modified sentence, we also use Minnen et al. (2001)'s tools for correcting the form of an indefinite. For example, after deleting *alternative*, the phrase "**an** alternative choice" would be modified to "**a** choice". The original and modified sentences are then parsed using a wide-coverage CCG parser (Clark and Curran, 2007). After parsing, each lexical token is associated with a syntactic description, called a lexical category, or supertag. With the significant amount of information included in supertags, comparing two sequences of supertags is similar to comparing two syntax trees. Thus we require a deletion to retain the same sequence of supertags as that of the original sentence in order to ensure grammaticality. Table 1 shows two adjective deletion examples and their supertags,[2] where *last* is the target adjective. The first deletion case passes the grammaticality check since all the supertags remain the same after deleting *last*; while in the second example, both *UK* and *time*'s supertags are changed after the deletion and thus, this deletion fails the check. Note that all the experiment data used in this paper pass the syntax check.

3.1 N-gram Count Method

Inspired by Chang and Clark (2010b), which used the Google n-gram corpus to check the applicability of a synonym in context based on Bergsma et al. (2009), we use a similar method to calculate a score based on the n-gram counts before and after a potential deletion, as demonstrated in Table 2. The Google n-gram corpus[3] is a large publicly available collection of bi-grams to five-grams generated from approximately 1 trillion tokens of online text. Only n-grams appearing more than 40 times are kept in the corpus.

For the example in Table 2 we first extract contextual bi- to five-grams containing the target adjective *alternative* as well as that across the target position with *alternative* removed. The Google n-gram corpus is then consulted to obtain their frequency counts. We sum up all the logarithmic counts[4] for the original and modified cases. The reason for using the logarithm count is that lower-order n-grams usually have much larger counts than higher-order n-grams so taking the logarithm may prevent the sum being dominated by lower-order n-gram counts. Since before the deletion there are more n-grams extracted, we divide the sum by the number

[2]There is a parse error in the first sentence, but it does not affect the supertag comparison.
[3]Available from the LDC as LDC2006T13.
[4]log(0) and division by zero are taken to be zero.

There is always an *alternative* choice in a mental situation.	
N-grams before the deletion (log *freq*)	N-grams after the deletion (log *freq*)
an *alternative* (15.5)	a choice (15.2)
alternative choice (9.8)	always a choice (8.8)
always an *alternative* (7.9)	a choice in (11.3)
an *alternative* choice (9)	is always a choice (8.3)
alternative choice in (6.2)	always a choice in (5.5)
is always an *alternative* (7.4)	a choice in a (7.6)
always an *alternative* choice (0)	There is always a choice (7)
an *alternative* choice in (5.5)	is always a choice in (4.3)
alternative choice in a (0)	always a choice in a (0)
There is always an *alternative* (6)	a choice in a mental (0)
is always an *alternative* choice (0)	
always an *alternative* choice in (0)	
an *alternative* choice in a (0)	
alternative choice in a mental (0)	
$Count_{average}^{Before} = 4.8$	$Count_{average}^{After} = 6.8$
$Score = \frac{Count_{average}^{After}}{Count_{average}^{Before}} = 1.4$	

Table 2: An example of the Google n-gram count method

of extracted n-grams and call the derived average value the $Count_{average}$. Finally, we use a *Score* function which is equal to $\frac{Count_{average}^{After}}{Count_{average}^{Before}}$ to measure how much the $Count_{average}^{Before}$ changes after deleting the target word *alternative*. In this example the *Score* for deleting *alternative* in this context is equal to 1.4 and will be determined as acceptable by a threshold with value below 1.4.

3.2 SVM Method

Since some n-grams may be more informative than others when deciding whether an adjective can be deleted, we combine the n-gram counts and other measures described in Section 4 to train a classifier. We use the LIBSVM (Chang and Lin, 2011) implementation of support vector machines (SVMs) for classification. As suggested by Hsu et al. (2010), we scale feature values to the range [-1, +1], and use the default radial basis function (RBF) kernel. The two parameters of the RBF kernel, C and γ, are determined by using the model selection tool grid.py provided from LIBSVM. After the best (C, γ) is found using the training data, the whole training set is used again to train the final classifier. In order to observe the trade-off between precision and recall, we use the probability estimate feature in LIBSVM and train the SVM model to output probabilities so users can decide the security level by varying a probability threshold.

4 Features for the SVM

4.1 N-gram Counts

The first set of features consists of logarithmic contextual bi- to five-gram counts. Before the deletion, there are 14 contextual n-grams; after the deletion, there are 10 contextual n-grams as shown in Table 2. If a contextual window is not available, for example if a window spans beyond

the current sentence, the n-gram count is set to zero. For each contextual window we provide an additional boolean feature to indicate whether a window is available. The second set of features consists of 5 score values. The first score is the *Score* function described in Section 3.1. The second to the fifth scores are the scores calculated by only considering a specific window size n, where $n = 2$ to 5, using the same method as for the *Score* function. Again, each score is provided with an additional boolean feature to indicate whether the $Count^{Before}_{average}$ is equal to zero. There are a total of 58 features contributed from the n-gram counts.

4.2 Lexical Association Measures

In addition to n-gram features, we exploit some standard lexical association measures to determine the degree of association between an adjective and a noun. Pointwise Mutual Information (PMI) (Church and Hanks, 1990) is roughly a measure of how much one word tells us about the other. In order to calculate PMI, we need the joint frequency of the noun-adjective pair, the frequency of the noun modified by any adjective and the frequency of the adjective modifying any noun.

We collect (adjective, noun) pairs and their frequency counts from grammatical relations (GRs). The GRs we use are derived by parsing a Wikipedia dump (dated October 2007) with Clark and Curran (2007)'s CCG parser. We first consider GRs having the pattern (ncmod _ *noun adjective*) and extract the (*adjective*, *noun*) pair. Next we extract pairs that match patterns (xcomp _ be *adjective*) and (ncsubj be *noun* _) in a sentence. For instance, the GRs of the sentence "The car is red" are (det car_1 the_0) (xcomp _ be_2 red_3) and (ncsubj be_2 car_1 _), and since *car* and *red* match the two patterns, (*red*, *car*) is seen as an eligible pair for our database. A total of 63,896,006 adjective-noun pairs are extracted form the parsed Wikipedia corpus which includes 832,320 noun types and 792,914 adjective types.

We also use the log likelihood ratio (LLR), an alternative to PMI, which is reported to handle rare events better (Dunning, 1993). Again, the contingency table for computing LLR can be derived from the parsed Wikipedia corpus described above. In the study of collocation extraction, both high PMI and LLR values are treated as evidence that the collocation components occur together more often than by chance. In this paper, we use PMI and LLR as features in the SVM.

4.3 Noun and Adjective Entropy

Suppose we observe a noun N_1 as being modified by adjective J_1 five times, J_2 twice and J_3 three times. The modifier entropy of N_1 is $H(N_1) = -((0.5\,log\,0.5) + (0.2\,log\,0.2) + (0.3\,log\,0.3)) = 1.5$. Now suppose there is a noun N_2 modified by J_4 nine times and J_5 once. The modifier entropy of N_2 is $H(N_2) = -((0.9\,log\,0.9) + (0.1\,log\,0.1)) = 0.5$. Thus we can conclude that the modifier of N_1 is more unpredictable than that of N_2. Similarly, we calculate an adjective's argument entropy based on the entropy of the noun given a fixed adjective.

We also observe the modification frequency of a noun using the parsed Wikipedia corpus. From the corpus, we obtain the frequency of a noun being modified by any adjective (mod_{adj}), the frequency of a noun being modified by something other than an adjective (mod_{other}), and the frequency of a noun not being modified at all (mod_{non}). The modification entropy of a noun is defined as: $M(N) = -(p(mod_{adj})\,log\,p(mod_{adj}) + p(mod_{non})\,log\,p(mod_{non}))$. Note that $p(mod_{other})$ is not included in the definition of $M(N)$ since we want to focus on the adjectival modification of a noun. Modifier entropy, argument entropy, modification entropy plus the modification probabilities $p(mod_{adj})$, $p(mod_{other})$ and $p(mod_{non})$ are used as SVM features.

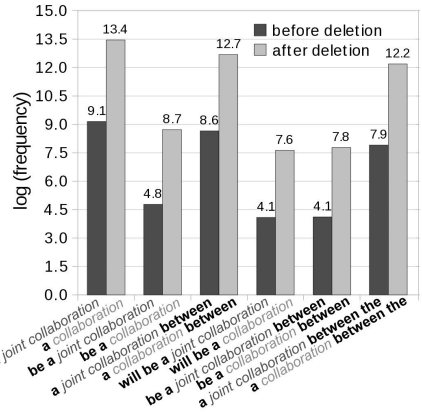


Figure 1: N-gram count distributions before and after deleting *joint*

4.4 Contextual α-Skew Divergence

We assume that if an adjective in a noun phrase is deletable, the noun should have a similar n-gram distribution to the original adjective-noun phrase across the various n-gram counts. Figure 1 shows the logarithmic n-gram counts of *joint collaboration* and *collaboration* being in the same context of the sentence "The task force will be a *joint collaboration* between the cities of Sterling Heights and Warren." In this example sentence, *joint* is determined as deletable. We can see that the counts have similar distributions before and after the deletion.

We use α-skew divergence (Lee, 1999) to calculate the n-gram distributional similarity between the original and the modified sentences. The α-skew divergence is a non-symmetric measure of the difference between two probability distributions P and Q. In our application, P is a probability vector containing normalised logarithmic counts derived from the contextual n-grams before removing the adjective, and Q is a probability vector obtained after deleting the adjective. The α-skew divergence measure is defined as:

$$S_\alpha(Q,P) = D(P\|\alpha{\cdot}Q + (1-\alpha){\cdot}P),$$

where $0 \leq \alpha \leq 1$ and D is the Kullback-Leibler divergence $D(P\|Q) = \sum_v P(v) log \frac{P(v)}{Q(v)}$. The α parameter is for avoiding the problem of zero probabilities, and in our system we use α=0.99. Under our assumption, a deletable adjective would have a smaller effect on the n-gram count distribution after deletion than an undeletable adjective and, therefore, a deletable adjective would have a smaller divergence value.

5 Data

5.1 Pilot Study Data

We first created a small dataset for a preliminary study. In order to experiment with redundant adjectives, we collected 90 sentences from the Internet, each of which contained an adjective-

noun pleonasm.[5] A pleonasm consists of two concepts (usually two words) that are mutually redundant: examples are *free gift*, *cold ice* or *final end*. In other words, pleonasms contain unnecessary words, and those words can be removed without affecting the meaning of the text.

Apart from positive data (deletable adjectives), we also need some negative data (undeletable adjectives) to test whether the n-gram count method and the SVM model have the ability to filter out bad deletions. We define an adjective as undeletable if the removal of an adjective in a noun phrase significantly affects the naturalness of the resulting sentence. The second author of this paper manually selected 76 undeletable cases from the British National Corpus (BNC) as the negative data.

Adjectives in pleonasms can be seen as extreme redundancies in text, and removing those redundancies would not reduce the level of security in terms of steganography. However, pleonasms are not general enough so might not be found frequently in text, which diminishes the amount of secret information which can be embedded in the text. Thus we collect more positive and negative data which are more frequent in text for training, developing and testing the n-gram count method and the SVM model. This additional set serves as our main data source (described in Section 5.2), with the pleonasm set serving as a useful pilot study.

5.2 Human Annotated Data

In order to have a labelled dataset for training and testing a classifier, we asked 30 native English speakers to judge whether the removal of an adjective in a noun phrase significantly affects the naturalness of the resulting sentence. The guideline is the same as that used for the pilot study data. Note that we only care about the naturalness of the resulting sentence rather than the meaning retention of the original sentence. In other words, the evaluation is based on the sentence-level naturalness rather than the coherence of the whole document. Table 3 shows the six examples that were used as part of the annotator instructions.

The sentences for creating the data were randomly selected from section A of the British National Corpus (BNC) with the constraint that each passed the syntax check as described in Section 3. We collected 1200 sentences, each of which contains one marked adjective to be annotated. In order to measure the inter-annotator agreement, 300 of the 1200 sentences were assessed by 3 different judges; the others were labelled only once. We calculated the inter-annotator agreement using Fleiss' kappa (Fleiss et al., 2003) scored between 0 and 1. Fleiss' kappa works for any fixed number of annotators and allows different items rated by different individuals. For the 300 multi-judged instances, the Fleiss' kappa is 0.49, which can be interpreted as *moderate agreement* according to Landis and Koch (1977).

The 300 multi-judged instances were labelled using the majority rule and were treated as the test set; the other 900 instances were randomly split into a 700-instance training set and a 200-instance development set. The ratio of the number of deletable adjectives to the number of undeletable adjectives is around 2:1 for all the datasets.

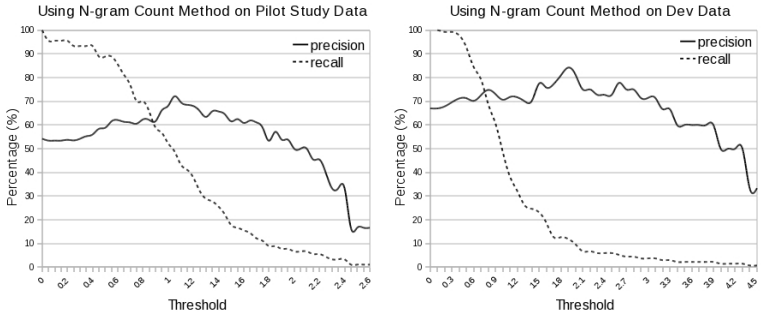
6 Experiments and Results

The performance on this adjective deletion task is measured in precision and recall on the positive deletable cases. From a steganography aspect, accuracy is not useful, while the trade-off between precision and recall is more relevant. A precision baseline is obtained by always saying

[5] A collection of pleonasms can be found at http://www.pleonasms.com.

Judgement	Example sentence
Deletable	He was putting on his *heavy* overcoat, asked again casually if he could have a look at the glass.
Deletable	We are seeking to find out what *local* people want, because they must own the work themselves.
Deletable	We are just at the beginning of the *worldwide* epidemic and the situation is still very unstable.
Undeletable	He asserted that a modern artist should be in tune with his times, careful to avoid *hackneyed* subjects.
Undeletable	With various groups suggesting police complicity in township violence, many blacks will find *little* security in a larger police force.
Undeletable	There can be *little* doubt that such examples represent the tip of an iceberg.

Table 3: Judgement examples given to annotators



(a) Results on the pilot study data

(b) Results on the development data

Figure 2: Performance of the n-gram count method

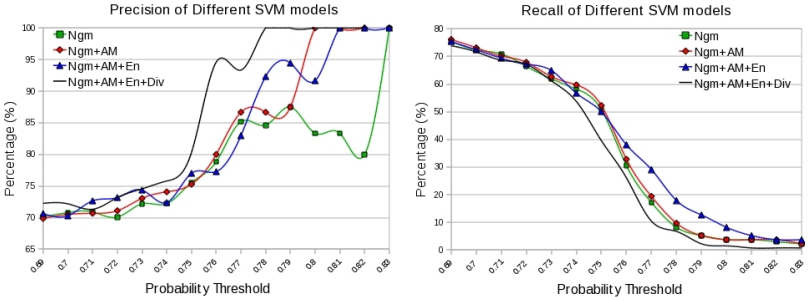
an adjective is deletable. The precision baselines in the pilot study data, development data and test data are 54.2%, 67.0% and 64.0%, respectively.

6.1 Experiments Using N-gram Count Method

We test the n-gram count method on the pilot study data and the development data. Figure 2(a) and Figure 2(b) show the precision and recall curves with respect to different thresholds for the pilot study data and the development data, respectively. For the pilot study data, the best precision 72.1% is achieved with a 48.9% recall by using a threshold equal to 1.05. For the development data, the best precision 84.2% is achieved using a threshold equal to 1.9. However, the recall value drops to 11.9% which means many deletable adjectives are being ignored.

6.2 Experiments using SVM

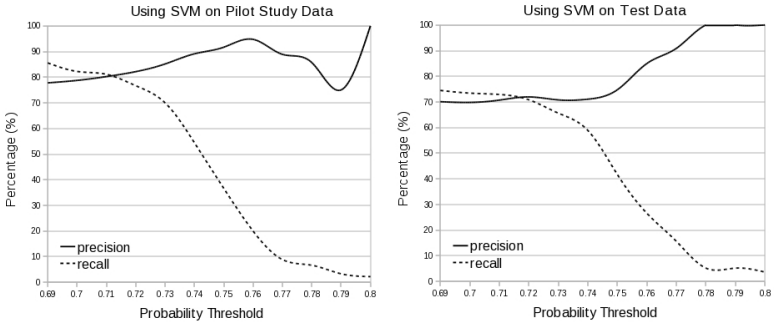
For the SVM learning approach, we first train models with different features and test the models on the development data. Figure 3(a) and Figure 3(b) show the precision and recall curves of the models with probability thresholds greater than 0.69 and lower than 0.83 (since these



(a) Precision curves of the models

(b) Recall curves of the models

Figure 3: Performance of SVM models using different features



(a) Results on the pilot study data

(b) Results on the test data

Figure 4: Performance of the Ngm+AM+En+Div model

values result in a reasonable precision range). In addition, we ignore results that have recall values lower than 10% even though a high precision is achieved. The SVM Ngm model is trained using the 58 features described in Section 4.1. Its best precision is 85.2% (with a recall greater than 10%) which is similar to that achieved by using the n-gram count method, but the corresponding recall is slightly improved to 17.2%. Next we add two association measures MI and LLR to the features and train the model Ngm+AM. The best precision of the Ngm+AM model is 86.7% and the corresponding recall is 19.4%. We then add features by including entropies and modification probabilities described in Section 4.3 and train the Ngm+AM+En model. This model achieves 92.3% precision with 17.9% recall. Finally, the Ngm+AM+En+Div model is trained with the divergence measure added to the features. The best precision of this model is 94.6% with 26.1% recall when the probability threshold 0.76 is used. Since the Ngm+AM+En+Div model achieves the best precision value among all the models, we further evaluate this model using the pilot study dataset and the test dataset.

Figure 4(a) shows the performance of the Ngm+AM+En+Div model on the pilot study data.

With 50% recall on the pilot study data, the SVM model achieves a precision of 90%, while the n-gram method only achieves 72.1% precision at this level of recall. We can see that there is a large improvement on classifying deletable adjectives from undeletable adjectives in the pilot study data compared to both the baseline and the n-gram count method. Finally, we use the probability threshold 0.76 that gives the best precision on the development set to evaluate the pilot study data and the test data. For the pilot study data, the classifier achieves a precision of 94.7% and a recall of 20%; for the test data, the classifier achieves a precision of 85% and a recall of 26.6%. Note that a precision of 100% is not necessarily required because the inter-annotator agreement on the collected human judgements is not 100% and therefore it is not clear whether the precision upper bound on this task is 100%.

Figure 4(b) shows the full range of precision-recall scores using different probability threshold values on the test data.[6] From this figure, we can clearly see the trade-off between precision and recall, which corresponds to the trade-off between imperceptibility and payload for the linguistic steganography application. In practice, steganography users can decide the threshold according to their requirements on the security level and embedding capacity. In addition, since the cover text can be selected by users, the payload can be improved by choosing a text containing more adjectives such as fictions or fairy tales, which might increase the density of deletable adjectives in text.

7 Linguistic Steganography Application

For linguistic steganography, there exists a convenient modularity between the linguistic transformation and the embedding method. In other words, the utility of a specific embedding method does not imply a particular linguistic transformation, although it will put some constraints on what transformation can be used. For example, synonym substitution, paraphrasing and translation can be applied to an embedding method which reconstructs the secret message as concatenating codewords that are directly associated with a choice. We will demonstrate that the adjective deletion technique can be integrated into our earlier Chang and Clark (2010a) secret embedding scheme.

In Chang and Clark (2010a) we proposed a secret embedding method based on text paraphrasing as shown in Figure 5. In the secret embedding phase, a cover text is first divided into embedding units of which each has an equal number of sentences and contains at least one paraphrasable sentence. In this example, the paraphrasable sentences are t_1, t_3, t_4, t_7 and t_8; the text can be divided into three embedding units u_1, u_2 and u_3 with the size equal to three sentences. One secret bit is then embedded in one embedding unit. If the secret bit is 0, all the paraphrasable sentences in the embedding unit are transformed into non-paraphrasable sentences; if the secret bit is 1, the embedding unit remains unchanged. The secret bitstring in this example is 101 so the paraphrasable sentence t_4 in u_2 is transformed into a non-paraphrasable sentence, and u_1 and u_3 are unmodified. The secret extracting can be easily performed by dividing the stego text into embedding units and using the existence of paraphrasable sentences to decide whether the embedding unit represents secret bit 0 or 1. In this embedding scheme, the embedding unit size is treated as the secret key that is only shared between the sender and the receiver.

We can replace the text paraphrasing in the above method with the adjective deletion technique as the linguistic transformation, so that each embedding unit contains at least one deletable

[6] Note that we are not optimising for one single score on the test set, e.g. F-score, but showing the full range of the precision-recall tradeoff that corresponds to a security-payload tradeoff in the steganography setting.

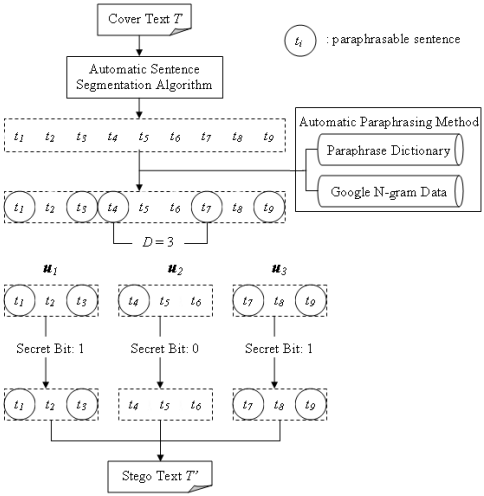


Figure 5: The Chang and Clark (2010a) stegosystem

adjective. If we want to embed 0, all the deletable adjectives will be removed from the embedding unit. Since the deletable adjectives are checked by the proposed model, the removal of the adjectives should achieve a certain level of naturalness in the sentences. If we want to embed 1, the embedding unit will not be modified. It is important to note that the recovery of the secret bitstring does not require the original text. The receiver only needs the secret key to define the size of an embedding unit and the adjective checking model to see whether there is a deletable adjective in an embedding unit.

8 Secret Sharing Scheme

We propose a novel secret sharing scheme based on the adjective deletion technique and text alignment. The secret sharing scheme converts a secret bitstring into two shares, $Share_0$ and $Share_1$, that are camouflaged in the form of natural language text. $Share_0$ holds secret bits as 0s and $Share_1$ holds secret bits as 1s. The order of the 0s and 1s can only be reconstructed by aligning the two texts.

Figure 6 illustrates an example of the secret sharing scheme. The secret bitstring is 101. We first give $Share_0$ and $Share_1$ the same text and use the proposed adjective checking method to determine deletable adjectives in the text. In this example, the n-gram count method with threshold equal to 1 is applied. The adjectives passing the check are *mysterious*, *terrible* and *single*, and one deletable adjective will embed a secret bit. The embedding rule is: to embed a secret bit 0/1, the target adjective is kept in the share that holds 0s/1s, and is removed from the other share. For example, the first secret bit is 1 so *mysterious* is kept in $Share_1$ and is deleted from $Share_0$. Next, we embed the second secret bit 0 by keeping *terrible* in $Share_0$ and removing it from $Share_1$. The third secret bit is 1 so we keep *single* in $Share_1$ and remove it from $Share_0$. Now the secret bitstring 101 is converted into two meaningful texts. The reconstruction of the

Secret bits: 101	Text: "Have you heard of the **mysterious** death of your late boarder Mr. Enoch J. Drebber, of Cleveland?" A **terrible** change came over the woman's face as I asked the question. It was some seconds before she could get out the **single** word "Yes" – and when it did come it was in a husky, unnatural tone.
Embed 1st bit: 1 Target adj: *mysterious*	$Share_0$: "Have you heard of the death of your late boarder Mr. Enoch J. Drebber, of Cleveland?" A **terrible** change came over the woman's face as I asked the question. It was some seconds before she could get out the **single** word "Yes" – and when it did come it was in a husky, unnatural tone. $Share_1$: "Have you heard of the **mysterious** death of your late boarder Mr. Enoch J. Drebber, of Cleveland?" A **terrible** change came over the woman's face as I asked the question. It was some seconds before she could get out the **single** word "Yes" – and when it did come it was in a husky, unnatural tone.
Embed 2nd bit: 0 Target adj: *terrible*	$Share_0$: "Have you heard of the death of your late boarder Mr. Enoch J. Drebber, of Cleveland?" A **terrible** change came over the woman's face as I asked the question. It was some seconds before she could get out the **single** word "Yes" – and when it did come it was in a husky, unnatural tone. $Share_1$: "Have you heard of the **mysterious** death of your late boarder Mr. Enoch J. Drebber, of Cleveland?" A change came over the woman's face as I asked the question. It was some seconds before she could get out the **single** word "Yes" – and when it did come it was in a husky, unnatural tone.
Embed 3rd bit: 1 Target adj: *single*	$Share_0$: "Have you heard of the death of your late boarder Mr. Enoch J. Drebber, of Cleveland?" A **terrible** change came over the woman's face as I asked the question. It was some seconds before she could get out the word "Yes" – and when it did come it was in a husky, unnatural tone. $Share_1$: "Have you heard of the **mysterious** death of your late boarder Mr. Enoch J. Drebber, of Cleveland?" A change came over the woman's face as I asked the question. It was some seconds before she could get out the **single** word "Yes" – and when it did come it was in a husky, unnatural tone.

Figure 6: An example of the secret sharing scheme

secret bitstring can be done by aligning the two texts. The alignment will reveal the positions of the deletable adjectives, which gives the order of the 0s and 1s, and therefore the secret can be extracted. Note that this scheme does not require either the original text or the adjective checking model to recover the secret bitstring.

Conclusion

One of the contributions of this paper is to explore the identification of redundant adjectives in noun phrases. We proposed two methods for checking the sentence naturalness after removing an adjective, which were evaluated by human judgements. The results suggest that the adjective deletion technique is applicable to cryptosystems since the transformation is able to achieve satisfactory imperceptibility leading to a high security level. According to our observations from section A of the BNC, on average there are two deletable adjectives per five sentences. In other words, the payload upper bound of using the adjective deletion technique is around 0.4 bits per sentence if a deletion encodes a secret bit. Apart from the cryptosystem applications, the proposed adjective checking model can also benefit other studies such as sentence compression, text simplification and text summarisation.

Another contribution of this paper is the integration of the adjective deletion technique into an existing stegosystem and the proposal of a novel secret sharing scheme based on adjective deletion. An advantage of our proposed system is that it is somewhat language and domain independent. For future work, we would like to explore more lexical redundancies in other POS, such as adverbs and punctuation, so the payload capacities of our cryptosystems can be further improved.

Acknowledgments

We would like to thank Dr. Laura Rimell and the anonymous reviewers for useful comments and the annotators for their time.

References

Atallah, M. J., McDonough, C. J., Raskin, V., and Nirenburg, S. (2001a). Natural language processing for information assurance and security: an overview and implementations. In *Proceedings of the 2000 workshop on New security paradigms*, pages 51–65, Ballycotton, County Cork, Ireland.

Atallah, M. J., Raskin, V., Crogan, M. C., Hempelmann, C., Kerschbaum, F., Mohamed, D., and Naik, S. (2001b). Natural language watermarking: design, analysis, and a proof-of-concept implementation. In *Proceedings of the 4th International Information Hiding Workshop*, volume 2137, pages 185–199, Pittsburgh, Pennsylvania.

Atallah, M. J., Raskin, V., Hempelmann, C. F., Karahan, M., Topkara, U., Triezenberg, K. E., and Sion, R. (2002). Natural language watermarking and tamperproofing. In *Proceedings of the 5th International Information Hiding Workshop*, pages 196–212, Noordwijkerhout, The Netherlands.

Bergsma, S., Lin, D., and Goebel, R. (2009). Web-scale n-gram models for lexical disambiguation. In *Proc. IJCAI*, pages 1507–1512, Pasadena, California.

Blakley, G. (1979). Safeguarding cryptographic keys. In *Proceedings of the National Computer Conference*, volume 48, pages 313–317. AFIPS Press.

Bolshakov, I. A. (2004). A method of linguistic steganography based on collocationally-verified synonym. In *Information Hiding: 6th International Workshop*, volume 3200, pages 180–191, Toronto, Canada.

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

Chang, C.-Y. and Clark, S. (2010a). Linguistic steganography using automatically generated paraphrases. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 591–599, Los Angeles, California.

Chang, C.-Y. and Clark, S. (2010b). Practical linguistic steganography using contextual synonym substitution and vertex colour coding. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, EMNLP '10, pages 1194–1203, Cambridge, MA.

Chapman, M. and Davida, G. I. (1997). Hiding the hidden: A software system for concealing ciphertext as innocuous text. In *Proceedings of the First International Conference on Information and Communication Security*, volume 1334, pages 335–345, Beijing.

Church, K. W. and Hanks, P. (1990). Word association norms, mutual information, and lexicography. *Comput. Linguist.*, 16:22–29.

Clark, S. and Curran, J. R. (2007). Wide-coverage efficient statistical parsing with CCG and log-linear models. *Comp. Ling.*, 33(4):493–552.

Cohn, T. and Lapata, M. (2008). Sentence compression beyond word deletion. In *Proceedings of the 22nd International Conference on Computational Linguistics (Coling 2008)*, pages 137–144, Manchester, UK.

Dunning, T. (1993). Accurate methods for the statistics of surprise and coincidence. *Comput. Linguist.*, 19:61–74.

Filippova, K. and Strube, M. (2008). Dependency tree based sentence compression. In *Proceedings of the Fifth International Natural Language Generation Conference*, pages 25–32.

Fleiss, J. L., Levin, B., and Paik, M. C. (2003). *Statistical Methods for Rates & Proportions*. Wiley-Interscience, 3rd edition.

Fridrich, J. (2009). *Steganography in Digital Media: Principles, Algorithms, and Applications*. Cambridge University Press, first edition.

Hsu, C.-W., Chang, C.-C., and Lin, C.-J. (2010). A practical guide to support vector classification. Technical report, Department of Computer Science, National Taiwan University.

Kim, M.-Y. (2008). Natural language watermarking for korean using adverbial displacement. In *Multimedia and Ubiquitous Engineering*, pages 576–581, Busan, Korea.

Kim, M.-Y. (2009). Natural language watermarking by morpheme segmentation. In *First Asian Conference on Intelligent Information and Database Systems*, pages 144–149, Dong hoi, Quang binh, Vietnam.

Knight, K. and Marcu, D. (2002). Summarization beyond sentence extraction: A probabilistic approach to sentence compression. *Artificial Intelligence*, 139(1):91–107.

Landis, J. R. and Koch, G. G. (1977). The measurement of observer agreement for categorical data. *Biometrics*, 33(1):159–174.

Lee, L. (1999). Measures of distributional similarity. In *Proceedings of the 37th annual meeting of the Association for Computational Linguistics on Computational Linguistics*, pages 25–32, College Park, Maryland.

Liu, Y., Sun, X., and Wu, Y. (2005). A natural language watermarking based on Chinese syntax. In *Advances in Natural Computation*, volume 3612, pages 958–961, Changsha, China.

Meral, H. M., Sankur, B., Sumru Özsoy, A., Güngör, T., and Sevinç, E. (2009). Natural language watermarking via morphosyntactic alterations. *Computer Speech and Language*, 23(1):107–125.

Meral, H. M., Sevinc, E., Unkar, E., Sankur, B., Ozsoy, A. S., and Gungor, T. (2007). Syntactic tools for text watermarking. In *Proceedings of the SPIE Conference on Security, Steganography, and Watermarking of Multimedia Contents*, volume 6505, San Jose, CA.

Minnen, G., Carroll, J., and Pearce, D. (2001). Applied morphological processing of English. *Nat. Lang. Eng.*, 7:207–223.

Murphy, B. (2001). Syntactic information hiding in plain text. Master's thesis, Trinity College Dublin.

Murphy, B. and Vogel, C. (2007a). Statistically-constrained shallow text marking: techniques, evaluation paradigm and results. In *Proceedings of the SPIE International Conference on Security, Steganography, and Watermarking of Multimedia Contents*, volume 6505, San Jose, California.

Murphy, B. and Vogel, C. (2007b). The syntax of concealment: reliable methods for plain text information hiding. In *Proceedings of the SPIE Conference on Security, Steganography, and Watermarking of Multimedia Contents*, volume 6505, San Jose, CA.

Shamir, A. (1979). How to share a secret. *Commun. ACM*, 22:612–613.

Stutsman, R., Grothoff, C., Atallah, M., and Grothoff, K. (2006). Lost in just the translation. In *Proceedings of the 2006 ACM symposium on Applied computing*, pages 338–345, Dijon, France.

Taskiran, C. M., Topkara, M., and Delp, E. J. (2006). Attacks on linguistic steganography systems using text analysis. In *Proceedings of the SPIE Conference on Security, Steganography, and Watermarking of Multimedia Contents*, volume 6072, pages 97–105, San Jose, CA.

Topkara, M., Riccardi, G., Hakkani-Tür, D., and Atallah, M. (2006a). Natural language watermarking: Challenges in building a practical system. In *Proceedings of the SPIE International Conference on Security, Steganography, and Watermarking of Multimedia Contents*, volume 6072, pages 106–117, San Jose, California.

Topkara, M., Taskiran, C. M., and Delp, E. J. (2005). Natural language watermarking. In *Proceedings of the SPIE Conference on Security, Steganography, and Watermarking of Multimedia Contents*, volume 5681, pages 441–452, San Jose, CA.

Topkara, M., Topkara, U., and Atallah, M. J. (2006b). Words are not enough: sentence level natural language watermarking. In *Proceedings of the ACM Workshop on Content Protection and Security*, pages 37–46, Santa Barbara, CA.

Topkara, U., Topkara, M., and Atallah, M. J. (2006c). The hiding virtues of ambiguity: quantifiably resilient watermarking of natural language text through synonym substitutions. In *Proceedings of the 8th Workshop on Multimedia and Security*, pages 164–174, Geneva, Switzerland.

Venugopal, A., Uszkoreit, J., Talbot, D., Och, F., and Ganitkevitch, J. (2011). Watermarking the outputs of structured prediction with an application in statistical machine translation. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 1363–1372, Edinburgh, Scotland, UK.

Vybornova, M. O. and Macq, B. (2007). A method of text watermarking using presuppositions. In *Proceedings of the SPIE Conference on Security, Steganography, and Watermarking of Multimedia Contents*, volume 6505, San Jose, CA.

Zhu, Z., Bernhard, D., and Gurevych, I. (2010). A monolingual tree-based translation model for sentence simplification. In *Proceedings of the 23rd International Conference on Computational Linguistics*, COLING '10, pages 1353–1361, Beijing, China.

The Secret's in the Word Order: Text-to-Text Generation for Linguistic Steganography

Ching – Yun Chang[1] *Stephen Clark*[1]
(1) University of Cambridge, Computer Laboratory, 15 JJ Thomson Avenue, Cambridge, UK
Ching-Yun.Chang@cl.cam.ac.uk, Stephen.Clark@cl.cam.ac.uk

ABSTRACT
Linguistic steganography is a form of covert communication using natural language to conceal the existence of the hidden message, which is usually achieved by systematically making changes to a cover text. This paper proposes a linguistic steganography method using word ordering as the linguistic transformation. We show that the word ordering technique can be used in conjunction with existing translation-based embedding algorithms. Since unnatural word orderings would arouse the suspicion of third parties and diminish the security of the hidden message, we develop a method using a maximum entropy classifier to determine the naturalness of sentence permutations. The classifier is evaluated by human judgements and compared with a baseline method using the Google n-gram corpus. The results show that our proposed system can achieve a satisfactory security level and embedding capacity for the linguistic steganography application.

KEYWORDS: linguistic steganography, word ordering, surface realisation.

Proceedings of COLING 2012: Technical Papers, pages 511–528,
COLING 2012, Mumbai, December 2012.

1 Introduction

Linguistic steganography is a form of covert communication using natural language to conceal the existence of the hidden message so that the very act of communication is undetectable to an outside observer (human or computer) (Fridrich, 2009). In other words, the covert communication would fail if an outside observer is suspicious of the existence of the hidden message. Note that the outside observer here is not the recipient; the recipient needs to expect to receive a hidden message in order to extract it. (Consider the scenario of covert communication between political activists, where the observer is a government "listening" agency, for example.)

In order to embed messages, a cover text must provide information carriers that can be modified to represent the secret. For example, a lexical substitution-based stegosystem substitutes selected words (the information carriers) with their synonyms so that the concatenation of the bitstrings represented by the synonyms is identical to the secret. Note that an unmodifiable text cannot carry information. Ideally the changes made to the text must be imperceptible, resulting in a high security level. In addition, an ideal linguistic steganography scheme should allow sufficient embedding capacity to achieve efficient information transmission, resulting in a large payload capacity. There is a fundamental trade-off between security and payload since any attempt to embed additional information is likely to increase the chance of introducing anomalies into the text, thus degrading the security level.

Existing studies have exploited different linguistic transformations for the application of steganography, such as lexical substitution (Chapman and Davida, 1997; Bolshakov, 2004; Taskiran et al., 2006; Topkara et al., 2006c; Chang and Clark, 2010b), phrase paraphrasing (Chang and Clark, 2010a), sentence structure manipulations (Atallah et al., 2001a,b; Liu et al., 2005; Meral et al., 2007; Murphy, 2001; Murphy and Vogel, 2007b; Topkara et al., 2006b), semantic transformations (Atallah et al., 2002; Vybornova and Macq, 2007) and text translation (Grothoff et al., 2005; Stutsman et al., 2006; Meng et al., 2011). These transformations often rely on sophisticated NLP tools and resources. For example, a lexical substitution-based stegosystem may require synonym dictionaries, POS taggers, word sense disambiguation tools and language models; a syntactic transformation-based stegosystem may require syntactic or semantic parsers and language generation tools. However, given the current state-of-the-art, such NLP techniques cannot guarantee the transformation's imperceptibility. Hence it is important to evaluate the security level of a stegosystem.

The security assessment methods used so far can be classified into two categories: automatic evaluation and human evaluation. Topkara et al. (2006b) and Topkara et al. (2006a) used Machine Translation (MT) evaluation metrics BLEU and NIST, automatically measuring how close a stego sentence is to the original. Topkara et al. (2006b) admitted that MT evaluation metrics are not sufficient for evaluating stegosystems; for example, BLEU relies on word sequences in the stego sentence matching those in the cover sentence and thus is not suitable for evaluating transformations that change the word order significantly. The other widely adopted evaluation method is based on human judgements. Meral et al. (2007), Kim (2008), Kim (2009) and Meral et al. (2009) asked subjects to edit stegotext for improving intelligibility and style. The fewer edit-hits a transformed text received, the higher the reported security level. Murphy and Vogel (2007b) and Murphy and Vogel (2007a) first asked subjects to rate the acceptability (in terms of plausibility, grammaticality and style) of the stego sentences on a seven-point scale. Then subjects were provided with the originals and asked to judge to what extent meaning was preserved on a seven-point scale. Chang and Clark (2010a) asked subjects to judge whether a

paraphrased sentence is grammatical and whether the paraphrasing retains the meaning of the original. In our work, we also use human judgements to evaluate the proposed stegosystem as this is close to the linguistic steganography scenario. Note that it is hard to compare the security level of different systems since there are no standard methods to measure the security (imperceptibility) of a linguistic steganography system.

The other aspect of the stegosystem evaluation is to calculate the amount of data capable of being embedded in a stego text, which can be quantified in terms of bits per language unit, for example per word or per sentence. Payload measurements can be theoretical or empirical. The theoretical payload measurement only depends on an encoding method and is independent of the quality of a stego text; whereas the empirical measurement takes the applicability of a linguistic transformation, namely the security of a stego text, into consideration and measures the payload capacity while a certain security level is achieved. Most of the payload rates reported in existing work are based on empirical measurements.

For the lexical substitution transformation, Topkara et al. (2005) and Topkara et al. (2006c) achieved an average embedding payload of 0.67 bits per sentence. The payload attained by syntactic transformations was around 0.5 to 1.0 bits per sentence (Atallah et al., 2001b; Topkara et al., 2006b; Meral et al., 2009). Since the ontological semantic transformation is currently impractical, the empirical payload is not available. Another semantic method (Vybornova and Macq, 2007) achieves a payload of 1 bit per sentence. Stutsman et al. (2006) showed that their translation-based stegosystem has a payload of 0.33 bits per sentence.

In this paper, we exploit word ordering as the linguistic transformation. A cover sentence is first used to provide a bag-of-words as input to the Zhang et al. (2012) word ordering realisation system. The generated permutations provide alternatives to the original and thus the cover sentence can play the role of an information carrier. However, not all the permutations are grammatical and semantically meaningful. To solve this problem we train a Maximum Entropy classifier (Berger et al., 1996) to distinguish natural word orders from awkward wordings, and evaluate the classifier using human judgements. Note that the proposed maximum entropy classifier is trained to classify sentence-leval naturalness and thus, it is possible that even individual natural sentences might lead to an unnatural document. Modeling the document-level coherence of modified text would be useful but is outside the scope of our study. In addition, we review some translation-based stegosystems and show that the word ordering transformation can be used with the existing translation-based embedding algorithms. For readers unfamiliar with linguistic steganography, Chang and Clark (2010a) present the general linguistic steganography framework and describe several existing stegosystems using different linguistic transformations.

A contribution of this paper is to create a novel link between word ordering realisation and linguistic steganography. The various permutations provide a possible covert channel for secret communication. To our knowledge, this is the first linguistic steganography system using word ordering as the transformation. In addition, we propose a maximum entropy classifier to determine the naturalness of a permutation. The collected human judgements of sentence naturalness and the resulting classifier may be of use for other NLP applications.

2 Word Ordering-based Steganography

The general word ordering problem is to construct grammatical, fluent sentences from a set of un-ordered input words. There have been some word ordering realisation systems that take a

Rank (binary)	Permutation	Secret Bitstring $s=1$	$s=2$	$s=3$
1 (001)	In our products now there is no asbestos.	1	01	001
2 (010)	No asbestos there is now in our products.	0	10	010
3 (011)	Now in our products there is no asbestos.	1	11	011
4 (100)	There is no asbestos in our products now.	0	00	100
5 (101)	There no asbestos in our products is now.	1	01	101
6 (110)	There now is no asbestos in our products.	0	10	110

Figure 1: Ranked sentence permutations and their secret bits

bag-of-words as input and automatically generate permutations (Wan et al., 2009; Zhang and Clark, 2011; Zhang et al., 2012). Any of these word ordering realisation systems can be treated as a module integrated into the proposed secret embedding method.

For linguistic steganography, there exists a convenient modularity between the linguistic transformation and the embedding method. In other words, a secret embedding method can treat the linguistic transformation as a black box for outputting alternatives for a given cover text. How an alternative encodes secret bits is decided by the embedding algorithm. In the next section, we propose an embedding method designed to use permutations from a word ordering system as alternatives for a given cover sentence. Later, in Section 6 we show that the word ordering technique can also be combined with existing translation-based embedding algorithms.

2.1 The Embedding Method

Before any message exchange can take place, the sender and receiver must share a word ordering system and a method to sort a set of sentence permutations, such as alphabetical ordering, or ordering by realisation quality scores determined by the generation system or a language model. The sorting method must be independent of the cover sentence since the receiver does not receive it, and only those permutations having the same length as the cover are considered during sorting. In addition, the number of secret bits carried by a permutation must be fixed and known by the sender and the receiver.

The sender first turns a cover sentence into a bag-of-words and then uses the pre-agreed word ordering system to generate permutations. After eliminating permutations shorter than the cover sentence (if any), the rest are sorted using the pre-agreed method. The rank of a permutation in the sorted list is converted into a binary string, and the lowest s bits are the secret bits carried by that permutation, where s is the pre-fixed payload. Figure 1 shows six alphabetically ranked permutations and the secret bit(s) carried by them when s is equal to 1, 2 and 3. Note that, in order to embed s bits, there must be at least 2^s permutations in the ordered list; otherwise, there will be at least one desired secret bitstring not carried by any permutation. For example, in Figure 1 secret bitstring 000 and 111 cannot be found when s is equal to 3. Finally, the sender can choose a permutation that represents the secret bitstring as the stego sentence.

To recover the secret bitstring, the receiver first transforms the received stego sentence into a bag-of-words. Since we only consider permutations having the same length as the cover during embedding, the bag-of-words obtained from the stego sentence will be identical to that originally obtained from the cover sentence. Next, the receiver reproduces the ordered

permutations. According to the rank of the stego sentence and the pre-agreed payload of s bits, the receiver can extract the secret bitstring. Note that, in the proposed stegosystem, the receiver can extract the secret without knowing the cover text.

The payload of the system is controlled by the variable s, and the security level depends on the quality of the selected permutations. For example, although "In our products now there is no asbestos" and "There no asbestos in our products is now" both carry secret bits "01" in Figure 1, using the latter would significantly decrease the security level as it is an unnatural sentence. Therefore, in Section 4.2 we develop a Maximum Entropy classifier to determine the naturalness of a permutation which can be integrated into the stegosystem during data embedding. Note that, in this paper, we do not investigate the document-level coherence of stego text generated using the word ordering transformation since this requires sophisticated knowledge of natural language semantics and pragmatics. Instead, we tackle the problem of distinguishing the naturalness of a sentence permutation in isolation from the rest of a document.

3 Resources

3.1 Word Ordering Realisation System

The word ordering realisation system plays a crucial role in the proposed steganography method as it largely determines the security level and payload capacity of the stegosystem. The sender in the proposed scheme uses the n-best list output from a word ordering system as the set of possible alternatives for a cover sentence; so the security level largely depends on the quality of the n-best list. In addition, the more acceptable permutations an n-best list has, the larger the payload the stegosystem can use.

Some recent work used syntax models to certify the grammaticality of generated sentences. The combination of permutation and syntactic modelling results in a large search space, which was tackled using heuristic search (Wan et al., 2009; Zhang and Clark, 2011; Zhang et al., 2012). Wan et al. (2009) use a dependency grammar to model word ordering and apply greedy search to find the best permutation. Zhang and Clark (2011) use a syntax-based discriminative model together with best-first search to find the highest scoring permutation plus CCG derivation. Zhang et al. (2012) is an extension of Zhang and Clark (2011) using online large-margin training and incorporating a large-scale language model. The three approaches are evaluated using the generation task of word order recovery, which is to recover the original word order from an input bag-of-words. The BLEU scores reported by Wan et al. (2009), Zhang and Clark (2011) and Zhang et al. (2012) are 33.7, 40.1 and 43.8, respectively, on WSJ newspaper sentences. In this paper, we use the Zhang et al. (2012) system to generate n-best permutations for a cover sentence, but, in practice, any word ordering realisation system can be integrated into the proposed word ordering-based stegosystem.

3.2 Human Judgement Corpus

Since not all the sentence permutations generated by the Zhang et al. (2012) system are grammatical and semantically meaningful, we develop a maximum entropy classifier to determine the naturalness of permutations. In order to have a labelled corpus for training and testing the classifier, we first randomly selected 765 sentences having length between 8 and 25 tokens from sections 02-21 of the Penn Treebank (Marcus et al., 1993) as the cover sentences. The restriction on the sentence length is because a short sentence may not have enough good permu-

Score	Explanation
1	Completely or largely non-fluent, and/or completely or largely lacking in meaning.
2	Very awkward wording, major punctuation errors, and/or logical errors, but still possible to understand.
3	Slightly awkward but still relatively fluent, clear and logical; may contain slightly awkward wording and/or minor punctuation errors.
4	Perfectly natural – both grammatical and semantically meaningful.

Figure 2: Rating scale and guidelines for human evaluation

Score	Permutation
3	As a price-conscious shopper, so far, Manville hasn't bought much.
1	So, as a price-conscious shopper, far Manville hasn't bought much.
1	Far so, as a price-conscious shopper, Manville hasn't bought much.
4	So far Manville, as a price-conscious shopper, hasn't bought much.
1	Far Manville, as a price-conscious shopper, so hasn't bought much.

Figure 3: A set of five permutations rated by a subject

tations for the steganography application, and a long cover sentence increases the complexity of finding acceptable word orders from the bag-of-words and therefore is unlikely to result in good permutations.

For each cover sentence, we created a bag-of-words as input and generated 100 permutations using the Zhang et al. (2012) system. For 88% of the sentences, the original cover sentence is in the 100-best list. This not only serves as a sanity check for the realisation system, but also means the original sentence can be used to carry secret bits without any modification.

To cut down a 100-best list for the human evaluation, we only keep five permutations that retain the most grammatical relationships of the original cover sentence since these permutations are more likely to convey the same meaning as the original. We parsed the cover sentences and their permutations using a CCG parser (Clark and Curran, 2007), and calculated a dependency F-score for each permutation by comparing the CCG predicate-argument dependencies of the permutation and its original. For each cover sentence, the top five F-score permutations which are different from the cover were chosen for human annotation, which results in 3,825 sentences (765 sets of 5 permutations).

A total of 34 native English speakers were asked to judge the naturalness of the sentence permutations on a 4-point scale. The guidelines are shown in Figure 2. The annotations were carried out via a web interface; on each page, a subject was presented with 5 permutations consisting of the same words, and a total of 125 permutations (25 sets) were annotated by each subject. Figure 3 shows a set of five permutations along with the scores rated by a subject. In order to calculate the inter-annotator agreement, 425 permutations were selected to be judged by two annotators.

For the steganography application, those permutations rated as perfectly natural (score 4) can achieve a high security level and are treated as positive data when training a Maximum Entropy classifier; those permutations with scores lower than 4 are treated as negative data. After converting the scores into a positive/negative representation, we measured the inter-annotator

agreement on the binary labelled data using Fleiss' kappa (Fleiss et al., 2003). The resulting kappa score of 0.54 for the data represents "moderate" agreement according to Landis and Koch (1977). There were 47 out of the 425 agreement-measuring sentences that received different labels after applying the positive/negative representation, for which the second author of this paper made the definitive judgement. In the end, the collected human judgement corpus contained 478 positive (perfectly natural) permutations and 3,347 negative examples.

According to the human judgements, 321 out of the 765 cover sentences have at least one natural sounding permutation in the top five F-score permutations. Therefore, the upper bound of the number of possible information carriers is roughly 42% of the cover sentences. Next, since there are studies using automatic evaluation metrics to evaluate the security level of a stegosystem as described in Section 1, we observe how well the BLEU score of a permutation correlates with the human judgement. For those multi-judged sentences, an average score is assigned. Both Pearson's correlation coefficient and Spearman's rank correlation coefficient between the human judged scores and the BLEU scores are calculated, which are 0.10 and 0.09, respectively, and are significant at $p < 0.001$. This result indicates there is little association between the human judgement of sentence naturalness and the BLEU score, indicating the need for a manual evaluation to determine the likely security level.

4 Sentence Naturalness Classification

According to the human judgement corpus, most of the machine-generated permutations are not natural. In addition, as described in Section 2, the proposed secret embedding algorithm sometimes involves choosing a permutation from a group of candidates that all encode the same secret bit(s). Therefore, it is crucial to develop a method that can distinguish acceptable permutations from those having awkward wordings in a word ordering-based stegosystem. It is important to note that the checking method can take the original sentence into consideration because the permutation selection happens at the secret embedding stage and is not needed during the decoding. Having the cover sentence available at the checking stage is a feature we will exploit.

A research area that relates to the proposed permutation checking method is realisation ranking (Cahill and Forst, 2010; White and Rajkumar, 2012) where a system is given a set of text realisations and is asked to rate each text in the set. However, in the realisation ranking task there is no "cover text". Since our methods require the knowledge of the original text, they cannot be applied to the realisation ranking task.

In this section we first explain a baseline method using the Google n-gram corpus (Brants and Franz, 2006) to check whether a particular word ordering has been used frequently on the Web. Then we propose another approach using some syntactic features to train a Maximum Entropy classifier (Berger et al., 1996). Both methods require a score/probability threshold to decide the acceptability of a permutation.

4.1 Google N-gram Method

The Google n-gram corpus (Brants and Franz, 2006) contains frequency counts for n-grams from unigrams through five-grams obtained from over 1 trillion word tokens of English Web text collected in January 2006. Only n-grams appearing more than 40 times were kept in the corpus. The Google n-gram corpus has been applied to many NLP tasks such as spelling correction (Carlson et al., 2008; Islam and Inkpen, 2009), multi-word expression classification

(Kummerfeld and Curran, 2008) and lexical disambiguation (Bergsma et al., 2009). Recently, in Chang and Clark (2010b) and Chang and Clark (2010a) we have used the corpus to check the text paraphrasing grammaticality and the synonym acceptability in context in our earlier steganography systems. Therefore, we propose a baseline method similar to Chang and Clark (2010b) and Chang and Clark (2010a) using the Google n-gram corpus to calculate a score based on the n-gram counts before and after word ordering. The task is as follows: given a cover sentence and corresponding permutation, decide if the permutation is acceptable. The baseline method will do so by comparing Google n-gram counts from the two sentences.

In the Google n-gram corpus, sentence boundaries are marked by <S> and </S>, where <S> represents the beginning of a sentence and </S> represents the end of a sentence. Both tags are treated like word tokens during the n-gram collection. Hence, after tokenising the cover sentence and its corresponding permutation, we add <S> and </S> tags to the beginning and end of the sentences as shown in Figure 4. Then we extract every bi- to five-gram from the cover sentence and the permutation. Since we are only interested in newly generated wordings in the permutation, n-grams that appear in both the cover and the permutation are eliminated, and the remaining n-grams and their Google n-gram frequencies are used to calculate the score. For example, after comparing the two sentences, there are 21 n-grams from the cover and 21 n-grams from the permutation left in Figure 4. We sum up all the logarithmic counts[1] for the cover and permutation n-grams and derive Sum_{Cover} and $Sum_{Permutation}$. The *Score* of a permutation is defined as the ratio of $Sum_{Permutation}$ and Sum_{Cover}, which measures how much the *Sum* varies after performing the word ordering. In the given example, the *Score* of the permutation is 0.81. Similar to Chang and Clark (2010b) and Chang and Clark (2010a), a score threshold is needed to determine the acceptability of a permutation. Even though this baseline method is only an n-gram count comparison, Bergsma et al. (2009) show that the approach works well for lexical disambiguation tasks and produces comparable performance to other more complex methods.

4.2 Maximum Entropy Classifier

In addition to the baseline method, we propose a machine learning approach to classify natural and unnatural permutations. We choose the method of maximum entropy modelling (MaxEnt for short) because of its proven performance for NLP tasks, such as part-of-speech tagging (Ratnaparkhi et al., 1996; Curran and Clark, 2003), parsing (Ratnaparkhi, 1999; Johnson et al., 1999) and language modelling (Rosenfeld, 1996), and the ease with which features can be included in the model (Ratnaparkhi, 1999). In addition, some work has shown that MaxEnt is viable for ranking the fluency of machine generated sentences (Nakanishi et al., 2005; Velldal and Oepen, 2006; Velldal, 2008). The concept of MaxEnt is to use observed features about a certain event (y) occurring in the context (x) to estimate a probability model $p(y|x)$. Its canonical form is:

$$p(y|x) = \frac{1}{Z(x)} exp \sum_{i=1}^{n} \lambda_i f_i(x, y)$$

where $Z(x)$ is a normalisation constant over all events in context x and λ_i is the weight of the feature $f_i(x, y)$. The standard way to train a maximum entropy model is to use conditional maximum likelihood (with a Gaussian prior for smoothing), which is equivalent to picking the most uniform model subject to constraints on the feature expectations (Berger et al., 1996).

[1] log(0) and division by zero are taken to be zero.

Cover: <S> There is no asbestos in our products now . </S>		Permu: <S> In our products now there is no asbestos . </S>	
log *freq*	n-gram	log *freq*	n-gram
19.1	<S> There	20.3	<S> In
11.6	asbestos in	14.3	now there
17.6	now .	12.0	asbestos .
17.8	<S> There is	15.2	<S> In our
6.1	no asbestos in	0	products now there
6.0	asbestos in our	13.0	now there is
9.3	products now .	6.5	no asbestos .
17.6	now . </S>	12.0	asbestos . </S>
16.4	<S> There is no	6.7	<S> In our products
5.1	is no asbestos in	0	our products now there
0	no asbestos in our	0	products now there is
0	asbestos in our products	11.1	now there is no
6.8	our products now .	3.7	is no asbestos .
0	products now . </S>	0	no asbestos . </S>
4.0	<S> There is no asbestos	0	<S> In our products now
4.8	There is no asbestos in	0	In our products now there
0	is no asbestos in our	0	our products now there is
0	no asbestos in our products	0	products now there is no
0	asbestos in our products now	0	now there is no asbestos
0	in our products now .	0	there is no asbestos .
0	our products now . </S>	0	is no asbestos . </S>
$Sum_{Cover} = 142.1$		$Sum_{Permutation} = 114.9$	

Figure 4: An example of the Google n-gram method

After training a maximum entropy classifier, we can calculate the probabilities of a permutation being a natural sentence according to the feature weights. Our proposed method says that a permutation is natural if the ratio of its naturalness probability to its unnaturalness probability is greater than a threshold α. The threshold α controls the trade-off between precision and recall of the maximum entropy classifier and can be decided by steganography users. The MaxEnt implementation we used was from the Curran and Clark (2003) tagger, adapted for classification rather than sequence tagging.

4.2.1 Features

In the formulation of MaxEnt we use, a feature f_i is an "indicator function" on events which simply indicates the presence of a feature. The first feature we included is the Levenshtein distance (Levenshtein, 1966) which measures the minimum number of edits needed to transform one cover sentence into its permutation, with the allowable edit operations being insertion, deletion, or substitution. After deriving the edit distance d, an indicator "EDIST_D" becomes the feature for that permutation, where D=$\lfloor log_2 d \rfloor$. For example, a permutation with an edit distance 4 and another permutation with an edit distance 5 both have the same indicator function "EDIST_2". In addition, if the difference between a permutation and its original sentence is only a single word movement, we add the *POS* tag of the moved word to the indicator function, so the feature becomes "EDIST_1-*POS*".

The second type of feature is derived by comparing the Stanford typed dependencies (De Marneffe and Manning, 2008) of a permutation and its original sentence. The Stanford typed dependencies provide descriptions of the grammatical relationships as well as semantically contentful information in a sentence, which can be obtained from the Stanford parser (De Marneffe et al., 2006). The Stanford typed dependencies are triples denoting a relation between a governor and a dependent. For example, *amod(wine, red)* denotes that *red* is an adjectival modifier of *wine*, and *agent(killed, spy)* denotes that an agent *spy* is the complement of a verb *killed*.

Cover: There is no asbestos in our products now.	Permutation: Our products there is no asbestos in now.
Stanford typed dependencies:	Stanford typed dependencies:
expl(is-VBZ-2, there-EX-1)	poss(products-NNS-2, our-PRP$-1)
root(ROOT-ROOT-0, is-VBZ-2)	nsubj(asbestos-NN-6, products-NNS-2)
det(asbestos-NN-4, no-DT-3)	advmod(asbestos-NN-6, there-RB-3)
nsubj(is-VBZ-2, asbestos-NN-4)	cop(asbestos-NN-6, is-VBZ-4)
prep(asbestos-NN-4, in-IN-5)	det(asbestos-NN-6, no-DT-5)
poss(products-NNS-7, our-PRP$-6)	root(ROOT-ROOT-0, asbestos-NN-6)
pobj(in-IN-5, products-NNS-7)	prep(asbestos-NN-6, in-IN-7)
advmod(is-VBZ-2, now-RB-8)	pobj(in-IN-7, now-RB-8)
dependency indicator functions of the permutation: P_dep_nsubj, P_deppos_nsubj_NN_NNS, P_dep_advmod, P_deppos_advmod_NN_RB, P_dep_cop, P_deppos_cop_NN_VBZ, P_dep_root, P_deppos_root_ROOT_NN, P_dep_pobj, P_deppos_pobj_IN_RB, R_dep_poss, R_deppos_poss_NNS_PRP$_0, R_dep_det, R_deppos_det_NN_DT_0, R_dep_prep, R_deppos_prep_NN_IN_0	

Figure 5: An example of the dependency indicator functions

We first parse a permutation and its original sentence using the Stanford parser and compare their Stanford typed dependencies. If a dependency *TYPE(WORD1, WORD2)* in the permutation cannot be found in the original, two indicator functions "P_dep_*TYPE*" and "P_deppos_*TYPE_POS1_POS2*" are added to the permutation's feature set, where *POS1* and *POS2* are the POS tags of *WORD1* and *WORD2*, respectively. If a dependency *TYPE(WORD1, WORD2)* in the permutation is the same as that in the original, two indicator functions "R_dep_*TYPE*" and "R_deppos_*TYPE_POS1_POS2_DISTANCE*" are added to the permutation's feature set, where *POS1* and *POS2* are the POS tags of *WORD1* and *WORD2*, and *DISTANCE* is the difference of the distance between the two words compared to the original. Figure 5 shows the dependency indicator functions of the permutation "our products there is no asbestos in now". In this example, *nsubj(asbestos, products)* is a newly generated relation after word ordering so two indicator functions *P_dep_nsubj* and *P_deppos_nsubj_NN_NNS* are added to the permutation's feature set; *poss(products, our)* is a recovered relation from the original and the distance between *product* and *our* remains the same as that in the origial so two indicator functions *R_dep_poss* and *P_deppos_poss_NNS_PRP$_0* are added to the permutation's feature set.

5 Experiments and Results

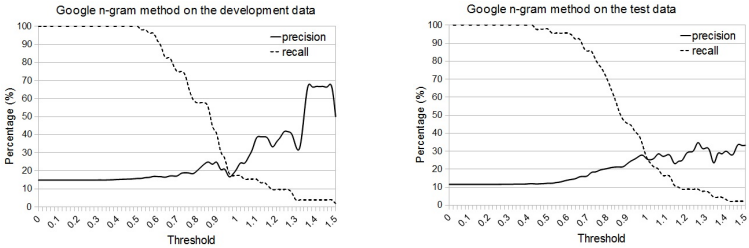
We evaluate the Google n-gram method and the maximum entropy classifier using the collected human judgements. The performance of the systems is measured in precision and recall over the natural permutations (i.e. the positive examples in the test set). Note that the trade-off between precision and recall implies the trade-off between security and payload capacity from a steganography perspective. A higher precision value means that the system-passed permutations are of high quality so using those permutations as stego sentences is unlikely to arouse suspicion; whereas a larger recall value can be interpreted as the system making the most of natural permutations and therefore embedding as much data as possible.

5.1 Data Sets

We divided the collected human judgement corpus described in Section 3.2 into a 2700-instance training set, a 350-instance development set and a 775-instance test set. The development set was mainly used for preliminary experimentation. We will present results on the development and test sets. Note that the 425 multi-judged sentences are all included in the test set. We treat

	Training set	Development set	Test set
number of positives	467	52	90
number of negatives	2,364	298	685

Table 1: Statistics of the experimental data sets



(a) Results on the development set

(b) Results on the test set

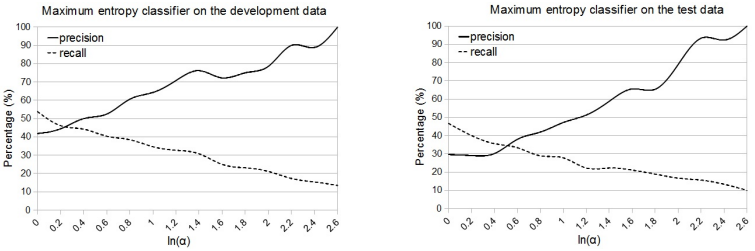
Figure 6: Performance of the Google n-gram method

natural permutations as positives and unnatural sentences as negatives. Since the number of negatives is 7 times more than the number of positives in the training set, we added another 131 positives annotated by the authors to the training set (but not the test set), in an attempt to address the imbalance. Table 1 presents the statistics of the data sets.

5.2 Experiments Using Google N-gram Method

We evaluated the Google n-gram method on the development set. Figure 6(a) shows the precision and recall curves with respect to different threshold values. The best precision achieved by the system is 66.7% with a very low recall of 3.9% when the threshold is equal to 1.36. Then we use the threshold 1.36 to classify positive and negative data in the test set. The derived precision and recall values are 28.6% and 4.4%, respectively. Figure 6(b) gives the precision and recall curves obtained by using the Google n-gram method on the test data. From the diagram we can see that, even when a threshold 1.26 is chosen, the best precision on the test set is only 34.8%, which is not appropriate for the steganography application since the low precision value would result in an unnatural stego text and hence fail the secret communication requirement.

A possible explanation of the poor performance of the n-gram baseline is that the n-gram method might be useful for checking local word changes (e.g. synonym substitution), but not the whole sentence rearrangement. In addition, longer n-grams are not frequently found in the Google n-gram corpus according to our data so in these cases the n-gram method only relies on checking the changes in lower-order n-grams, such as bi-grams or tri-grams.



(a) Results on the development set

(b) Results on the test set

Figure 7: Performance of the maximum entropy classifier

5.3 Experiments Using Maximum Entropy Classifier

Next we train a maximum entropy classifier using sentences in the training set. Each permutation in the training set is first represented by its indicator functions and is labelled as either a positive or a negative instance according to the human judgements. A total of 4,490 indicator functions are extracted from the training set. Since in the training set, the ratio of positives to negatives is about 1:5, we duplicate the positives 5 times to balance the amount of positives and negatives in the training set. The trained 5,815 weights are used to calculate the probabilities of a test instance being positive and negative. As mentioned in Section 4.2, the system determines an instance as positive if:

$$\frac{exp\sum_{i=1}^{n}\lambda_i f_i(x,positive)}{exp\sum_{i=1}^{n}\lambda_i f_i(x,negative)} > \alpha \implies \sum_{i=1}^{n}\lambda_i f_i(x,positive) - \sum_{i=1}^{n}\lambda_i f_i(x,negative) > ln(\alpha)$$

We observe the precision and recall values of the classifier with different α values. Figure 7(a) shows the performance of the classifier on the development set. The classifier achieves a 90% precision with 17.3% recall when the threshold $ln(\alpha)$ is equal to 2.2. A precision of 100% can be obtained by raising the threshold to 2.6 with the corresponding recall being 13.5%. Since the inter-annotator agreement on the collected human judgements is not 100%, as shown in Section 3.2, it is not clear whether the 90% precision achieved by the classifier really means that the remaining 10% sentences (false positives) would be viewed as suspicious in a real steganography setting. Therefore, we consider 90% to be a high level of precision/security.

The same classifier is then used to determine natural permutations in the test set and the $ln(\alpha)$ is set to 2.2 since this setting gives a satisfactory imperceptibility and payload capacity for the development set. The classifier achieves a precision of 93.3% and a recall of 15.6% with the 2.2 threshold, which again provides a confident security level and reasonable embedding capacity for the steganography application. This result is much better than the precision of 34.8% achieved by the baseline Google n-gram method. Since the training data used in our classifier is imbalanced (the number of negatives is five times more than that of positives), the features observed from positive data may not be enough to gain a higher recall. Therefore, we expect the recall value can be improved using more balanced training data.

In order to show the trade-off between precision and recall, which corresponds to the trade-off between imperceptibility and payload capacity for the linguistic steganography application, the precision and recall curves of the classifier on the test set are given in Figure 7(b). Note that we are not optimising on the test set; Figure 7(b) is just a demonstration of where on the precision-recall tradeoff a practical stegosystem might lie. In practice, the threshold value would depend on how steganography users want to trade off security for payload.

6 Word Ordering-based Stegosystem

In this section we first review the existing translation-based stegosystems. Then we explain how those translation-based embedding algorithms can not only work with a machine translation system, but can also be combined with a word ordering realisation system.

6.1 Translation-based Steganography

The first translation-based stegosystem was proposed by Grothoff et al. (2005). In their method, the sender and the receiver first agree on a set of machine translation systems that generate multiple translations for a given cover sentence. According to the translation probabilities, the sender then encodes each translation using Huffman coding (Huffman, 1952). Using Huffman coding, translations with higher probability (higher quality) have shorter codes and thus are more likely to match the secret bitstrings. After choosing translations that represent the secret bits, both the cover text and the stegotext are sent to the receiver. To extract the message, the receiver uses the same set of translation systems to generate translations for each cover sentence and runs the Huffman coding algorithm. Using the stegotext, the receiver can reconstruct the secret.

Grothoff et al. (2005) also proposed another scheme which does not require sending the original text to the receiver. Instead of using a set of machine translation systems as the secret key, the sender and the receiver share a hash function that transforms a sentence into a bitstring. The sender first generates multiple translations for a given cover sentence and then hashes each translation into a bitstring. A translation having its least significant bit as identical to the secret bit is selected as the stego sentence. To recover the message, without knowing the original text, the receiver only needs to compute the hash codes of the received sentences and concatenate the lowest bits of every stego sentence. Unlike the previous method that may be able to embed more than one bit in a translation, this embedding scheme has an upper bound of 1 bit per sentence.

Later Stutsman et al. (2006) improved the payload capacity of the hash function embedding scheme by introducing a header (h bits) to indicate that b bits are embedded in a translation, where h is shared between the sender and the receiver, and b is the integer represented by the header bits. The lowest h bits of a translation hash bitstring are the header bits and the lowest $[h{+}1, h{+}b]$ bits carry the secret. For example, assume $h = 2$; a hash bitstring "...10111" has header bits "11" to indicate 3 bits are embedded in this translation, and the three secret bits are "101". The problem of using a hash function is that the generation of a desired bitstring cannot be guaranteed. Therefore, error correction codes must be used in this protocol when there is no feasible hash code available, which increase the size of the transmission data.

Since Grothoff et al. (2005) and Stutsman et al. (2006) use multiple machine translation systems to generate alternative translations, the selected stego sentences may not have uniform style and therefore it is easy to detect the existence of the secret message (Meng et al., 2010;

Chen et al., 2011). Instead of obtaining alternative translations from multiple translation systems, Meng et al. (2011) use a statistical machine translation system to generate the n-best translations for a given cover sentence. Since translations are from one system, each of them is more similar to the rest than that derived from another translation system. Meng et al. (2011) show that the existing steganalysis methods cannot distinguish ordinary translation text and the stegotext generated by their stegosystem.

6.2 Using Word Ordering in Translation-based Embedding

The translation-based embedding algorithms described in the previous section take a cover sentence as the input and choose one of the translations as the stego sentence. We can easily replace the machine translation system(s) in these algorithms with a word ordering realisation system: a cover sentence is used to provide a bag-of-words as input, and the generated permutations can be seen as the "translations" of the cover sentence. One difference between the proposed embedding method and the translation-based embedding methods is that the former restricts the length of a permutation to be the same as the cover sentence so that the receiver is able to recover the list of permutations; while the latter allows a permutation to only include a subset of the input words and therefore provides more choices for a given cover sentence. However, dropping words introduces the risk of deleting information in the cover sentence and may lead to incoherence in the resulting text.

Conclusion

In this paper we have explored the applicability of using word ordering for linguistic steganography. We proposed a steganography method using word ordering as the linguistic transformation and also showed that the word ordering technique can be applied to existing translation-based embedding algorithms. As well as the embedding method, we also proposed a method for determining the naturalness of sentence permutations by training a maximum entropy classifier. The classifier was evaluated by human judgements and compared with a baseline method using the Google n-gram corpus. The results show that the classifier achieves 93.3% precision with 15.6% recall on the test set, which is much better than the best precision of 34.8% achieved by the Google n-gram method. In addition, the ultimate precision of 100% is reported from the experiments when using a higher probability threshold. This means the proposed maximum entropy classifier can provide a high security level for the linguistic steganography application.

For future work, it is worth tackling the problem of low embedding capacity in the existing linguistic stegosystems compared with other steganography systems using images or audios as the cover medium. For example, although the proposed classifier can achieve 100% precision on the task of determining natural sentence permutations, the corresponding recall of 10% means that many available permutations are ignored, which is a waste of information carriers.

Acknowledgments

We would like to thank Dr. Yue Zhang for providing the word ordering realisation system. In addition, we would like to thank Dr. Laura Rimell, Dr. Yue Zhang and the anonymous reviewers for useful comments and the annotators for their time.

References

Atallah, M. J., McDonough, C. J., Raskin, V., and Nirenburg, S. (2001a). Natural language processing for information assurance and security: an overview and implementations. In

Proceedings of the 2000 workshop on New security paradigms, pages 51–65, Ballycotton, County Cork, Ireland.

Atallah, M. J., Raskin, V., Crogan, M. C., Hempelmann, C., Kerschbaum, F., Mohamed, D., and Naik, S. (2001b). Natural language watermarking: design, analysis, and a proof-of-concept implementation. In *Proceedings of the 4th International Information Hiding Workshop*, volume 2137, pages 185–199, Pittsburgh, Pennsylvania.

Atallah, M. J., Raskin, V., Hempelmann, C. F., Karahan, M., Topkara, U., Triezenberg, K. E., and Sion, R. (2002). Natural language watermarking and tamperproofing. In *Proceedings of the 5th International Information Hiding Workshop*, pages 196–212, Noordwijkerhout, The Netherlands.

Berger, A., Pietra, V., and Pietra, S. (1996). A maximum entropy approach to natural language processing. *Computational linguistics*, 22(1):39–71.

Bergsma, S., Lin, D., and Goebel, R. (2009). Web-scale n-gram models for lexical disambiguation. In *Proceedings of the 21st International Joint Conference on Artifical Intelligence*, pages 1507–1512, Pasadena, CA.

Bolshakov, I. A. (2004). A method of linguistic steganography based on collocationally-verified synonym. In *Information Hiding: 6th International Workshop*, volume 3200, pages 180–191, Toronto, Canada.

Brants, T. and Franz, A. (2006). Web 1T 5-gram corpus version 1.1. Technical report, Google Research.

Cahill, A. and Forst, M. (2010). Human evaluation of a german surface realisation ranker. In *Empirical Methods in Natural Language Generation*, volume 5790, pages 201–221. Springer.

Carlson, A., Mitchell, T. M., and Fette, I. (2008). Data analysis project: Leveraging massive textual corpora using n-gram statistics. Technical report, School of Computer Science, Carnegie Mellon University.

Chang, C.-Y. and Clark, S. (2010a). Linguistic steganography using automatically generated paraphrases. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 591–599, Los Angeles, California.

Chang, C.-Y. and Clark, S. (2010b). Practical linguistic steganography using contextual synonym substitution and vertex colour coding. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 1194–1203, Cambridge, MA.

Chapman, M. and Davida, G. I. (1997). Hiding the hidden: A software system for concealing ciphertext as innocuous text. In *Proceedings of the First International Conference on Information and Communication Security*, volume 1334, pages 335–345, Beijing.

Chen, Z., Huang, L., Meng, P., Yang, W., and Miao, H. (2011). Blind linguistic steganalysis against translation based steganography. *Digital Watermarking*, pages 251–265.

Clark, S. and Curran, J. R. (2007). Wide-coverage efficient statistical parsing with CCG and log-linear models. *Comp. Ling.*, 33(4):493–552.

Curran, J. and Clark, S. (2003). Investigating GIS and smoothing for maximum entropy taggers. In *Proceedings of the tenth conference on European chapter of the Association for Computational Linguistics-Volume 1*, pages 91–98.

De Marneffe, M., MacCartney, B., and Manning, C. (2006). Generating typed dependency parses from phrase structure parses. In *Proceedings of LREC*, volume 6, pages 449–454.

De Marneffe, M. and Manning, C. (2008). The Stanford typed dependencies representation. In *Coling 2008: Proceedings of the workshop on Cross-Framework and Cross-Domain Parser Evaluation*, pages 1–8.

Fleiss, J. L., Levin, B., and Paik, M. C. (2003). *Statistical Methods for Rates & Proportions*. Wiley-Interscience, 3rd edition.

Fridrich, J. (2009). *Steganography in Digital Media: Principles, Algorithms, and Applications*. Cambridge University Press, first edition.

Grothoff, C., Grothoff, K., Alkhutova, L., Stutsman, R., and Atallah, M. (2005). Translation-based steganography. In *Information Hiding*, pages 219–233. Springer.

Huffman, D. (1952). A method for the construction of minimum-redundancy codes. *Proceedings of the IRE*, 40(9):1098–1101.

Islam, A. and Inkpen, D. (2009). Real-word spelling correction using Google Web IT 3-grams. In *EMNLP '09: Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 1241–1249, Singapore.

Johnson, M., Geman, S., Canon, S., Chi, Z., and Riezler, S. (1999). Estimators for stochastic unification-based grammars. In *Proceedings of the 37th annual meeting of the Association for Computational Linguistics on Computational Linguistics*, pages 535–541.

Kim, M. (2008). Natural language watermarking for korean using adverbial displacement. In *Multimedia and Ubiquitous Engineering*, pages 576–581.

Kim, M. (2009). Natural language watermarking by morpheme segmentation. In *Intelligent Information and Database Systems*, pages 144–149.

Kummerfeld, J. K. and Curran, J. R. (2008). Classification of verb particle constructions with the Google Web 1T Corpus. In *Proceedings of the Australasian Language Technology Association Workshop 2008*, pages 55–63, Hobart, Australia.

Landis, J. R. and Koch, G. G. (1977). The measurement of observer agreement for categorical data. *Biometrics*, 33(1):159–174.

Levenshtein, V. (1966). Binary codes capable of correcting deletions, insertions and reversals. *Soviet Physics Doklady*, 10:707–710.

Liu, Y., Sun, X., and Wu, Y. (2005). A natural language watermarking based on Chinese syntax. In *Advances in Natural Computation*, volume 3612, pages 958–961, Changsha, China.

Marcus, M., Marcinkiewicz, M., and Santorini, B. (1993). Building a large annotated corpus of english: The penn treebank. *Computational linguistics*, 19(2):313–330.

Meng, P., Hang, L., Chen, Z., Hu, Y., and Yang, W. (2010). STBS: A statistical algorithm for steganalysis of translation-based steganography. In *Information Hiding*, pages 208–220. Springer.

Meng, P., Shi, Y., Huang, L., Chen, Z., Yang, W., and Desoky, A. (2011). LinL: Lost in n-best list. In *Information Hiding*, pages 329–341. Springer.

Meral, H. M., Sankur, B., Sumru Özsoy, A., Güngör, T., and Sevinç, E. (2009). Natural language watermarking via morphosyntactic alterations. *Computer Speech and Language*, 23(1):107–125.

Meral, H. M., Sevinc, E., Unkar, E., Sankur, B., Ozsoy, A. S., and Gungor, T. (2007). Syntactic tools for text watermarking. In *Proceedings of the SPIE Conference on Security, Steganography, and Watermarking of Multimedia Contents*, volume 6505, San Jose, CA.

Murphy, B. (2001). Syntactic information hiding in plain text. Master's thesis, Trinity College Dublin.

Murphy, B. and Vogel, C. (2007a). Statistically-constrained shallow text marking: techniques, evaluation paradigm and results. In *Security, Steganography, and Watermarking of Multimedia Contents IX*, volume 6505, San Jose, CA.

Murphy, B. and Vogel, C. (2007b). The syntax of concealment: reliable methods for plain text information hiding. In *Proceedings of the SPIE Conference on Security, Steganography, and Watermarking of Multimedia Contents*, volume 6505, San Jose, CA.

Nakanishi, H., Miyao, Y., and Tsujii, J. (2005). Probabilistic models for disambiguation of an HPSG-based chart generator. In *Proceedings of the Ninth International Workshop on Parsing Technology*, pages 93–102.

Ratnaparkhi, A. (1999). Learning to parse natural language with maximum entropy models. *Machine learning*, 34(1):151–175.

Ratnaparkhi, A. et al. (1996). A maximum entropy model for Part-Of-speech tagging. In *Proceedings of the conference on empirical methods in natural language processing*, pages 133–142.

Rosenfeld, R. (1996). A maximum entropy approach to adaptive statistical language modelling. *Computer speech and language*, 10(3):187–228.

Stutsman, R., Grothoff, C., Atallah, M., and Grothoff, K. (2006). Lost in just the translation. In *Proceedings of the 2006 ACM symposium on Applied computing*, pages 338–345.

Taskiran, C. M., Topkara, M., and Delp, E. J. (2006). Attacks on linguistic steganography systems using text analysis. In *Proceedings of the SPIE Conference on Security, Steganography, and Watermarking of Multimedia Contents*, volume 6072, pages 97–105, San Jose, CA.

Topkara, M., Riccardi, G., Hakkani-Tür, D., and Atallah, M. (2006a). Natural language watermarking: Challenges in building a practical system. In *Proceedings of SPIE*, volume 6072, pages 106–117.

Topkara, M., Taskiran, C. M., and Delp, E. J. (2005). Natural language watermarking. In *Proceedings of the SPIE Conference on Security, Steganography, and Watermarking of Multimedia Contents*, volume 5681, pages 441–452, San Jose, CA.

Topkara, M., Topkara, U., and Atallah, M. J. (2006b). Words are not enough: sentence level natural language watermarking. In *Proceedings of the ACM Workshop on Content Protection and Security*, pages 37–46, Santa Barbara, CA.

Topkara, U., Topkara, M., and Atallah, M. J. (2006c). The hiding virtues of ambiguity: quantifiably resilient watermarking of natural language text through synonym substitutions. In *Proceedings of the 8th Workshop on Multimedia and Security*, pages 164–174, Geneva, Switzerland.

Velldal, E. (2008). *Empirical realization ranking*. Ph.D. thesis, University of Oslo, Department of Informatics.

Velldal, E. and Oepen, S. (2006). Statistical ranking in tactical generation. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, pages 517–525.

Vybornova, M. O. and Macq, B. (2007). A method of text watermarking using presuppositions. In *Proceedings of the SPIE Conference on Security, Steganography, and Watermarking of Multimedia Contents*, volume 6505, San Jose, CA.

Wan, S., Dras, M., Dale, R., and Paris, C. (2009). Improving grammaticality in statistical sentence generation: Introducing a dependency spanning tree algorithm with an argument satisfaction model. In *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics*, pages 852–860, Athens, Greece.

White, M. and Rajkumar, R. (2012). Minimal dependency length in realization ranking. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 244–255, Jeju Island, Korea.

Zhang, Y., Blackwood, G., and Clark, S. (2012). Syntax-based word ordering incorporating a large-scale language model. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 736–746, Avignon, France.

Zhang, Y. and Clark, S. (2011). Syntax-based grammaticality improvement using CCG and guided search. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 1147–1157, Edinburgh, Scotland, UK.

Joint Modeling for Chinese Event Extraction with Rich Linguistic Features

Chen CHEN Vincent NG
Human Language Technology Research Institute
University of Texas at Dallas
Richardson, TX 75083-0688, USA
{yzcchen,vince}@hlt.utdallas.edu

ABSTRACT
Compared to the amount of research that has been done on English event extraction, there exists relatively little work on Chinese event extraction. We seek to push the frontiers of supervised Chinese event extraction research by proposing two extension to Li et al.'s (2012) state-of-the-art event extraction system. First, we employ a joint modeling approach to event extraction, aiming to address the error propagation problem inherent in Li et al.'s pipeline system architecture. Second, we investigate a variety of rich knowledge sources for Chinese event extraction that encode knowledge ranging from the character level to the discourse level. Experimental results on the ACE 2005 dataset show that our joint-modeling, knowledge-rich approach significantly outperforms Li et al.'s approach.

TITLE AND ABSTRACT IN CHINESE

运用丰富语言学特征的中文事件抽取联合模型

与英文的事件抽取研究相比，对于中文的事件抽取研究工作相对较少。在 Li et al.(2012) 的基于监督学习的事件抽取系统基础上，我们提出了两个扩展以进一步推动中文事件抽取的研究。首先，我们使用了一个联合模型，以解决 Li et al. 管道式系统中的错误传播问题。其次，针对中文信息抽取，我们研究了一系列从字符层面到文章层面的特征。在 ACE2005 数据上的实验结果表明，我们运用丰富语言学特征的联合模型显著地优于 Li et al. 的方法。

KEYWORDS: event extraction, Chinese language processing.

KEYWORDS IN CHINESE: 信息抽取, 中文自然语言处理.

Proceedings of COLING 2012: Technical Papers, pages 529–544,
COLING 2012, Mumbai, December 2012.

1 Introduction

Recent years have seen a surge of interest in automatically extracting events from textual documents. While diverse types of event extraction have been examined in the literature, the one we will focus on in this paper is ACE event extraction, which involves extracting instances of a predefined event type from documents. For example, consider the following Chinese text segment:

Resneft 收购尤甘斯克付出了仅 93.5 亿美元
(Resneft acquired Yugansk, paying only 9.35 billion US dollars)

When applied to this example, an ACE event extraction system for Chinese should identify one event instance, which (1) is *triggered* by the verb 收购 [acquired] whose type is TRANSFER-MONEY, and (2) has three *arguments*, Resneft, 尤甘斯克 [Yugansk] and 93.5 亿美元 [9.35 billion], which fulfill the roles of BUYER, ARTIFACT and PRICE, respectively.

This example illustrates the four primary subtasks of an ACE event extraction system, namely, (1) *trigger identification* (e.g., 收购 [acquired] should be identified as the trigger of an event); (2) *trigger type determination* (e.g., 收购 [acquired] should be assigned the type TRANSFER-MONEY; (3) *argument identification* (e.g., Resneft, 尤甘斯克 [Yugansk] and 93.5 亿美元 [9.35 billion] are the arguments of this TRANSFER-MONEY event); and (4) *argument role determination* (e.g., Resneft, 尤甘斯克 [Yugansk] and 93.5 亿美元 [9.35 billion] play the roles of BUYER, ARTIFACT and PRICE respectively in this event).

Compared to the amount of research on English event extraction (e.g., Finkel et al. (2005), Grishman et al. (2005), Ahn (2006), Hardy et al. (2006), Maslennikov and Chua (2007), Ji and Grishman (2008), Patwardhan and Riloff (2009), Liao and Grishman (2010), Hong et al. (2011)), there is considerably less work on Chinese event extraction. Work on end-to-end Chinese event extraction was pioneered by Chen and Ji (2009b), who adopt a pipeline system architecture composed of four components that correspond to the four major subtasks mentioned above. More specifically, in *training*, they learn a classifier to perform each of the four subtasks independently using primarily lexico-syntactic features but also a couple of semantic features (see Section 3 for an overview of these features); and in *testing*, they feed a raw document through the pipeline of components where the output of one component is the input of the subsequent one. Li et al.'s Chinese event extraction system also employs a pipeline architecture, but aims to improve the first component, trigger identification, via two techniques, namely compositional semantics and discourse consistency. They show that with these two techniques, their system substantially outperforms Chen and Ji's system, achieving state-of-the-art results.

Our goal in this paper is to improve the state of the art in Chinese event extraction. Specifically, we take Li et al.'s event extraction system as a baseline, and investigate two extensions to their system. Our first extension is a machine learning extension where we employ *joint learning* for event extraction. As is commonly known in the natural language processing (NLP) community, a pipeline architecture, such as the one adopted by Chen and Ji and Li et al. , suffers from the *error propagation* problem, where the errors made by an upstream component will propagate to and could adversely affect the performance of a downstream component. We address this problem by recasting event extraction as two *joint learning* tasks where we (1) jointly learn trigger identification and trigger type determination and (2) jointly learn argument identification and argument role determination.[1]

[1] A natural alternative would be to jointly learn the four subtasks. Though possible, this would substantially increase the complexity of the learning task. In fact, our preliminary experiments indicate that this alternative yields inferior results to our way of applying joint learning to event extraction and therefore will not be pursued in this paper.

Our second extension is linguistic extension where we employ a *knowledge-rich* approach, investigating a variety of knowledge sources for Chinese event extraction. In this extension, not only do we propose more effective use of existing features such as character-based features, but we also investigate novel features that exploit results of zero pronoun resolution and noun phrase (NP) coreference resolution, as well as features that exploit trigger probability and trigger type consistency (see Section 3 for details). The strength of our linguistic extension stems in part from the richness in the variety of features it considers: these features capture linguistic information ranging from the character level to the discourse level, and exploit Chinese-specific phenomena such as the presence of zero pronouns.

We evaluate our approach on the ACE 2005 Chinese event extraction task, which involves identifying event instances that belong to one of 33 predefined event types. Unlike previous work (Chen and Ji, 2009; Li et al., 2012), which reserves only 10% of the annotated data for testing and uses the rest for training, we provide a more robust evaluation of our system via performing 10-fold cross-validation experiments. We discover that the F-scores achieved on different folds can vary by as many as 10 percentage points for all four subtasks. The sensitivity of the system performance can be attributed in part to the small size of the ACE 2005 dataset, which is composed of only 633 documents, suggesting that cross validation is needed to more accurately reveal the performance of an event extraction system when evaluated on this dataset. Overall, our experimental results demonstrate that our joint-learning, knowledge-rich approach substantially improves Li et al.'s system, suggesting that (1) joint learning offers benefits over the pipeline approach; (2) all but the coreference features improve performance; and (3) while each of our features provide small gains, their cumulative benefits are substantial.

The rest of the paper is organized as follows. Section 2 discusses related work. In Section 3, we provide an overview of our baseline Chinese event extraction system. Sections 4 and 5 describe our machine learning extension and our linguistic extension to the baseline system, respectively. We present evaluation results in Section 6 and conclude in Section 7.

2 Related Work

Much attention has been devoted to the event extraction task in the NLP community. In the early years, researchers focused on sentence-level extraction, employing local information from just one sentence (e.g., Grishman et al. (2005), Hardy et al. (2006), Ahn (2006)).

However, in many cases local information alone is insufficient to make the right decisions, so later work incorporates more context around a sentence and seeks high level information. For example, Gu and Cercone (2006) and Patwardhan and Riloff (2009) consider broader sentential context. Ji and Grishman (2008) extend the scope to a cluster of topic-related documents and utilize global information from related documents. Gupta and Ji (2009) employ cross-event information to extract implicit time information. Liao and Grishman (2010) leverage document-level cross-event inference; Liao and Grishman (2011a) extract topic features to improve event extraction; and Liao and Grishman (2011b) present a self-training strategy and combine it with global inference. McClosky et al. (2011) use the tree of event-argument relations in a reranking dependency parser to capture global event structure properties. Hong et al. (2011) explore entity type consistency to predict event mentions. Huang and Riloff (2012b) initially identify arguments and then include discourse properties to model textual cohesion. More recently, some researchers have tried to improve other aspects of event extractions. For example, Lu and Roth (2012) introduce a novel sequence labeling framework called structured preference modeling, and Huang and Riloff (2012a) propose a bootstrapping solution for argument extraction with little annotated data.

As far as work on Chinese event extraction is concerned, Chen and Ji (2009b) point out the Chinese-specific issue of word segmentation errors and create an errata table to alleviate this problem, analyzing the impactof different types of features. Chen and Ji (2009a) bootstrap Chinese event extraction with extra information from an English event extraction system using cross-lingual information projection. Ji (2009) extracts cross-lingual predicate clusters and uses a cross-lingual information extraction system to improve Chinese event extraction. Li et al. (2012) explore compositional semantics and discourse consistency to address the unknown trigger problem and word segmentation errors.

3 Baseline System

In order to establish a strong baseline Chinese event extraction system adopting the pipeline architecture mentioned in the introduction, we train a classifier for each of the four components by using a feature set that is the *union* of the features employed by Chen and Ji (2009b) and Li et al. (2012). We augment it with compositional semantics and discourse consistency, the two extensions proposed by Li et al. that aim to improve the trigger identification component, as described in detail in this section. Below we will discuss our implementation of each of the four components of the baseline Chinese event extraction pipeline. All classifiers are trained using the implementation of SVM available from the SVM^{multiclass} package[2]. Word segmentation, syntactic parsing, and dependency parsing are performed using Stanford's Chinese NLP and Speech Processing tool.[3].

3.1 Trigger Identification Component

Following Li et al. (2012), we employ a two-step approach to identify triggers. First, in the *extraction* step, we use heuristics to extract candidate triggers. Then, in the *pruning* step, we aim to improve precision by employing two types of pruning, namely *heuristic-based* pruning and *learning-based* pruning. Both steps are detailed below.

Extraction. To extract candidate triggers, we first follow Chen and Ji (2009b), positing a word in a test document as a candidate trigger if it appears in a training document as a (true) event trigger. Li et al. (2012) observe that this simple candidate extraction method has a low recall: it fails to extract many true triggers in a test document since many of them do not appear in the training set. To improve recall, they propose a technique to extract additional candidate antecedents based on *compositional semantics*.

The use of compositional semantics is motivated by the observation that the meaning of a Chinese word is largely determined by the meaning of its component characters. For instance, the meaning of 刺伤 [injure by stabbing] can be determined from the meaning of its component characters, 刺 [stab] and 伤 [injure]; similarly, the meaning of 撞伤 [injure by hitting] can be determined from the meaning of 撞 [hit] and 伤 [injure]. Now, assume that 撞伤 appears in a test document. If 撞伤 does not appear in the training data but 刺伤 appears as a trigger in the training data, we want to be able to infer that 撞伤 is a trigger from the fact that 刺伤 is a trigger since the two verbs both describe an "injure" event.

To be able to do this kind of inference for extracting additional candidate triggers, we employ a simple method proposed by Li et al.: (1) add all single-character verb triggers to a set (call it BV[4]); (2) split all other verb triggers in the training set into characters and add each character to BV; and

[2]http://svmlight.joachims.org/svm_multiclass.html
[3]http://nlp.stanford.edu/projects/chinese-nlp.shtml
[4]Li et al. (2012) refer to these verbs as "basic verbs" (BV).

(3) posit a word in a test document as a candidate trigger if it contains an element in BV. It should be easy to see that this method can easily handle cases such as the 撞伤 example discussed above.

Heuristic-based pruning. To prune spurious triggers from the list of candidate triggers, we employ the three heuristics proposed by Li et al.: non-trigger filtering, POS filtering, and verb structure filtering. We refer the reader to their paper for details of these heuristics.

Learning-based pruning. After heuristic-based pruning, we follow Li et al. and apply learning-based pruning to further prune the candidate triggers. Specifically, we train a classifier on the training data to determine whether a candidate is a trigger or not. Since Li et al. did not specify the training instance creation method, we experiment with several methods and found that creating one training instance from each word worked best. We train the classifier using 19 linguistic features, most of which were proposed by Chen and Ji, as shown below:

- Lexical features (6): trigger word; POS of trigger word; previous word + trigger word; previous POS + trigger POS; trigger word + next word; trigger POS + next POS
- Syntactic features (5): depth of trigger word in syntax parse tree; the path from leaf node of trigger to the root in syntax parse tree; the phrase structure expanded by the father of the trigger; phrase type of the trigger; the path from the leaf node of the trigger to the governing clause
- Semantic dictionaries (2): whether trigger word exists in a predicate list from Chinese PropBank (Xue and Palmer, 2008); the entry number of the trigger in a Chinese synonym dictionary[5]
- Nearest entity information (6): entity type of the syntactically/physically nearest entity to the trigger in syntax parse tree; entity type of the syntactically/physically left/right nearest entity to the trigger in syntax parse tree + entity

Reclassifying unconfidently labeled instances. Not all trigger candidates were classified with the same confidence by the trigger identifier. For our SVM-based trigger identifier, those instances that are closer to the hyperplane are classified with less confidence than those that are farther away. Li et al. propose to improve the accuracy of trigger identification by reclassifying those instances that were not confidently classified[6], specifically by training a *discourse consistency* (DC) classifier.

Before describing the DC classifier, let us motivate DC. Consider the sentence *The talks are serious*, where *talks* is a trigger of a MEET event whose arguments are not in the same sentence as the trigger itself. Because of the absence of nearest entity information (and hence the inability to compute the nearest entity information features), Li et al. observe that the corresponding test instances were typically classified with low confidence. To address this problem, they make an observation. Given a candidate trigger word t, if many other occurrences of t in the same discourse are being classified as a trigger, then t is likely to be a trigger due to DC. Similarly, if many other occurrences of t in the same discourse are being classified as non-triggers, then t is not likely to be a trigger due to DC.

Li et al. create five linguistic features that encode this observation (see their paper for details), train a DC classifier on a feature set composed of these five features as well as the 19 features used by the trigger identifier, and use it to reclassify those instances not confidently classified by the trigger identifier. Following Li et al., we train this DC classifier on the development set, which comprises 5% of the available training data reserved solely for the purpose of training this classifier.

[5]This dictionary is created by Harbin Institute of Technology's NLP Group and is available from http://ir.hit.edu.cn/phpwebsite/index.php?module=pagemaster&PAGE_user_op=view_page&PAGE_id=162. The entry number can be thought of as the equivalent of the synset id in English WordNet.

[6]Following Li et al., we posit that an instance is not confidently classified if the probability associated with its classification is between 0.05 and 0.95. We obtain these probability values by converting the signed distance values returned by the SVM using a sigmoid function.

3.2 Trigger Type Determination Component

Given a word identified as a trigger by the preceding component, the trigger type determination component employs a classifier to classify a word as belonging to one of the 33 predefined subtypes. Following Li et al., we train this classifier using the same 19 linguistic features that were used to train the trigger identifier.

3.3 Argument Identification Component

Given a typed trigger produced by the trigger type determination component, the argument identification component employs a classifier to determine whether a candidate argument is its actual argument or not. The list of candidate arguments for a trigger includes all and only those entity mentions, values and time expressions that appear in the same sentence as the trigger under consideration. Hence, training instances are created by pairing each trigger with each of its candidate arguments. The class value of a training instance is either YES (if the candidate is a true argument) or NO (if it is not). Following Li et al., we train this classifier using 19 linguistic features, as shown below.

- Basic features (6): trigger subtype; type of entity mention; head word of entity mention; event subtype + head word; event subtype + entity subtype; POS of trigger word
- Neighboring words (6): left/right neighbor word of entity; left/right neighbor word of the entity + word's POS; left/right neighbor word of the trigger + word's POS
- Syntactic features (4): the phrase structure expanding the parent of the trigger in the syntactic parse tree; whether the trigger is before or after the trigger; the minimal path from the entity to the trigger in the syntactic parse tree; the shortest length of the above minimal path
- Dependency feature (1): the dependency path from the entity to the trigger

3.4 Argument Type Determination Component

Given an entity mention identified as an argument of a trigger, the argument type determination component employs a classifier to determine its argument role. Following Li et al., we train this classifier using the same 19 linguistic features that were used to train the argument identifier.

We conclude this section with a note on feature computation. Following the setting of the ACE diagnostic tasks, we use ground truth entities, times and values in argument identification and argument type determination, but the rest of the features are all computed entirely automatically. This is also the setup adopted by Chen and Ji (2009b) and Li et al. (2012).

4 Machine learning Extension

In this section, we present our machine learning extension to the baseline system, which involves joint modeling of the event extraction subtasks. Unlike in pipeline modeling where we train four classifiers (i.e., one classifier per subtask), in joint modeling we train only two classifiers: the joint trigger classifier jointly performs trigger identification and trigger type determination, and the joint argument classifier jointly performs argument identification and argument role determination. Below we describe how these two joint models are trained.[7]

[7] Like us, Tan et al. (2008) also train two classifiers for event extraction, one related to labeling trigger type and the other argument type. So, at first glance, it seems that they are also performing some sort of joint learning for event extraction. However, there is an important difference between our goal and theirs: while we are performing end-to-end event extraction, they are *not*. Their first classifier, the trigger type labeler, is a multi-label *sentence* classifier that assigns to a sentence the

4.1 The Joint Trigger Classifier

To train the joint trigger classifier, we create one training instance for each word in the training set. If the word is not a trigger, the class label of the corresponding training instance is NONE. Otherwise, the class label is the type of the trigger. Each instance is represented by the same 19 features that were used to train the trigger identifier in the baseline system. As in the baseline, we employ SVM^{multiclass} to train this multiclass classifier.

After training, we apply the resulting SVM classifier to classify the test instances. Test instances are created via the same trigger candidate extraction process (including the use of compositional semantics) as described in the baseline system and are represented using the same 19 features as the training instances. If a test instance is assigned the class NONE by the classifier, the corresponding trigger candidate will be posited as a non-trigger. On the other hand, if the instance is classified as belonging to one of the 33 trigger types, the corresponding trigger is posited as a true trigger, and its type is the class value assigned by the classifier.

4.2 The Joint Argument Classifier

To train the joint argument classifier, we create a training instance by pairing each predicted trigger with each of its candidate arguments, where a candidate argument can be an entity mention, a value, or a time expression that appears in the same sentence as the predicted trigger. If the candidate argument is indeed a true argument of the trigger, the class label of the training instance is the argument's role. Otherwise, its class label is NONE. Each instance is represented by the same 19 features that were used to train the argument identifier in the baseline system. As in the baseline, we employ SVM^{multiclass} to train this multiclass classifier.

After training, we apply the resulting SVM classifier to classify the test instances. Test instances are created in the same way as the training instances. If a test instance is assigned the class NONE by the classifier, the corresponding argument candidate is classified as not an argument of the trigger under consideration. Otherwise, the argument candidate is indeed a true argument of the trigger, and its role is the class value assigned by the classifier.

5 Linguistic Extension

In this section, we describe our linguistic extension to the baseline system, where we introduce six groups of features for Chinese event extraction. Before we describe these features, there are two points that deserve mention. First, while this is a linguistic extension to the baseline system, there is nothing that prevents it from being applied to an event extraction system that employs joint learning. Second, solely for the sake of convenience, we follow the convention adopted in the baseline system, ensuring that (1) the trigger identifier and the trigger type labeler employ the same set of features, and (2) the argument identifier and the argument role labeler employ the same set of features. This implies that any group of features that we introduce below has to be used by either (1) both trigger-related classifiers; or (2) both argument-related classifiers, or (3) all four classifiers.

set of trigger types of the triggers it contains, but unlike ours, it does not explicitly identify triggers, even though it has some joint learning flavor in the sense that it allows the NULL class to be assigned to a sentence to indicate that it does not contain any triggers. Their second classifier, the argument type labeler, assumes that the input arguments of a trigger are correctly identified and simply performs argument labeling.

5.1 Character-Based Features

Recall that Li et al. (2012) have employed compositional semantics to improve the extraction of candidate triggers, enabling us to extract 撞伤 [injure by hitting], which appears in the test set but not the training set, as a candidate trigger if 刺伤 [injure by stabbing] is present in the training set as a true trigger, for instance.

Now, recall that each trigger candidate will be classified by the learned trigger identifier as either a true trigger or a non-trigger. Consider, for example, how the instance corresponding to 撞伤 will be classified. One of the linguistic features representing this instance is the word itself. But since 撞伤 never appears in the training set, the word feature is useless as far as classification is concerned. In other words, although 撞伤 and 刺伤 are similar verbs, the word feature does not capture such similarity, and therefore the classifier cannot exploit this similarity when classifying 撞伤.

To address the problem, we decompose a word into two units, putting the two units into separate bins. There are a four cases to consider: (1) if the word has two characters, we put the first character into the first bin and the second character into the second bin; (2) if the word has only one character, we put this character into both bins; (3) if the word has three characters and it can be segmented by a word segmenter, we put the resulting units into the two bins (for example, given the word 公开信 [open letter], 公开 [open] is placed in the first bin and 信 [letter] in the second); and (4) if the word has four characters or the word has three characters that cannot be segmented, we simply put the first two characters into the first bin and the remaining characters into the second bin.

Given these bins, we create four character-based features for the two trigger-related classifiers: The first two features come from the characters in the first and second bins respectively. The third and fourth features consist of the Harbin Institute of Technology NLP Group's synonym dictionary entry numbers for characters in the first and second bins respectively.

5.2 Semantic Role Labeling

It should be easy to see why semantic role labeling is useful for event extraction. First, a large portion of the triggers defined in event extraction task are predicates. Second, if a predicate happens to be a trigger, the predicate's arguments are essentially its event arguments. Furthermore, even though a semantic role labeler typically assigns PropBank-style roles (e.g., Arg0, Arg1) and the event argument roles are FrameNet-style roles, there is a close correspondence between the roles in these two styles.

Given the above observation, we hypothesize that semantic roles are useful for argument identification and argument role labeling. Consequently, we introduce four binary features for the argument classifiers: whether the trigger under consideration is a predicate according to the semantic role labeler, and whether the argument is the predicate's Arg0, Arg1, and time argument. We obtain semantic roles automatically using a publicly available semantic role labeling tool (Björkelund et al., 2009).

In addition, we hypothesize that semantic roles are also useful for trigger identification and trigger type classification. Our hypothesis stems in part from two observations: (1) many predicates identified by a semantic role labeler are triggers; and (2) knowing the entity types of the arguments identified by the semantic role labeler can help predict the type of the trigger. As a result, we propose to employ features that encode semantic role information for trigger identification and trigger type classification. Specifically, we introduce five features: whether the word under consideration is a predicate according to the semantic role labeler; if yes, the entity type and subtype of its Arg0;

the entity type and subtype of its Arg1.

5.3 Trigger Probability Feature

We define *trigger probability* of a word *w* as the probability that *w* appears as a true trigger in the training set. This probability is potentially useful for trigger identification: a word with a higher probability is more likely to be a true trigger.

We create a new feature for the two trigger-related classifiers whose value is the trigger probability of the word under consideration. If the word does not appear in the training set, we determine whether one of its "similar" verbs appears as a trigger in the training set.[8] If so, its trigger probability is that of its similar verb.[9] Otherwise, its trigger probability is set to zero.

5.4 Zero Pronoun Features

To motivate zero pronoun features, consider the following sentence:

国家主席江泽民今天晚上乘专机离开深圳前往文莱。
President Jiang Zemin took the plane tonight, left Shenzhen and went to Brunei.

The verb 乘 [took] has an overt subject 江泽民 [Jiang Zemin]. However, neither of two verbs 离开 [left] or 前往 [went] has an overt subject. Here, the two gaps before 离开 [left] and 前往 [went] are called zero pronouns. A zero pronoun has as its antecedent an entity mention that can fill the gap. In this example, 江泽民 [Jiang Zemin] is the entity mention that should be used to fill the gap: as we can see, 江泽民 [Jiang Zemin] is coreferent with those two zero pronouns.

Kim (2000) studied the difference of the usage of overt subject between English and Chinese. He found that the usage percent of overt subject in Chinese is only 64%, while for English the percent is more than 96%. Thus, zero pronoun is a prominent phenomenon in Chinese, and also appears frequently in the ACE 2005 dataset.

For event extraction, if there is a zero pronoun before a trigger, the entity mention to which this zero pronoun refers is likely to be an argument of this trigger. Thus, zero pronoun resolution, which involves (1) detecting zero pronouns and (2) finding their antecedents, is helpful for argument extraction. To our knowledge, results of zero pronoun resolution have not been exploited to improve event extraction.

In order to exploit zero pronoun resolution for use in event extraction, we need to build a zero pronoun resolver. In our experiments, we apply a rule-based method for zero pronoun resolution. Specifically, to detect zero pronouns, we employ a simple heuristic, which posits that a zero pronoun exists before a word if it is a verb and it has no overt subject in the corresponding syntactic parse tree. After detecting a zero pronoun, we resolve it in one of two ways:

Case 1: the verb following the zero pronoun is in a CP node and modifies a NP to its right (i.e., the

[8] We adopt Li et al.'s (2012) method for determining whether two Chinese verbs are similar. Specifically, we first analyze the structure of each of the verbs under consideration. According to Li et al.'s empirical observation, Chinese verbs possess one of six main structures, where BV is one of the elements of the set *BV* (as defined in Section 3): (1) BV (e.g., " 逮" [arrest]); (2) BV + verb (e.g., " 追杀" [chase to kill]); (3) verb + BV (e.g., " 躲进" [hide]); (4) BV + complementation (e.g., " 进了" [enter]); (5) BV + noun/adjective (e.g., " 开枪" [shoot]); (6) noun/adjective + BV (e.g., " 内战"[civil war]). If the two verbs have the same BV (basic verb) and the same structure, they are considered similar to each other.

[9] It is, of course, possible for a verb to have more than one similar verb. In this case, we compute its trigger probability based on one of its randomly selected similar verbs.

character " 的" appears between the verb and the NP), as shown in the following example, where 歼敌 [kills] modifies 胡修道 [Hu XiuDao]

歼敌 280 人的胡修道.
Hu XiuDao, who kills 280 enemies.

In this case, we resolve the zero pronoun to the NP modified by the verb, 胡修道 [Hu XiuDao].

Case 2 (default case): we resolve the zero pronoun to the nearest preceding NP that occupies the subject position and appears in the same sentence as the zero pronoun.

Next we create features that encode the output of the zero pronoun resolver for the two argument-related classifiers. Recall that each instance in the argument-related classifiers corresponds to a trigger and one of its candidate arguments. Keeping this in mind, the first feature we create encodes whether or not there is a zero pronoun before this trigger. If so, the second feature tells whether or not this argument is coreferent with the zero pronoun.

5.5 Trigger Type Consistency Features

All the features we have described thus far have focused on sentence-level extraction. However, document-level information also plays an important role in event extraction task. A good example is DC (Li et al., 2012). In this subsection, we propose another kind of document-level information, which we call trigger type consistency, to improve event extraction.

Trigger type consistency is motivated by one observation: documents in the ACE 2005 Chinese corpus are mostly news articles, each of which describes one theme, and most of the true triggers are compatible with this document theme. For example, if a document is about a fire accident, most of the annotated triggers in the gold standard are of type DIE. Therefore, knowing the document theme may help to identify triggers. We represent the theme of a document by the trigger type that occurs most frequently among the triggers in the document.[10] For example, if a document has 10 triggers and six of them have type DIE, then we use DIE to represent its theme. If a candidate trigger's type is the same as that of the majority of the triggers in the document, it is being *trigger-type-consistent* with the other triggers in the document and is more likely be a true trigger.

We create 33 features for the two trigger-related classifiers based on trigger type consistency. Each feature corresponds to one of the 33 predefined trigger types in the ACE 2005 event extraction task. We compute the feature values as follows. If, for example, one trigger has type DIE, then (1) the value of the feature corresponding to DIE is the probability that a trigger in this document has type DIE; and (2) the values of the remaining 32 trigger type consistency features are all zero.

A natural question is: since the type consistency features are to be used by the trigger-related classifiers, how is it possible that they are computed based on knowing which words are triggers and what their types are? The answer is that before computing the type consistency features, we run the baseline trigger identifier and the trigger type classifier to identify triggers and predict their types on each document.[11]

[10] For another way of computing the document theme, see Liao and Grishman (2011a).

[11] To identify triggers and predict their types on a test document, we train the two trigger-related baseline classifiers on the training set. On the other hand, to identify triggers and predict their types on a training document, we employ cross validation on the training set: we partition the training set into 9 folds, train the baseline classifiers on 8 folds, apply them to the documents in the remaining fold, and repeat this process 9 times so that we can obtain triggers and their types for each document in the training set.

5.6 Argument Consistency Feature

Another piece of document-level information we employ is argument consistency. It is based on the following observation: the true triggers typically correspond to events that are related to the main person or some major entities mentioned in the documents. Hence, if a candidate trigger has arguments that are coreferent with the arguments of true triggers, the candidate trigger will likely be a true trigger. Consider, for example, the following sentences:

[一家三口] 在昨天深夜集体喝下农药 [自杀].

[A family of three] drank pesticide to [suicide] last night.

[三个人] 总算是稳住了 [病情].

[Three people] finally stabilize the [patient's condition].

Since 自杀 [suicide] is a predicate that is frequently annotated as a trigger in the training data, it should be fairly easy for the learned trigger identifier to predict 自杀 as a trigger. On the other hand, it is difficult to predict 病情 [patient's condition] in the second sentence as a true trigger because many useful features, such as being a predicate or having many entities nearby, cannot be computed due to the lack of useful local information. However, if we know that (1) 病情 takes the argument 三个人 [Three people]; (2) 三个人 is coreferent with 一家三口 [A family of three]; and (3) 一家三口 is an argument of a true trigger 自杀, then we may be able to provide useful document-level information to make the classifier correctly classify the trigger candidate 病情 as a true trigger.

Based on the above observation, we create one feature that encodes this kind of document-level information, which we call argument consistency, for the trigger-related classifiers. The feature is the *role* of the argument that is coreferent with a predicted true trigger's argument.[12] Here is the reason behind using roles as feature values: some roles are more important than the others. Specifically, roles for arguments that serve as subjects or objects, such as VICTIM, are intuitively more important than roles of adjunct arguments, such as PLACE. Using the above two sentences as an example, the role of 三个人 [Three people] is VICTIM, so we will set the value of the feature corresponding to VICTIM as 1.

A natural question is: since the argument consistency feature is to be used by the argument-related classifiers, how is it possible that it is computed based on knowing which words are triggers and what their arguments and argument roles are? The answer is that before computing this feature, we run the baseline classifiers to identify triggers, predict their types, their arguments, and the argument roles on each document (see Footnote 11 for details on how to train these baseline classifiers).

6 Evaluation

Next, we evaluate our joint-learning, knowledge-rich approach to Chinese event extraction.

6.1 Experimental Setup

Dataset and evaluation methodology. All 633 Chinese documents in the ACE 2005 training corpus[13] are used in our evaluation. Unlike previous work (Chen and Ji, 2009b; Li et al., 2012) which designate 10% of the 633 documents as the test set, we perform 10-fold cross-validation experiments in order to obtain more accurate estimation of system performance. While we report

[12] As mentioned at the end of Section 3, since our evaluation setting follows that of the ACE diagnostic tasks, we compute our argument consistency feature based on gold coreference information.

[13] Note that the ACE 2005 test documents are not made publicly available.

results that are averaged over 10 folds, it is worth noting that the results achieved for each of the four subtasks on different folds vary considerably, sometimes by as many as 10 percentage points in F-score, due to the small size of the training and test sets. This suggests the importance of reporting cross-validation results when conducting experiments on the ACE 2005 corpus.

Evaluation measures. For each subtask, we report performance in terms of recall (R), precision (P), and F-score (F). These performance measures are computed based on the following definitions of correctness for the subtasks. For trigger identification, a trigger is correctly identified if its offsets exactly match a reference trigger. For trigger type determination, a trigger type is correctly determined if its trigger type and offsets exactly match a reference trigger. For argument identification, an argument is correctly identified if its offset, related trigger type and trigger's offsets exactly match a reference argument. Finally, for argument role determination, an argument role is correctly determined if its offsets, role, related trigger type and trigger's offsets exactly match a reference argument. Note that these definitions are also adopted by Chen and Ji (2009b) and Li et al. (2012).

6.2 Feature Selection

To determine which of the feature groups described in Section 5 are useful when used in combination with the baseline features in Section 3, we conduct feature selection experiments to identify the best feature subset. There are seven feature groups to be considered in our feature selection experiments: (G1) discourse consistency features (Li et al., 2012); (G2) semantic role labeling features; (G3) trigger probability features; (G4) character-based features; (G5) the argument consistency feature; (G6) trigger type consistency features; and (G7) zero pronoun features. Two points deserve mention. First, among these seven feature groups, G1, G2, G3, G4, G5 and G6 are used for training the trigger-related classifiers whereas G2 and G7 are used for training the argument-related classifiers. Second, while G1 is not a feature group proposed by us, we consider it in our feature selection experiments. The reason is that some of our feature groups (e.g., G5 and G6) also capture document-level information like G1, and because they overlap in terms of the information they capture, we may be better off not retaining all of them.

Feature selection is done using cross validation on the training documents. Specifically, we partition the training documents into 9 folds, train the classifier whose features are to be selected on 8 folds, apply the classifier to the remaining fold, and repeat this process 9 times in order to select the feature groups that have the best average performance over the 9 folds when used in combination with the baseline features.

As far as the feature selection algorithm is concerned, we employ backward elimination. It starts with the full feature set (containing the 7 feature groups to be selected plus the baseline features), and removes in each iteration the feature group whose removal yields the best system performance. We run the algorithm until all but the baseline features are removed, and identify the feature subset that achieves the best performance during the feature selection process.

Note that feature selection is performed separately for each of the four classifiers used in the pipeline approach and each of the two classifiers used in the joint learning approach.

Table 1 shows the feature groups selected for each classifier. Let us first consider the classifiers for the pipeline approach. For trigger identification, all feature groups are retained. For trigger type identification, only G4 (character-based features) are retained. This makes sense because other feature groups are designed only to help discriminate true triggers from wrong triggers. Finally,

G2 (semantic role labeling) and G7 (zero pronoun features) prove to be effective for argument identification but not argument role labeling. This means that the best result for argument role labeling is achieved by training the classifier on only the baseline features.

Approach	Classifier	Selected Features
Pipeline	Trigger Identification	G1, G2, G3, G4, G5, G6
	Trigger Type Determination	G4
	Argument Identification	G2, G7
	Argument Role Determination	---
Joint	Trigger Component	G2, G3, G4, G6
	Argument Component	G2, G7

Table 1: Feature selection results

Let us turn to the two classifiers in the joint approach. Interestingly, for the trigger classifier, G1 (discourse consistency) and G5 (argument consistency) are removed. This result provides suggestive evidence that G1 and G5 serve overlapping purposes with the rest of the feature groups we proposed and therefore not all of them need to be retained to achieve the best performance. For the argument classifier, both G2 (semantic role labeling) and G7 (zero pronoun features) are retained. This is consistent with the pipeline results, where both feature groups are shown to be useful for argument identification.

6.3 Test Set Results

Using the features selected for each classifier, we obtain test set results. The average 10-fold cross-validation results for the pipeline approach and the joint approach are shown in Tables 2 and 3, respectively. In each case, we compare our approach against two baselines, one where the classifiers are trained using the baseline feature set without the discourse consistency features, and one where the classifiers are trained using the baseline feature set including the DC features. This setup enables us to better evaluate the usefulness of the DC features: since our feature selection experiments indicate that the DC features are not always useful when used in combination with our proposed features, we want to examine whether they are always useful when used in combination with the baseline features.

	Trigger Identification			Trigger Type Determination			Argument Identification			Argument Role Determination		
Feature Set	R	P	F	R	P	F	R	P	F	R	P	F
Baseline features without DC	50.6	75.5	60.6	47.5	70.8	56.8	35.1	52.3	42.0	31.2	46.5	37.4
Baseline features with DC	55.6	72.7	63.0	52.0	67.9	58.9	38.9	50.2	43.8	34.8	45.0	39.2
Our selected features	60.5	70.1	**64.9**	56.6	65.6	**60.8**	43.8	50.2	**46.8**	39.3	45.1	**42.0**

Table 2: Pipeline modeling results on the test set.

	Trigger Identification			Trigger Type Determination			Argument Identification			Argument Role Determination		
Feature Set	R	P	F	R	P	F	R	P	F	R	P	F
Baseline features without DC	50.0	77.0	60.7	47.5	73.1	57.6	34.1	58.7	43.2	30.4	52.3	38.5
Baseline features with DC	55.3	75.6	63.9	52.6	71.8	60.7	38.2	57.4	45.9	34.3	51.5	41.1
Our selected features	62.2	71.9	**66.7**	58.9	68.1	**63.2**	43.6	57.3	**49.5**	39.2	51.6	**44.6**

Table 3: Joint modeling results on the test set.

A few points about these results deserve mention. First, the DC features offer benefits when used

in combination with the baseline features in both the pipeline and joint approaches. Second, we can see that joint modeling always offers benefits over pipeline modeling when we consider comparable rows in the two tables. In fact, using the selected features, the joint approach performs significantly better than the pipeline approach on all four subtasks (paired t-test; $p < 0.05$). Finally, our best-performing system (row 3 of Table 3) significantly outperforms the approach adopted by state-of-the-art event extraction systems (row 2 of Table 2) on all four subtasks (paired t-test, $p < 0.05$): F-score increases by 3.7% for trigger identification, by 4.3% for trigger type determination, by 5.7% for argument identification, and by 5.4% for argument role labeling.

6.4 Feature Analysis

To gain better insight into the contribution of each feature group to our best-performing system (row 3 of Table 3), we add each feature group incrementally to the baseline feature set. Results are shown in Table 4. Recall that the DC features were not selected by the feature selection algorithm for our best-performing system, so none of the results in this table involves DC features. In particular, we start with the baseline features without any DC features (row 1), and add the feature groups incrementally to this baseline feature set.

	Trigger Identification			Trigger Type Determination			Argument Identification			Argument Role Determination		
Features	R	P	F	R	P	F	R	P	F	R	P	F
Baseline features without DC	50.0	77.0	60.7	47.5	73.1	57.6	34.1	58.7	43.2	30.4	52.3	38.5
+G2 (Semantic role labeling)	52.1	77.7	62.4	49.8	74.4	59.7	36.9	61.7	46.2	33.2	55.4	41.5
+G3 (Trigger probability)	56.0	75.3	64.3	53.3	71.5	61.1	39.2	59.7	47.3	35.2	53.7	42.5
+G4 (Character features)	59.8	73.8	66.1	56.6	69.6	62.6	41.2	57.9	48.2	37.2	52.3	43.5
+G6 (Trigger type consistency)	62.2	71.9	66.7	58.9	68.1	63.2	42.7	56.5	48.6	38.5	50.9	43.8
+G7 (Zero pronouns)	62.2	71.9	66.7	58.9	68.1	63.2	43.6	57.3	49.5	39.2	51.6	44.6

Table 4: Results of incremental addition of features to the joint model on the test set

As we can see, except for the argument consistency feature, all feature groups provide gains when added to the feature set in an incremental fashion. For example, in each of the four subtasks, adding semantic role labeling improves F-score by 1.7%, 2.1%, 3.0%, and 3.0%, respectively. Adding trigger probability next improves F-score by 1.9%, 1.4%, 1.1% and 1.0%. After that, adding character features improves F-score by 1.8%, 1.5%, 0.9% and 1.0%. Note that the zero pronoun features are only designed to improve the two argument-related subtasks. So, with the addition of these zero pronoun features, the two trigger-related subtasks are unaffected, while argument identification and argument role determination are improved by 0.9% and 0.8% in F-score, respectively.

Conclusion and Perspectives

We proposed a joint-learning, knowledge-rich approach to a relatively under-studied yet important task, Chinese event extraction, aiming to extend Li et al.'s (2012) state-of-the-art Chinese event extraction system. Linguistically, not only did we propose more effective use of existing features such as character-based features, but we also investigated novel features that exploit results of zero pronoun resolution and noun phrase coreference resolution, as well as those that exploit trigger probability and trigger type consistency. In 10-fold cross-validation experiments on the ACE 2005 dataset, we showed that our system outperformed Li et al.'s system by 3.7−5.7% on the four event extraction subtasks. Our results also indicated that all but the argument consistency feature contributed positively to overall performance. In particular, while each of these feature groups provided small gains, their cumulative benefits were substantial.

Acknowledgments

We thank the three anonymous reviewers for their invaluable comments on an earlier draft of the paper. This work was supported in part by NSF Grant IIS-1147644.

References

Ahn, D. (2006). The stages of event extraction. In *Proceedings of the COLING/ACL 2006 Workshop on Annotating and Reasoning about Time and Events*, pages 1−8, Sydney, Australia.

Björkelund, A., Hafdell, L., and Nugues, P. (2009). Multilingual semantic role labeling. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning: Shared Task*, pages 43−48.

Chen, Z. and Ji, H. (2009a). Can one language bootstrap the other: A case study on event extraction. In *Proceedings of the NAACL HLT 2009 Workshop on Semi-supervised Learning for Natural Language Processing*, pages 66−74.

Chen, Z. and Ji, H. (2009b). Language specific issue and feature exploration in chinese event extraction. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics, Companion Volume: Short Papers*, pages 209−212.

Finkel, J. R., Grenager, T., and Manning, C. (2005). Incorporating non-local information into information extraction systems by gibbs sampling. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics*, pages 363−370.

Grishman, R., Westbrook, D., and Meyers, A. (2005). NYU's English ACE 2005 system description. Technical report, Department of Computer Science, New York University.

Gu, Z. and Cercone, N. (2006). Segment-based hidden markov models for information extraction. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th Annual Meeting of the Association for Computational Linguistics*, pages 481−488.

Gupta, P. and Ji, H. (2009). Predicting unknown time arguments based on cross-event propagation. In *Proceedings of the ACL-IJCNLP 2009 Conference Short Papers*, pages 369−372.

Hardy, H., Kanchakouskaya, V., and Strzalkowski, T. (2006). Automatic event classification using surface text features. In *Proceedings of the AAAI 2006 Workshop on Event Extraction and Synthesis*, pages 36−41.

Hong, Y., Zhang, J., Ma, B., Yao, J., Zhou, G., and Zhu, Q. (2011). Using cross-entity inference to improve event extraction. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 1127−1136.

Huang, R. and Riloff, E. (2012a). Bootstrapped training of event extraction classifiers. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 286−295.

Huang, R. and Riloff, E. (2012b). Modeling textual cohesion for event extraction. In *Proceedings of the 26th AAAI Conference on Artificial Intelligence*.

Ji, H. (2009). Cross-lingual predicate cluster acquisition to improve bilingual event extraction by inductive learning. In *Proceedings of the NAACL HLT 2009 Workshop on Unsupervised and Minimally Supervised Learning of Lexical Semantics*, pages 27−35.

Ji, H. and Grishman, R. (2008). Refining event extraction through cross-document inference. In *Proceedings of ACL-08: HLT*, pages 254−262.

Kim, Y.-J. (2000). Subject/object drop in the acquisition of korean: A cross-linguistic comparison. *Journal of East Asian Linguistics*, 9:325−351.

Li, P., Zhou, G., Zhu, Q., and Hou, L. (2012). Employing compositional semantics and discourse consistency in chinese event extraction. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 1006−1016.

Liao, S. and Grishman, R. (2010). Using document level cross-event inference to improve event extraction. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 789−797.

Liao, S. and Grishman, R. (2011a). Acquiring topic features to improve event extraction: in pre-selected and balanced collections. In *Recent Advances in Natural Language Processing*, pages 9−16.

Liao, S. and Grishman, R. (2011b). Can document selection help semi-supervised learning? a case study on event extraction. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 260−265.

Lu, W. and Roth, D. (2012). Automatic event extraction with structured preference modeling. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics*, pages 835−844.

Maslennikov, M. and Chua, T.-S. (2007). A multi-resolution framework for information extraction from free text. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 592−599.

McClosky, D., Surdeanu, M., and Manning, C. (2011). Event extraction as dependency parsing. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 1626−1635.

Patwardhan, S. and Riloff, E. (2009). A unified model of phrasal and sentential evidence for information extraction. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 151−160.

Tan, H., Zhao, T., and Zheng, J. (2008). Identification of chinese event and their argument roles. In *Proceedings of the 2008 IEEE 8th International Conference on Computer and Information Technology Workshops*, CITWORKSHOPS '08, pages 14−19, Washington, DC, USA. IEEE Computer Society.

A Simplification-Translation-Restoration Framework for Cross-Domain SMT Applications

Han-Bin Chen[1], Hen-Hsen Huang[1], Hsin-Hsi Chen[1], and Ching-Ting Tan[2]
(1) Department of Computer Science and Information Engineering,
National Taiwan University, Taipei, Taiwan
(2) National Taiwan University Hospital, Taipei, Taiwan
{hbchen, hhhuang}@nlg.csie.ntu.edu.tw,
{hhchen, tanct5222}@ntu.edu.tw

ABSTRACT

Integration of domain specific knowledge into a general purpose statistical machine translation (SMT) system poses challenges due to insufficient bilingual corpora. In this paper we propose a **s**implification-**t**ranslation-**r**estoration (*STR*) framework for domain adaptation in SMT by simplifying domain specific segments of a text. For an in-domain text, we identify the critical segments and modify them to alleviate the data sparseness problem in the out-domain SMT system. After we receive the translation result, these critical segments are then restored according to the provided in-domain knowledge. We conduct experiments on an English-to-Chinese translation task in the medical domain and evaluate each step of the *STR* framework. The translation results show significant improvement of our approach over the out-domain and the naïve in-domain SMT systems.

用於跨領域統計式機器翻譯系統之簡化-翻譯-還原架構

摘要

因為雙語語料的不足，將特定領域知識整合到一般用途的統計式機器翻譯(SMT)系統具有相當挑戰性。在本篇論文中，我們提出一個簡化-翻譯-還原(*STR*)的架構，藉由簡化特定領域的片段來達成SMT的領域調適。對於一篇領域內的文字，我們首先辨識其中重要的片段再做修改以減輕領域外SMT系統的資料稀疏問題。我們取得翻譯結果後，根據提供的領域內知識將這些重要片段還原。最後我們進行了醫療領域的英中翻譯的實驗，並且評估*STR*架構內的每一步驟。翻譯結果顯示我們的方法顯著地優於領域外的SMT系統，以及簡易型的領域內SMT系統。

KEYWORDS : Cross-Domain SMT, Domain Adaptation, Statistical Machine Translation
關鍵詞 : 跨領域統計式機器翻譯, 領域調適, 統計式機器翻譯

Proceedings of COLING 2012: Technical Papers, pages 545–560,
COLING 2012, Mumbai, December 2012.

1 Introduction

Over the past decades, the rapid growth of available parallel corpus makes SMT development feasible, and SMT system has gradually moved toward practical use because of its relatively acceptable translation speed and quality. A phrase-based SMT system (Koehn et al., 2003; Koehn, 2004), for example, trains a phrase table from a large bilingual corpus as its translation model, and decodes source language input in polynomial time with greedy algorithms such as beam search. It translates phrases as basic units, and thus captures short-range reordering phenomena between source and target languages. Generally phrase-based SMT models outperform word-based ones (Koehn et al., 2003). However, an SMT system fails to capture long-range contextual knowledge due to the limited horizon and the sparseness nature of lexical n-grams. These drawbacks reduce the translation quality in terms of translation and reordering errors, especially when limited bilingual corpus is available for estimating translation model.

The data sparseness problem may worsen when we build an SMT system for a specific domain but have small or no bilingual in-domain corpus. The translation performance could be seriously degraded since the SMT system cannot gather the statistical evidence of a segment containing out-of-vocabulary (OOV) words. For some domains, a bilingual dictionary containing source-target term pairs is available. With such in-domain knowledge, one can force an SMT system to translate OOV terms according to the dictionary. In this way, we correctly translate in-domain terms. However, translation quality may still be unsatisfying under this naïve integration because in-domain terms are rare or unseen in the background SMT model. Hence the context of these in-domain terms can hardly be captured. FIGURE 1 shows an incorrect English-Chinese translation of a sentence in a medical record by the online Google Translate service. In this example, it correctly translates the diagnosis term "crystal induced arthritis". However, it mistranslates its nearby phrasal verb "suffered from" by translating these two words separately. This example shows an SMT system may recognize the domain specific terms with either bilingual dictionary or its background SMT model, but still gives improper translations of the surrounding words due to insufficient context knowledge of these in-domain terms.

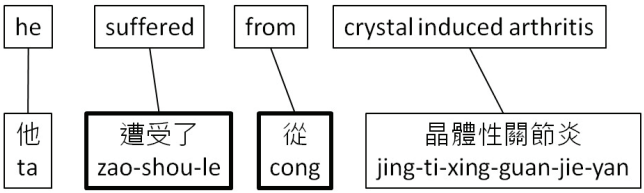


FIGURE 1 – Translating domain specific text with Google Translate. The bold target phrases are the inappropriate translations.

Our work originates from the following idea: modifying an in-domain segment in a source text such that the background SMT model not only recognizes it but also translates it together with its context words. Based on the motivation, for example, we modify the complicated diagnosis term "crystal induced arthritis" in FIGURE 1 into a more general term that occurs more frequently in the background SMT model, such as "cancer", "pneumonia" or "hypertension". Since these terms are more common in the general corpora, the general SMT system is able to better handle the modified text. Take the common word "cancer" as an example, the Google Search engine shows

that Web pages containing "suffered from cancer" significantly outnumber those containing "suffered from crystal induced arthritis".

In this paper, we propose a **s**implification-**t**ranslation-**r**estoration (*STR*) framework to address domain adaptation in SMT by modifying in-domain text in favour of the background SMT system. The *STR* framework includes four steps to produce a cross-domain translation with higher quality than a general purpose SMT system. More specifically, to translate an in-domain source text, we identify the domain-specific segments and simplify them into more general expressions. The modified text is then translated by an out-domain SMT system. After that, we manipulate the translation result by replacing the modified parts and their translations with the correct bilingual segments in our in-domain knowledge (e.g., bilingual dictionary). Our framework is suited to the cross-domain SMT scenario where an out-domain SMT system and bilingual in-domain dictionaries are available. We show the effectiveness of the framework through a case study by building an SMT system for the medical domain. In this domain, OOV is a frequent problem if a general purpose SMT system is applied. On the other hand, there are plenty of bilingual dictionaries in this area. We implement the *STR* framework and report the experimental results of the translation tasks on medical summaries in a hospital. The central issues in this framework include (1) collecting bilingual in-domain knowledge and identifying in-domain segments in a source text, (2) replacing the in-domain segments with the proper simplified forms, (3) translating the modified text with a background SMT system and (4) restoring the original in-domain segments after receiving the translation results from the background SMT system.

We are not the first to rephrase source language text in order to improve SMT output. As a pilot study, Resnik et al. (2010) proposed a targeted paraphrasing approach which identifies the critical source segments difficult for the background SMT system to translate. These segments are then manually paraphrased in many ways in order to provide the SMT system with more choices of decoding paths. Different from their work, we automatically identify the critical segments with in-domain knowledge, simplify them with linguistic information, and restore these critical segments after receiving the SMT results.

This paper is organized as follows. In Section 2, we review the previous works on domain adaptation and related work of our approach in SMT. In Section 3, we formally describe our simplification-translation-restoration framework to deal with domain adaptation in SMT. In Section 4, we evaluate the effectiveness of our *STR* framework by conducting a case study of English-Chinese medical summary translation, and discuss the experimental results. Section 5 draws the conclusion, indicates the potentials of our method, and shows some future work.

2 Related Work

Building an SMT system from large scale bilingual data for a specific application has become a practical option today. On the other hand, SMT model heavily relies on the statistical evidences in the training corpus. As a result, it may learn a biased SMT model, and suffer from the data sparseness problem of the training corpus when dealing with the ambiguity nature of human language. This drawback results in some typical issues such as translation disambiguation problem (Carpuat et al., 2007; Chan et al., 2007), in which a word has several senses, but the corpus is biased towards a particular subset of the senses. The SMT model trained from such a corpus is therefore prone to give the wrong translation due to the wrong choice of sense.

When a cross-domain SMT application is concerned, data sparseness problem is worsened by limited in-domain bilingual corpus. Domain adaptation techniques therefore play a key role in building an in-domain SMT system under a resource poor environment. A number of adaptation approaches have been proposed by leveraging either bilingual or monolingual in-domain resources. Foster and Kuhn (2007) proposed a mixture-model approach that divides and trains a bilingual corpus into several models. Different models were then weighted by estimating the similarity between a model and the in-domain development data. Matsoukas et al. (2009) devised sentence-level features and weighted the domain relevance to each sentence in the bilingual training corpus by optimizing an objective function. Foster et al. (2010) further raised the granularity by weighting at the level of phrase pairs. Similarly, a mixture-model approach was also applied in word-alignment task (Civera and Juan, 2007). Zhao et al. (2004) applied information retrieval techniques to select in-domain documents from large monolingual text collections and enhanced the baseline language model. Bertoldi and Federico (2009) exploited an in-domain monolingual corpus to synthesize a pseudo bilingual corpus and trained an in-domain translation model from the synthesized corpus.

While previous works concentrated on model and parameter estimation to achieve domain adaptation, they worked on the data sets with similar lexicons. Few studies dealt with large domain gap, which is a practical issue for a cross-domain SMT system. In the medical domain, for example, a term may not even appear in training corpus and therefore SMT system gives no translation to it. This OOV problem is common when translating domain specific terms such as diagnosis and surgical names in biomedical literature or medical records using a general purpose SMT system. Different from the previous works, we address cross-domain issues in SMT across two largely distinct domains by simplifying in-domain segments to the ones that can be recognized by the out-domain or the background SMT system, and restoring the in-domain segments after receiving the SMT results.

Text simplification (Zhu et al., 2010; Woodsend and Lapata, 2011; Wubben et al., 2012) itself has some straightforward NLP applications. For example, we produce a simpler version of a text by modifying the lexical contents and shortening the grammatical structures without changing the original text at the semantic level. Such simplified contents are beneficial for language learners and people with lower levels of literacy. One of the real world applications is Simple English Wikipedia (http://simple.wikipedia.org), which uses simple English words and grammar, and thus English language learners can benefit from it. In this paper we apply sentence simplification techniques to improve machine translation quality. The source language input is simplified into the version that makes it easier for the SMT system to translate. This simplification step serves as a pre-processing module of the out-domain SMT model which has poor in-domain knowledge.

The simplification approach can be viewed as a variant of paraphrasing, which expresses the same meaning in different ways. Paraphrasing is employed for various NLP tasks, such as machine translation, natural language generation and computer assisted language learning. For SMT task, paraphrasing is often used to alleviate the data sparseness problem in translation model. For example, we paraphrase a source language text so that the paraphrased parts are easier for the background SMT system to translate. Callison-Burch et al. (2006) pioneered a pivoting approach through parallel corpus to improve phrase-based SMT model. Marton et al. (2009) proposed a monolingual framework to select paraphrases of a term by comparing its context with those of candidate paraphrases. Aziz et al. (2010) proposed a semi-automatic approach to mine paraphrases from hypernyms and hyponyms in ontology. Resnik et al. (2010)

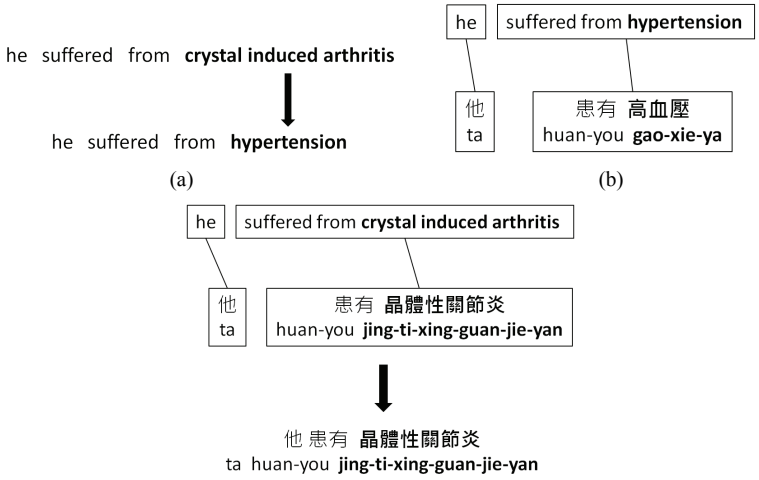


FIGURE 2 – The idea behind our *STR* framework applied to the bold phrases. We (a) simplify the source diagnosis term before translation, (b) translate the simplified sentence with an SMT system, and (c) restore the original diagnosis term and produce the translation result.

conducted a pilot study of targeted paraphrasing in which monolingual speakers on both sides collaborate to improve SMT output by paraphrasing the critical segments of source text.

Different from previous studies that applied paraphrasing to SMT in the general domain, our work focuses on addressing cross-domain issues. We aim at adapting in-domain knowledge into the out-domain SMT system in a more smooth fashion than the naïve integration approach. While other works paraphrase general segments to provide more decoding options, we concentrate on domain specific segments which account for the performance degradation of a cross-domain SMT system. We try to identify these in-domain segments and simplify them to better fit the background out-domain SMT model.

3 A Simplification-Translation-Restoration (*STR*) Framework

We express the basic idea behind our *STR* framework through the example of the incorrect translation result shown in FIGURE 1. In order to fit the in-domain term to its general context, we change the obscure medical term to a more general one. In this example, the rare source medical term "crystal induced arthritis" in FIGURE 1 is thus simplified to the more public diagnosis term "hypertension", as illustrated in FIGURE 2(a). This simplified sentence is then sent to Google Translate. FIGURE 2(b) shows the translation result, which gives not only the correct translation of the diagnosis term "高血壓", but also the nearby context "患有". Moreover, the phrase "suffered from hypertension" is translated as a unit, implying that the translation model capture the context distribution of the simple diagnosis term "hypertension". Finally, to acquire the actual translation rather than the simplified one, the simplified parts are then cut out and the

original bilingual in-domain terms are restored as illustrated in FIGURE 2(c). The final translation result is correct and fluent in contrast with the one in FIGURE 1.

The proposed *STR* framework is composed of four steps as follows.

1. Identifying in-domain segments s_1, s_2, …, s_n from an input sentence S.
2. Simplifying s_1, s_2, …, s_n in S and deriving a new source sentence S'.
3. Translating the source sentence S' into a target sentence T'.
4. Restoring the bilingual in-domain segments s_1-t_1, s_2-t_2, …, s_n-t_n back to S'-T' and deriving the final translation result T.

The following subsections describe each of them in detail.

3.1 Identification

An SMT system performs worse when translating segments which are rare in the background model. Therefore, in the first step of our framework, we identify the in-domain segments in an input source text for the next simplification step. To this end, we collect bilingual in-domain resources which include source-target string pairs.

Although a parallel corpus may not be available in a special domain, there are various ways to collect bilingual in-domain knowledge. For example, bilingual dictionaries can be found in the specific areas with long histories, such as medicine, physics and economics. They provide in-domain terminology with high quality and less noise. Not limited to the hand-crafted dictionaries, the bilingual in-domain knowledge may include phrase pairs or synchronous grammar rules, depending on the translation model and the decoding style of our background SMT system. Such bilingual knowledge can be collected by using automatic approaches (Wu and Chang, 2007; Haghighi et al., 2008) or semi-automatic approaches (Morin et al., 2007; Chen et al., 2011).

3.2 Simplification

In the simplification step, the identified in-domain segments of a text are transformed into the more general expressions. The modified text is then ready to be translated by the background SMT system. The simplification step serves as a pre-processing step before translation. We simplify an in-domain segment according to its type – say, **terminological unit** and **syntactic unit**, in this study.

Terminological units refer to terms that appear in domain specific dictionaries and glossaries without specifiers and modifiers. For these in-domain terms, we simplify them by finding the related terms such as hypernyms or synonyms which have relatively more occurrences and contextual information in the background SMT model. Syntactic units are linguistically meaningful segments which constitute special writing styles of a domain. These units usually bear syntactic categories at clausal or phrase levels such as S, NP, VP, PP, etc. They contain heads along with their modifiers. These syntactic categories can be derived from the parsed or the chunked results. We simplify a syntactic unit based on the rule corresponding to its syntactic category shown as follows.

(a) NP (Noun Phrase)

We keep the head of an NP and remove its specifier and modifier. If the head noun is a domain specific term, then it is further treated by the simplification rule to a terminological unit.

FIGURE 3 shows a parsing tree of a string as an example. The string is labelled as NP at the root node which contains two sub-trees with categories NP and PP, respectively. According to this simplification rule, we therefore remove its PP modifier. As a result, the string "a patient of skin rash with multiple erythematous papules" is simplified to only its head "a patient".

(b) VP (Verb Phrase)

VP → V + NP: We keep V untouched and simplify NP according to the simplification rule (a). For example, we simplify "had underlying diseases of ventricular tachycardia and dyslipidemia" to "had diseases".

VP → V + PP: We keep V untouched and remove PP if PP is a modifier. If PP is mandatory, it is further simplified based on the simplification rule (c). Whether PP is optional or mandatory is determined by the subcategory of V. For example, we simplify the sentence "he was discharged on the morning of 6/30" to "he was discharged".

(c) PP (Prepositional Phrase)

PP → P + NP: We keep P and simplify NP according to the simplification rule (a). For example, we simplify "with underlying diseases of ventricular tachycardia and dyslipidemia" to "with diseases".

(d) S (Clause)

We simplify a clause by simplifying its children recursively according to the above simplification rules.

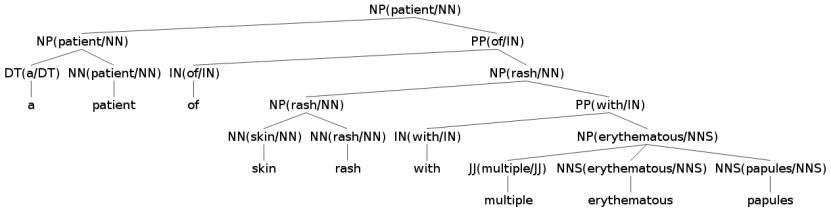


FIGURE 3 – A parse tree of a syntactic unit. A bracketed string at each non-terminal node is the head of the corresponding syntactic category.

The rule-based simplification approach is straightforward, but can be effectively applied to most of the syntactic units as discussed in Section 4. Applying transformation or rewriting rules on source sentences based on their syntactic structures has been adopted in other works. Wang et al. (2007) listed a set of prominent syntactic reordering rules that systematically describe the word order difference between the source and target languages. Based on these rules, they parsed a source language input and reordered its structure to match the target language grammar for training a better translation model and improving a phrase-based SMT system. In their work, source side syntactic reordering also serves as a pre-processing module of the SMT system. Different from their work, we simplify a source language input in favour of the background SMT system instead of changing the order of its structure.

3.3 Translation

In the translation step, the background SMT system translates the simplified in-domain text and produces its translation result. Since the input source text is simplified in favour of the translation system, the contextual distributions of the phrases can be estimated better than those of the original text, as demonstrated in FIGURE 2(b).

Our *STR* framework performs domain adaptation under the scenarios where bilingual in-domain segments are available. It is possible to be combined with other domain adaptation approaches that exploit monolingual or bilingual in-domain corpus to help further improve the translation quality. For example, if a parallel in-domain corpus is available, we can perform learning-based domain adaptation approaches described in Section 2, and tune the background translation model toward the specific domain. In this way, we may receive better translation results from the background SMT system and facilitate the next restoration step.

For a phrase-based SMT system and its variations (Chiang, 2005; Xiong et al., 2006; Huang and Chiang, 2007), we can further customize the decoder to produce the translation with higher quality and facilitate the next restoration step of our framework. For example, the in-domain segments in a text are either terminological or syntactic units in our experiments, and therefore their translations are continuous without other interleaving translations. However, an out-domain SMT system may give wrong ordering without in-domain knowledge. For instance, the translation of the medical term "bone lesion" is separated as illustrated in FIGURE 4. In our work, we set up a Moses (Koehn et al., 2007) SMT system and apply its advanced feature of specifying reordering constraint to each of the simplified phrases. Under the constraint, a simplified phrase is translated as a block and its translation is continuous on the target side.

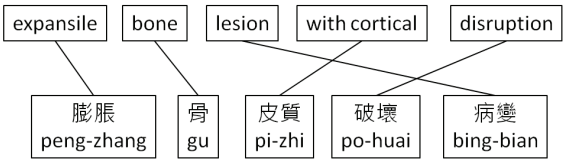


FIGURE 4 – The incorrect reordering of the in-domain segments "bone lesion".

3.4 Restoration

In the restoration step, we receive the translation result from the background SMT system and perform post-processing steps. We locate the simplified phrase pairs and replace them with their corresponding bilingual in-domain segments as illustrated in FIGURE 2(c). The resulting parallel text is the final output of our framework and its target side is the translation of the in-domain text.

To successfully restore the bilingual in-domain segments, we need the internal alignment information between the source and the target sides. Depending on the difficulty of extracting the simplified phrase pairs, different levels of granularity including phrase alignment, word alignment and word alignment score are needed. The restoration methods under different situations are summarized in FIGURE 5. We apply these restoration approaches one by one in the order of increasing granularity: **phrase alignment**, **word alignment** and **probability-based extraction**. As will be discussed in the later section, the empirical results show that these approaches are simple but can be effective to deal with most of the cases.

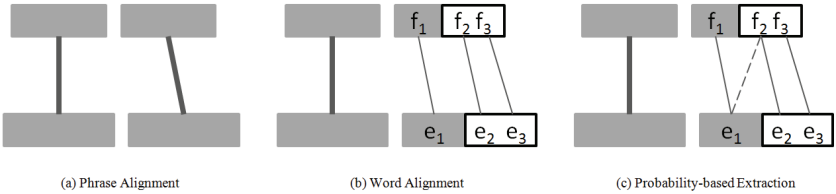
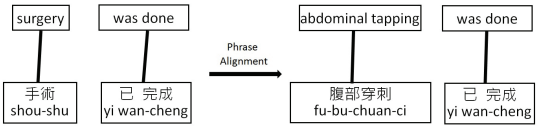
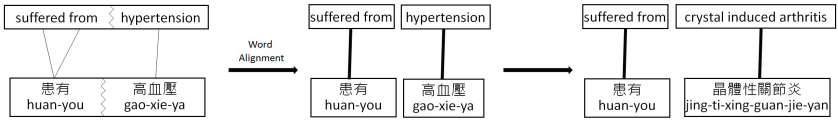


FIGURE 5 – Three restoration methods: phrase alignment, word alignment and probability-based extraction. The thick lines are phrase alignments and the thin lines are word alignments.

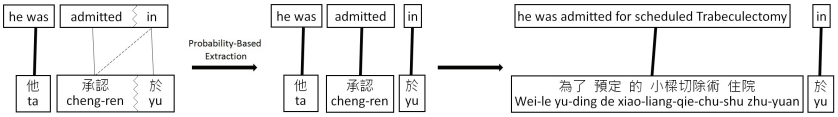
If a simplified source phrase is translated alone without its nearby context, phrase-level alignment information is sufficient to perform the restoration: we replace the simplified phrase pairs with the original bilingual in-domain segments. FIGURE 5(a) illustrates the phrase alignment method. The shaded blocks and thick lines denote simplified phrase pairs. By checking the phrase alignments provided by the decoder, the simplified phrase pairs can be replaced without further processing. The method is exemplified in FIGURE 6(a). In this example, the simplified term "surgery" is translated as a single phrase. Based on the phrase alignments, the simplified phrase pair "surgery-手術" is replaced with its original form "abdominal tapping-腹部穿刺".



(a) Restoration with phrase alignment.



(b) Restoration with word alignment.



(c) Restoration with probability-based extraction.

FIGURE 6 – Three restoration methods applied with different levels of alignment information. The thick lines are phrase alignments and the thin lines are word alignments.

In some cases, a simplified phrase is translated together with its nearby context, and therefore we need to determine the translation of the simplified phrase before we can restore its original form, such as the term "hypertension" in FIGURE 2(b). Phrase level alignments are insufficient now and higher granularity is needed, such as word-level alignments, to separate a simplified phrase from

its contexts. With word alignment information, we may extract a simplified phrase pair without violating the consistency judgment of a phrase pair (Och et al., 1999). We then replace the simplified phrase pair with its original bilingual in-domain segment. Our word alignment method is illustrated in FIGURE 5(b). The hollow blocks denote the context f_2f_3, which is translated together with its nearby simplified term f_1. The thin lines and the thick lines are word alignments and phrase alignments, respectively. Compliant with the consistency of phrase extraction, we separate the phrase pair f_1-e_1 from the phrase pair $f_1f_2f_3$-$e_1e_2e_3$, and perform the restoration. The method is exemplified in FIGURE 6(b). Based on the word alignments, the simplified term "hypertension" can be separated from its context "suffered from" without violating the consistency. Therefore, we can successfully restore the original form "crystal induced arthritis".

There are still cases in which word alignment method fails due to the fertility feature between source and target languages under IBM models (Brown et al., 1993). For a simplified term, which is usually a content word, its translation may be aligned to non-content words on the source side. In this case, extracting simplified phrase pairs would violate the consistency of phrase extraction. Here we apply a probability-based extraction of simplified phrases. This approach deletes weak word alignments based on alignment probabilities. For a source simplified phrase $f_{i,j}$ spanning from word f_i to f_j which is aligned to its target translation $e_{i,j}$, we examine each word in $e_{i,j}$. If there exists an e_k ($i<=k<=j$) which is aligned to two source words within $f_{i,j}$ and outside $f_{i,j}$ respectively, we try to delete one of the alignments by comparing their word alignment probabilities. FIGURE 5(c) illustrates the probability-based extraction method. The word e_1 is aligned to both f_1 and f_2. The fuzzy alignment causes the unsuccessful separation of f_1-e_1 from its context f_2f_3-e_2e_3. If the word alignment probabilities show $P(f_1|e_1) > P(f_2|e_1)$, we can delete the weak word alignment f_2-e_1, and meet the consistency judgment. The method is exemplified in FIGURE 6(c) where the word alignment method is unsuccessful. In this case, we fail to determine the translation of the simplified syntactic unit "he was admitted" because the target word "承認" is aligned to both "admitted" and "in". With the probability-based extraction approach, we delete the alignment (the dashed line) "in-承認", because P(in|承認) < P(admitted | 承認). After that, we are able to extract the phrase pair "admitted-承認". By determining the translation of "he was admitted" with two consecutive phrase pairs, the restoration of the original form "he was admitted for scheduled Trabeculectomy" can be done easily.

4 Experiments

We experiment our *STR* framework on an English-Chinese SMT application in the medical domain. Moses is built as our general domain SMT system. The translation model is trained on 6.8M sentence pairs collected from Hong Kong corpus (LDC2004T08) and UN corpus (LDC2004E12). We train the trigram language model on the Chinese part of the above parallel corpus and the Central News Agency part of the Tagged Chinese Gigaword (LDC2007T03).

We test our method with the English-Chinese translation task on the medical summaries from the National Taiwan University Hospital (NTUH). The dataset comprises 1,077 parallel sentences within 18 medical summaries. In our SMT application, the gap between in-domain and out-domain corpora is very large in terms of vocabulary and writing styles. In our test set, the average length of a sentence in a medical summary is short (12.58 words) compared to the background general corpus (29 words). TABLE 1 shows some interesting statistics of the domain specific segments in the test set, including the average number of terminological units, syntactic units and OOV words per sentence, and the percentages of words they account for in the test data.

On the average, the in-domain segments (terminological and syntactic units) constitute over 36% of the experimental dataset. Nearly 21% of OOV words including surgical, diagnosis and drug terms occur in a sentence and most of them are parts of terminological and syntactic units. This justifies our motivation to alleviate the OOV problem in our cross-domain SMT system by simplifying the text in this special domain before sending it to the background SMT system.

	Occurrence	Percentage
Terminological Unit	2.04/sentence	18.37%
Syntactic Unit	0.65/sentence	17.86%
OOV Word	2.93/sentence	21.18%

TABLE 1 – Statistics of the in-domain segments and the OOV words in the test set.

To identify the in-domain segments in a text, we collect bilingual terminological and syntactic units from the medical domain. Section 3.1 describes the identification step and discusses various ways to collect the bilingual in-domain knowledge with automatic and semi-automatic approaches. In our experimental domain (i.e., English medical summaries), bilingual medical dictionaries are available from plenty of resources and thus they are sufficient for collecting bilingual terminological units. On the other hand, bilingual syntactic units are relatively hard to obtain. Only monolingual corpus can be obtained in this domain, and obtaining parallel text by manual translation incurs high cost due to the involvements of domain experts (e.g., doctors). Therefore we collect the bilingual syntactic units by taking a semi-automatic approach in order to make the best of manual efforts. The collection of terminological and syntactic units is briefly described below, and readers can refer to Chen et al. (2011) for more details.

The terminological units are collected from both public and non-public resources. They include public medical dictionaries from domestic medical colleges and Department of Health. We are also provided with frequent bilingual term pairs used by NTUH staff. Merging these bilingual dictionaries causes ambiguity in which a medical term has multiple translations with similar meanings but in different styles. Since consistent translation of terminology is desired in this application, ambiguous translations are reviewed and edited by the staff at NTUH. So far nearly 70,000 bilingual terminological units are collected and stored in our database.

The syntactic units are semi-automatically collected with a pattern mining algorithm and annotations by the in-domain experts. Since we choose phrase-based SMT as our background SMT system, the n-gram based syntactic units are preferred for easier integration. We first automatically extract the candidate n-grams from the experimental medical summary corpus with medical entity recognition (Ben Abacha and Zweigenbaum, 2009) and NLP techniques. These source language n-grams are then reviewed and fast translated by doctors with the help of a user-friendly annotation tool. The resulting bilingual n-grams are then collected and organized into the bilingual syntactic units. The NLP techniques aim to reduce the cost of annotations by doctors, and increase the coverage of bilingual syntactic units. TABLE 2 gives some samples of these n-gram based syntactic units in this domain. Both source and target n-grams are listed for each bilingual syntactic unit. The words in bold denote the medical categories which represent diseases or symptoms (**DIAGNOSIS**), medical tests (**TEST**), surgical or non-surgical treatments (**TREATMENT**), etc. These syntactic units capture in-domain writing styles and local reorderings (note the different orders of medical categories between source and target sides).

Source Syntactic Unit Target Syntactic Unit	Syntactic Label
underwent **TEST** on **DATE**	VP
於 **DATE** 接受 **TEST** 檢查 yu jie-shou jian-cha	
DRUG was given for **DIAGNOSIS**	S
使用 **DRUG** 用於治療 **DIAGNOSIS** shi-yong yong-yu-zhi-liao	
received **TREATMENT** with **DRUG**	VP
接受 **TREATMENT** 及 **DRUG** 治療 jie-shou ji zhi-liao	
DIAGNOSIS at the right **REGION**	NP
在右側 **REGION** 之 **DIAGNOSIS** zai-you-ce zhi	
TREATMENT of the right **REGION**	NP
右側 **REGION** 的 **TREATMENT** you-ce de	

TABLE 2 – Samples of bilingual syntactic units.

In the simplification step, we simplify the identified in-domain segments of a text. For runtime efficiency, we perform the simplification on all of the collected terminological and syntactic units in advance and store the results to avoid redundant work. The terminological units are simplified based on the ontology provided by Unified Medical Language System (UMLS). For a medical term, we search for its hypernyms and find the most frequent one in the background translation model and designate it as the simplified form.

The syntactic units are parsed with Stanford parser (Klein and Manning, 2003) and simplified with the rules described in Section 3.2. TABLE 3 gives the number of syntactic units for each syntactic label. There are parsing errors for few n-grams and they are manually corrected. As shown in TABLE 3, more than 70% of the n-grams are correctly simplified. Most of the errors come from the unexpected syntactic labels which can be processed with new simplification rules. For example, the common syntactic unit "was admitted due to **DIAGNOSIS**" can be simplified to "was admitted". However, the syntactic label ADJP which covers "due to **DIAGNOSIS**" is not considered in our rules and therefore remains unchanged after the simplification step. We plan to devise a more robust method beyond the rule-based one for future work.

In the restoration step, we receive and post-process the SMT results as described in Section 3.4. In our experiments, phrase level alignments are provided by the Moses decoder, and word level alignments are obtained as the intermediate results after training the phrase table. Our methods successfully perform the restorations on most of the test data. Total 1,004 (93.22%) of the 1,077 sentences can be successfully restored with the three proposed restoration methods. For the other 73 sentences, most of these failure cases result from translation errors of the simplified phrases by Moses, which in turn affect the word alignments and introduce difficulties. TABLE 4 shows the performance of each restoration method. The second column counts the number of applications of each restoration method on the test set, and the third column counts how many

Syntactic Unit	Count	Parsing Error	Simplification Error
NP	697	4.85%	28.15%
VP	287	2.94%	27.12%
PP	228	1.85%	8.89%
S	342	1.23%	9.59%
Other	12	33.33%	

TABLE 3 – Parsing and simplification performances on syntactic units. We manually simplify the 12 syntactic units with minority labels.

	Total Count	Sentence Count
Phrase Alignment	1,767 (60.93%)	865 (86.16%)
Word Alignment	981 (33.83%)	657 (65.44%)
Probability-based Extraction	152 (5.24%)	137 (13.65%)

TABLE 4 – Various restoration methods.

sentences include the application of each restoration method. Total 2,900 applications of the restoration methods have been done in the testing. With phrase alignments, we can deal with over 60% of the simplified phrases. For the remaining simplified phrases, the background SMT model captures their contexts with higher confidence, and therefore they are translated with the surrounding words. For these cases, we use word alignment information, and perform the word alignment and the probability-based extraction methods. Although the probability-based extraction accounts for only 5.24% of the restorations, it is applied on 13.65% of the sentences, as shown in the third column. This confirms that the approach of deleting weak word alignment is simple but effective in handling the inconsistency during phrase extraction.

We compare our *STR* framework against three baselines. These four SMT systems share the same background translation and language models which are trained from the corpus described in the beginning of this section. The Moses system is trained with its default scripts without any in-domain prior knowledge. We choose Moses' default behaviour to handle OOV words, i.e., they are copied to translation results. We also use an advanced feature of Moses to build another setting (Moses+Terminology) by adding an in-domain phrase table with the terminological units. It serves as the back-off model during the runtime decoding. That is, Moses searches the back-off model only when no translation for a phrase is found in the background model. In the naïve integration system, we apply the identification step to a source sentence and mark the in-domain segments. Rather than simplifying these segments, we use the XML markup function of Moses to force the translations of these terminological and syntactic units.

TABLE 5 shows 4-gram BLEU (Papineni et al., 2002) and TER (Snover et al., 2006) scores of these systems. With poor in-domain knowledge, the baseline Moses achieves the low performance on the test set. On the other hand, the Moses system with terminology back-off model eliminates most of the OOV problems and gets better performance. The naïve integration system gives further improvement over the Moses+Terminology system by performing medical entity recognition on a source input before sending it to Moses. Therefore, it can identify the multiword terminological units and the syntactic units during the identification step. The system

with the *STR* framework achieves the best translation performance among the four systems measured by both BLEU and TER scores. Comparing the translation results of the *STR* framework and the naïve integration system, we find the former gives better translations and reorderings on the general segments. In summary, the experimental results and observations show that our simplification approach improves the baseline SMT systems by identifying specific segments with bilingual in-domain knowledge and modifying these special segments to better fit the background SMT model.

	BLEU-4	TER
Moses	14.35	64.911
Moses+Terminology	20.56	55.712
Naïve Integration	26.29	47.683
STR Framework	35.87	40.650

TABLE 5 – Translation performances on medical summaries.

Conclusion and Future Work

This paper proposes a simplification-translation-restoration framework for cross-domain applications in SMT. We integrate bilingual in-domain knowledge into a background out-domain SMT system. That deals with the cross-domain and the data sparseness problems at the same time. The in-domain text goes through identification, simplification, translation, and restoration steps. Important issues are addressed and discussed for each step, including preparing bilingual in-domain knowledge, simplification with syntactic information, and different restoration strategies for extracting simplified phrases from the SMT results. We evaluate the performance of our framework through a cast study of medical summary translation. The empirical results show the effectiveness of our approach at each step of the framework. For the end-to-end translation task, our method outperforms the background SMT system and the systems with different integration approaches.

Currently, we apply a rule-based simplification approach to four common syntactic units, i.e., NP, VP, PP, and S. The alternative is to simplify in-domain segments based on statistical evidence. Searching for a simplified form for a phrase in a more robust way will give more hints to aid the background SMT system. That is worthy of further investigation. Besides, to introduce a feedback mechanism to improve an *STR*-based MT system is also one of the research issues. Which parts, pre-processing (i.e., identification and simplification), translation, and post-processing (i.e., restoration), have to be modified through the feedback and how they are modified are critical. In our scenario, the doctors post-edit the MT results of English medical summaries and produce the correct Chinese medical summaries. The editing logs can be analysed to improve identification, simplification, translation, and restoration steps for further domain adaptation.

Acknowledgments

This work was partially supported by National Science Council (Taiwan) and Excellent Research Projects of National Taiwan University under contracts NSC101-2221-E-002-195-MY3 and 101R890858.

References

Abacha, A. B. and Zweigenbaum, P. (2011). Medical entity recognition: a comparison of semantic and statistical methods. In *Proceedings of the 2011 Workshop on Biomedical Natural Language Processing*, pages 56–64.

Aziz, W., Dymetman, M., Mirkin, S., Specia, L., Cancedda, N., and Dagan, I. (2010). Learning an expert from human annotations in statistical machine translation: the case of out-of-vocabulary words. In *Proceedings of EAMT 2010*.

Bertoldi, N. and Federico, M. (2009). Domain adaptation for statistical machine translation with monolingual resources. In *Proceedings of the Fourth Workshop on Statistical Machine Translation*, pages 182–189.

Callison-Burch, C., Koehn, P. and Osborne, M. (2006). Improved statistical machine translation using paraphrases. In *Proceedings NAACL 2006*, pages 17–24.

Carpuat, M. and Wu, D. (2007). Improving statistical machine translation using word sense disambiguation. In *Proceedings of EMNLP 2007*, pages 61–72.

Chan, Y. S., Ng, H. T. and Chiang, D. (2007). Word sense disambiguation improves statistical machine translation. In *Proceedings of ACL 2007*, pages 33–40.

Chen, H., Huang H., Tjiu, J., Tan, C. and Chen, H. (2011). Identification and translation of significant patterns for cross-domain SMT applications. In *Proceedings of Machine Translation Summit XIII*, pages 277–284.

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

Civera, J. and Juan, A. (2007). Domain adaptation in statistical machine translation with mixture modeling. In *Proceedings of the Second Workshop on Statistical Machine Translation*, pages 177–180.

Foster, G., Goutte, C., and Kuhn, R. (2010). Discriminative instance weighting for domain adaptation in statistical machine translation. In *Proceedings of EMNLP 2010*, pages 451–459.

Haghighi, A., Liang, P., Berg-Kirkpatrick, T. and Klein, D. (2008). Learning bilingual lexicons from monolingual corpora. In *Proceedings of ACL 2008*, pages 771–779.

Huang, L. and Chiang, D. (2007). Forest rescoring: Faster decoding with integrated language models. In *Proceedings of ACL 2007*, pages 144–151.

Klein, D. and Manning, C. D. (2003). Accurate unlexicalized parsing. In *Proceedings of ACL 2003*, pages 423–430.

Koehn, P., Och, F. J. and Marcu, D. (2003). Statistical phrase-based translation. In *Proceedings of HLT-NAACL 2003*, pages 127–133.

Koehn, P. (2004). Pharaoh: a beam search decoder for phrased-based statistical machine translation models. In *Proceedings of AMTA 2004*, pages 115–124.

Koehn, P., Hoang, H., Birch, A., Callison-Burch, C., Federico, M., Bertoldi, N., Cowan, B., Shen, W., Moran, C., Zens, R., Dyer, C., Bojar, O., Constrantin, A., and Herbst, E. (2007). Moses:

Open source toolkit for statistical machine translation. In *Proceedings of ACL 2007*, Demonstration Session, pages 177–180.

Marton, Y., Callison-Burch, C. and Resnik, P. (2009). Improved statistical machine translation Using monolingually-derived paraphrases. In *Proceedings of EMNLP 2009*, pages 381–390.

Matsoukas, S., Rosti, A. I. and Zhang, B. (2009). Discriminative corpus weight estimation for machine translation. In *Proceedings of EMNLP 2009*, pages 708–717.

Morin, E., Daille, B., Takeuchi, K. and Kageura, K. (2007). Bilingual terminology mining - using brain, not brawn comparable corpora. In *Proceedings of ACL 2007*, pages 664–671.

Och, F. J., Tillmann, C. and Ney, H. (1999). Improved alignment models for statistical machine translation. In *Proceedings of EMNLP 1999*, pages 20–28.

Papineni, K., Roukos, S., Ward, T. and Zhu, W. (2002). BLEU: a method for automatic evaluation of machine translation. In *Proceedings of ACL 2002*, pages 311–318.

Resnik, P., Buzek, O., Hu, C., Kronrod, Y., Quinn, A. and Bederson, B. (2010). Improving translation via targeted paraphrasing. In *Proceedings of EMNLP 2010*, pages 127–137.

Snover, M., Dorr, B., Schwartz, R., Micciulla, L. and Makhoul, J. (2006). A study of translation edit rate with targeted human annotation. In *Proceedings of AMTA 2006*, pages 223–231.

Wang, C., Collins, M. and Koehn, P. (2007). Chinese syntactic reordering for statistical machine translation. In *Proceedings of EMNLP 2007*, pages 737–745.

Woodsend, K. and Lapata, M. (2011). Learning to simplify sentences with quasi-synchronous grammar and integer programming. In *Proceedings of EMNLP 2011*, pages 409–420.

Wu, J. and Chang, J. S. (2007). Learning to find English to Chinese transliterations on the web. In *Proceedings of EMNLP-CoNLL 2007*, pages 996–1004.

Wubben, S. and van den Bosch, A. and Krahmer, E. (2012). Sentence simplification by monolingual machine translation. In *Proceedings of ACL 2012*, pages 1015–1024.

Xiong, D., Liu, Q. and Lin, S. (2006). Maximum entropy based phrase reordering model for statistical machine translation. In *Proceedings of ACL-COLING 2006*, pages 521–528.

Zhao, B., Eck, M. and Vogel, S. (2004). Language model adaptation for statistical machine translation via structured query models. In *Proceedings of COLING 2004*, pages 411–417.

Zhu, Z., Bernhard, D. and Gurevych, I. (2010). A monolingual tree-based translation model for sentence simplification. In *Proceedings of COLING 2010*, pages 1353–1361.

A Semi-Supervised Bayesian Network Model for Microblog Topic Classification

Yan Chen[1,2] *Zhoujun Li*[1] *Liqiang Nie*[2] *Xia Hu*[3]
Xiangyu Wang[2] *Tat−Seng Chua*[2] *Xiaoming Zhang*[1]
(1) State Key Laboratory of Software Development Environment, Beihang University, Beijing, China
(2) School of Computing, National University of Singapore, Singapore
(3) Arizona State University, United States
chenyan@cse.buaa.edu.cn, lizj@buaa.edu.cn, nieliqiang@gmail.com,
xia.hu@asu.edu, dcswangx@nus.edu.sg, dcscts@nus.edu.sg,
yolixs@buaa.edu.cn

Abstract

Microblogging services have brought users to a new era of knowledge dissemination and information seeking. However, the large volume and multi-aspect of messages hinder the ability of users to conveniently locate the specific messages that they are interested in. While many researchers wish to employ traditional text classification approaches to effectively understand messages on microblogging services, the limited length of the messages prevents these approaches from being employed to their full potential. To tackle this problem, we propose a novel semi-supervised learning scheme to seamlessly integrate the external web resources to compensate for the limited message length. Our approach first trains a classifier based on the available labeled data as well as some auxiliary cues mined from the web, and probabilistically predicts the categories for all unlabeled data. It then trains a new classifier using the labels for all messages and the auxiliary cues, and iterates the process to convergence. Our approach not only greatly reduces the time-consuming and labor-intensive labeling process, but also deeply exploits the hidden information from unlabeled data and related text resources. We conducted extensive experiments on two real-world microblogging datasets. The results demonstrate the effectiveness of the proposed approaches which produce promising performance as compared to state-of-the-art methods.

Title and abstract in Chinese

基于半监督贝叶斯网络模型的微博主题分类模型研究

微博作为一种新型的社会媒体，其海量数据及展现主题的多样性使得用户要找到感兴趣主题的微博非常困难。 当前的一些研究主要采用面向长文本的挖掘方法来分类微博的主题，但微博作为一种短文本，其稀疏性和用词不规范严重影响了这些方法的性能。 针对微博的特点，本文提出了一种半监督贝叶斯网络模型，其充分利用外部相关的网络资源来丰富微博的文本特征， 并利用少量的未标注数据以及辅助的外部资源来预测大量待标注微博数据的主题。 本文的方法不仅能减少繁琐的人工标注过程，而且能够从未标注的微博数据以及相关资源中挖掘出微博隐藏主题的相关语义信息。 我们的实验基于Twitter和新浪微博两个数据集，试验结果表明，与目前的方法相比，本文提出方法的性能有显著提高。

Keywords: Semi-supervised algorithm, microblog classification, probabilistic graph model.

Keywords in L_2: 半监督分类算法，微博，话题分类，概率图模型.

Proceedings of COLING 2012: Technical Papers, pages 561–576,
COLING 2012, Mumbai, December 2012.

1 Introduction

Microblogging services are becoming immensely popular in breaking-news disseminating, information sharing, and events participation. This enables users to express their thoughts and intentions in short textual snippets on a daily and even hourly basis. The most well-known one is Twitter (`www.twitter.com`), which has more than 140 million active users with 1 billion Tweets every 3 days[1] as of March 2012. Over time, a tremendous number of messages have been accumulated in their repositories, which greatly facilitate general users seeking information by querying their interested topics using the corresponding hashtag.

However, users often have to browse through large amount of results in order to find the information of their interests. This is due to the ambiguous hashtag and the presentation style. The microblogging platforms mix search results in a ranked list, determined by their relevance to the corresponding hashtag and published time. Unfortunately, most hashtags are very short, ambiguous and even vague, leading to unsatisfactory search results. For example, the returned list for queried hashtag "#apple" is extremely messy and diversified, potentially covering several different sub-topics: smartphone, computer, fruit and so on. In this case, users can benefit from overviews of search results based on meaningful and structural categories, such as, grasping at a glance the spread of categories covered by a given search topic and quickly locating the information of their interests with the assistance of the labeled categories. This is especially important for mobile search through handheld devices such as smartphones.

Classifying microblogs into pre-defined subtopic-oriented classes poses new challenges due to the following reasons. First, unlike normal documents, these messages are typically short, consisting of no more than 140 characters. They thus do not provide sufficient word co-occurrences or shared contexts for effective similarity measure (Hu et al., 2009). The data sparseness hinders general machine learning methods to achieve desirable accuracy. Second, microblogging messages are not well conformed as standard structures of documents. Sometimes they do not even obey grammatical rules (Hu and Liu, 2012b). Third, microblogs lack label information. It is time and labor consuming to label the huge amounts of messages.

Intensive efforts have been made on the classification of short texts utilizing machine learning techniques (Nie et al., 2011). Some representative research efforts are based on topic model (Ramage et al., 2009) (Zhao et al., 2011). As these approaches heavily rely on the term co-occurrence information, the sparsity of short and informal messages unduly influence the significant improvement of the performance. Some others explore some traditional supervised learning methods to classify microblogging messages (Lee et al., 2011) (Zubiaga et al., 2011) (Sriram et al., 2010) (Tang et al., 2012). The sparsity problem again hinders the similarity measurement. Moreover, it is laborious and time consuming to obtain labeled data from microblogging. Consequently, new approaches towards microblog classification are highly desired.

In this paper, we propose a semi-supervised learning approach to the classification of microblogging messages. We aim to tackle three challenges in this paper. First, to handle the data sparseness problem, our approach submits a query that is related to hashtag and category to Google Search Engine; meanwhile it incorporates the external information provided by search engine results to enrich the short microblogs. Second, to alleviate negative effect brought by informal words in microblogging, we employ linguistic corpus to detect informal words in microblogging messages and correct them into formal expressions. Third, with the integration of hashtag related resources,

[1] http://blog.twitter.com/2012/03/twitter-turns-six.html

our model is robust with only a small amount of training data, which greatly reduces the manually labeling costs. Our algorithm alternates between performing an E-step and M-step. Specifically, it first trains a classifier based on the available labeled messages as well as some auxiliary cues mined from the web, and probabilistically predicts the class labels of the unlabeled messages. It then trains a new classifier using all messages and the auxiliary cues, and iterates to convergence. We conduct experiments on the real-world datasets, and demonstrate that our proposed scheme yields significant accuracy in microblogging messages categorization.

The main contributions of this research can be summarized as follows,

- To the best of our knowledge, this work is the first attempt towards microblogs categorization using semi-supervised learning approach, which requires less labeled data and can thus be practically extended to large-scale datasets.
- Our approach incorporates external statistical knowledge to enrich the short microblogs, which greatly remedies the data sparseness issue.
- Our approach adopts a category-word distribution analysis, which well addresses the broader phenomenon existed in microblogs: non-standard language presentation and abundant spelling errors.

The reminder of this paper is organized as follows: we introduce the details of our proposed approach and experimental results in Section 2 and Section 3 respectively. In section 4, we briefly reviews of the related work, followed by concluding remarks in Section 5.

2 Semi-Supervised Graphical Model for Microblogs Classification

Before formulating our approach, we first define some notations. A set of messages is collected by a given hashtag t, which are partitioned into two subsets: a labeled set $M^l = \{m_1, m_2, \ldots, m_L\}$ and an unlabeled set $M^u = \{m_{L+1}, m_{L+2}, \ldots, m_{L+N}\}$. M^l includes only the example messages provided through user interaction, where each instance is associated with a predefined category c_i with belonging to $C = \{c_1, c_2 \ldots c_K\}$; while M^u includes all the other messages. We aim to predict the category label for each data point in M^u. Here we assume that each tweet belongs to only one category. Similar idea of assigning a single topic or category to a short sequence of words has been used before in (Diao et al., 2012) (Gruber et al., 2007) (Zhao et al., 2011).

2.1 The General Framework

We now introduce the overview of the whole processing that aims to classify microblogging messages by exploiting the internal and external resources. The workflow consists of three phrases, as shown in Figure 1. It includes the preprocessing of external resources, preprocessing of microblogging messages, and construction of Semi-Supervised Bayesian Network (SSBN) model.

Phrase 1: Preprocessing of External Resources Due to their short length, microblogging messages do not provide sufficient word co-occurrence or context shared information for effective similarity measure. Thus we utilize the external Google Search snippets to enrich the original feature space of the microblogging messages. The procedure of enrichment is as follows: we mine the list of hot topics from Google such as Apple, Obama, NBA, Facebook, *etc*. For each hot topic, we search them as hashtags for microblog messages from Twitter. The results contain a list of proper sub-hashtags, such as stock, ipad, ipo, app, ticket, education, *etc*. These sub-hashtags are manually

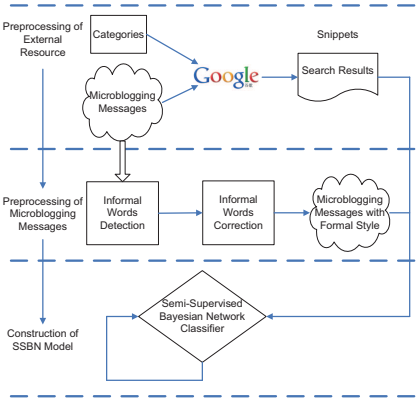


Figure 1: The General Framework

classified into K pre-defined categories. For a given hashtag t (for example, stock), we build K hashtag-category pairs (for example, stock Sports, stock Business, *etc.*), and consider each pair as a query to return 20 extended documents from Google Search Engine, denoted as S. Comparing with the way that only takes each hashtag as a query, the combination of hashtag and category can find more accurate documents. Next, we assign the tf.idf weight of each word for each category in S. We further use the google search results to estimate the category prior distribution.

Phrase 2: Preprocessing of Microblogging Messages It is worth noting that there is a large amount of misspelled and informal expressions in microblogging messages. This is different from the formal expressions and words used in Google Search results. To handle this mismatch problem, we first construct a microblog dictionary containing all the abbreviate forms of words used in Twitter from some dictionaries, such as Twitternary[2], twitterforteachers[3]. The dictionary contains 727 words. Giving a microblogging message, we first use this dictionary to detect the informal words, then correct them to the formal words. In this way, we are also able to collect more words related to the predefined categories from the labeled messages to tackle the sparseness problem in microblogging messages.

Phrase 3: Construction of SSBN model In order to fully integrate hashtag related resources and unlabeled data to a classifier, we propose a semi-supervised Bayesian network model. The semi-supervised classifier can offer robust solution to microblog topic classification for two reasons. First, it utilizes those labeled microblogging messages with hashtags by training a topic model based classifier, which is then used to find the category (label) distribution of unlabeled messages accurately. Second, it leverages the related external resources to provide a valuable context to microblogging messages. In this way, compared with supervised learning methods, we need only few labeled data for training. The details of SSBN model construction will be introduced in the next subsection.

[2]http://www.twittonary.com/
[3]http://twitterforteachers.wetpaint.com/page/Twitter+Dictionary

θ	The vector indicating category weights for message data collection.
ϕ	The vector indicating category weights for specific message.
θ', ϕ'	The $\|C\| \times \|N\|$ matrix indicating category-word distribution.
λ	The contribution of unlabeled data to prior probability.
α	The contribution of prior knowledge from θ.
$1-\alpha$	The contribution of prior knowledge from ϕ.
β	The contribution of likelihood probability from θ'.
$1-\beta$	The contribution of likelihood probability from ϕ'.
η_d, η_g	Hyperparameters and priors of Dirichlet distributions.
C	The category vector.
c_j	The jth category.
M	The message collection in the original message data.
m	The message.
N	The word collection in the original message data.
t	The hashtag.
w	The word.
y	The category label of message.

Table 1: Important notations used in this paper and their descriptions.

2.2 Probabilistic Graph Model Construction

The above formulations intuitively reflect that the category prediction task comprises two estimations: coarse-grained category distribution and fined-grained category-word distribution. It is schematically illustrated in Figure 2, in which the corresponding notations are summarized in Table 1.

1. Category distribution: There are two kinds of category distribution in the data. Let θ denotes the category distribution obtained from the original message M, which is a weight vector representing the weight for each category. Similarly, let ϕ denotes the category distribution for external resources obtained from the search results S. The category distribution for the total data D is assumed to be a linear combination of θ and ϕ. Parameter α is employed as the weight to adjust the contributions of different sources. In addition, the original message data also consists of labeled and unlabeled data; and λ is used to denote the contribution of unlabeled data in generating the category distribution for M.

2. Category-word distribution: The category-word distribution also has two parts: θ' denotes the distribution of different words over different categories in the original messages, which is a $|C| \times |N|$ matrix. Here, $|C|$ is the number of categories, and $|N|$ is the number of words in the data. Similarly, ϕ' denotes the category-word distribution in the search results. The category-word distribution for data D is again assumed to be a linear combination of θ' and ϕ', where parameter β is employed as the weight to adjust the contributions of different sources.

Our semi-supervised Bayesian Network (SSBN) belongs to probabilistic graphical model, which formally denotes the probability of a message m falling into a category c as,

$$P(c|m) = \frac{P(c)P(m|c)}{\sum_c P(c)P(m|c)} \tag{1}$$

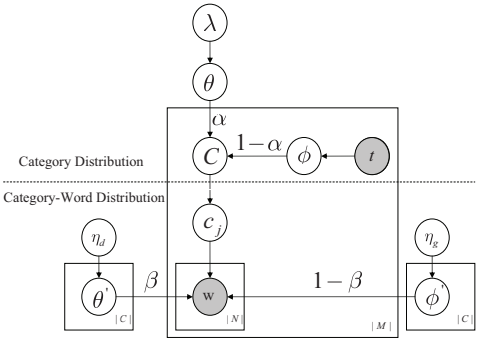


Figure 2: Probabilistic graphical representation of semi-supervised Bayesian network model.

where $P(c)$ is the prior probability of category in the message data collection. By assuming the presence of a word w is independent to the presence of any other word in m, we derive

$$P(m|c) = \prod_{w \in m} P(w|c) \tag{2}$$

2.3 Parameter Inference

In this section, we turn our attention to procedures for parameter inference with EM approach. In the expectation step, the distributions θ, ϕ, $\hat{\theta'}^{w_k}_{c_j}$ and $\hat{\phi'}^{w_k}_{c_j}$, will be estimated. Besides the labeled data and external resource, the parameter estimations also make use of the unlabeled data. Initially we assign category labels to unlabeled data with an uniform distribution, i.e., the probability is $\frac{1}{|C|}$ for each category. In the following iterations, labels of unlabeled data and SSBN model are alternatively updated and reinforced until convergence.

Estimating θ: θ represents the probability of each category in the original message data collection. It is proportional to the expected number of messages that was assigned to this category.

$$\hat{\theta_{c_j}} \equiv P(c_j|\hat{\theta}) = \frac{1 + \sum_{i=1}^{|M|} \Lambda(i) P(y_i = c_j|m_i)}{|C| + |M^l| + \lambda|M^u|} \tag{3}$$

As aforementioned in section 2.2, the message data collection consists of labeled messages M^l and unlabeled messages M^u. They have different contribution to the category probability estimation. The function $\Lambda(i)$, defined as in equation (4), is employed to achieve that goal. The parameter $\lambda \in [0, 1]$.

$$\Lambda(i) = \begin{cases} \lambda & \text{if} \quad m_i \in M^u; \\ 1 & \text{if} \quad m_i \in M^l. \end{cases} \tag{4}$$

Estimating ϕ: ϕ denotes the prior category probability distributes over the Google Search results. In this paper, the prior probability of category c_j for a hashtag t completely depends on the

relationship between the corresponding hashtag t and the predefined category names,

$$\hat{\phi_{c_j}} \equiv P(c_j|\hat{\phi}) = \frac{\frac{1}{NGD(t,c_j)} + \mu}{\sum_{j=1}^{|C|} \frac{1}{NGD(t,c_j)} + |C|\mu} \tag{5}$$

where μ is a smoothing factor and $NGD(t, c_j)$ is the Normalized Google Distance[4], which is employed to calculate distance between the tag t and the category c_j. It can be observed that a smaller value of NGD leads to more contribution of c_j for the specific message.

Estimating θ' and ϕ': θ' and ϕ' respectively denote the category-word distributions over original message collection and Google Search results. Both of them are $|C| \times |N|$ matrices. They can be estimated using the following formulas:

$$\hat{\theta'}_{c_j}^{w_k} \equiv P(w_k|c_j, \hat{\theta'}) = \frac{n_{d_{c_j}}^{w_k} + \eta_d}{\sum_{p'=1}^{|N|} n_{d_{c_j}}^{w_{p'}} + |N|\eta_d} \tag{6}$$

$$\hat{\phi'}_{c_j}^{w_k} \equiv P(w_k|c_j, \hat{\phi'}) = \frac{n_{g_{c_j}}^{w_k} + \eta_g}{\sum_{q'=1}^{|N|} n_{g_{c_j}}^{w_{q'}} + |N|\eta_g} \tag{7}$$

where $n_{d_{c_j}}^{w_k}$ and $n_{g_{c_j}}^{w_k}$ are respectively the number of times that the word w_k has occurred in the category c_j in message data collection and Google Search results (retrieved by the combination of hashtag t and the name of the j-th category). η_d and η_g are hyperparameters with a small value for smoothing purpose to avoid the zero problem.

The maximum likelihood category label for a given message m_i is,

$$y_i = \arg\max_{c_j} P(c_j|m_i, \hat{\theta}, \hat{\phi}, \hat{\theta'}, \hat{\phi'}) = \frac{P(c_j|\hat{\theta}, \hat{\phi}, \hat{\theta'}, \hat{\phi'})P(m_i|c_j, \hat{\theta}, \hat{\phi}, \hat{\theta'}, \hat{\phi'})}{P(m_i|\hat{\theta}, \hat{\phi}, \hat{\theta'}, \hat{\phi'})} \tag{8}$$

where $P(m_i|\hat{\theta}, \hat{\phi}, \hat{\theta'}, \hat{\phi'})$ is formally written as follows,

$$P(m_i|\hat{\theta}, \hat{\phi}, \hat{\theta'}, \hat{\phi'}) = \sum_{c_j} P(c_j|\hat{\theta}, \hat{\phi}, \hat{\theta'}, \hat{\phi'})P(m_i|c_j, \hat{\theta}, \hat{\phi}, \hat{\theta'}, \hat{\phi'}) \tag{9}$$

where the prior probability for category c_j is obtained by linearly fusing two estimations on two resources,

$$P(c_j|\hat{\theta}, \hat{\phi}, \hat{\theta'}, \hat{\phi'}) = P(c_j|\hat{\theta}, \hat{\phi}) = \tilde{\alpha}P(c_j|\hat{\theta}) + (1 - \tilde{\alpha})P(c_j|\hat{\phi}) \tag{10}$$

where α is a trade-off parameter to balance the contributions between two kinds of category distribution. The maximum likelihood probability for the each message m_i can be derived as:

$$\begin{aligned} P(m_i|c_j, \hat{\theta}, \hat{\phi}, \hat{\theta'}, \hat{\phi'}) &= P(m_i|c_j, \hat{\theta'}, \hat{\phi'}) = \prod_{k=1}^{|m_i|} P(w_k|c_j, \hat{\theta'}, \hat{\phi'}) \\ &= \prod_{k=1}^{|m_i|} \{\beta P(w_k|c_j, \hat{\theta'}) + (1 - \beta)P(w_k|c_j, \hat{\phi'})\} \end{aligned} \tag{11}$$

Similar to α, β is tuned to control the contribution between the the category-word distribution over two different resources.

[4] `http://en.wikipedia.org/wiki/Normalized_Google_distance`, here in case of NGD not equal to zero, we add a small constant closing to zero.

3 Experiments

In this section, we first evaluate our proposed model on two real-world datasets, utilizing a range of popular metrics. We then compare our model with the state-of-the-art text classification approaches on microblogs. Also, we study the sensitivity of the training dataset size, convergence analysis followed by the impact analysis on the parameters.

3.1 Experimental Settings

In our experiments, two large-scale real-world datasets were constructed:

- **Twitter**: The Twitter dataset was generated from Trec-Twitter2011[5]. First, we collected 10 hot topics from Google Trends[6], including NBA, Apple, facebook, *etc.* For each topic, we manually selected several low-level sub-topics and combined each of them with the high-level topic. Take the topic "Apple" as an example. We extended it with "Apple stock", "Apple ipad", etc. We manually determine which category the sub-topics belong to. For example, "stock" is classified to Business, while "ipad" is assigned to science. These pairs are naturally viewed as queries. Then the Twitter dataset was constructed by retrieving all the related messages from Trec-Twitter2011 based on these queries. To validate the robustness of our proposed model on partially noisy data, we deliberately did not provide ground truth for this dataset. Instead, the returned messages under a query are directly considered as belongings to the same category as the sub-topic. The Twitter dataset is in this way labeled semi-automatically based on sub-topics. The ground truth is so-called pseudo ground truth. For example, all the messages searched by "Apple stock" are regarded as business category.
- **Sina Weibo**: Based on selected trending topics of Sina Weibo, we crawled a collection of messages. And then manually assigned each messages into one of 7 predefined categories: sports, politics, science&tech, game, movie, music and others. The messages fallen into "others" are removed; and up to $15,811$ unique messages were remained. To build the ground truth, we adopted a manual labeling procedure. We divided 15 people with different background into 3 teams to manually label these messages. Every team labeled the complete dataset. The voting method was employed to combine the label results from different teams. For each message, only one category label with the majority voting was selected as the ground truth label. For the cases that a message received three different categories, a discussion was carried out among the labelers to decide the final ground truths.

The distributions of different categories over two datasets are displayed in Table 2. For each dataset, we devise 4 test configurations with different amount of training data: 5%, 20%, 50% and 90% for training respectively, and use the corresponding reminders for testing. The training data is randomly selected.

In this work, we utilize several widely-used performance metrics to evaluate our classification task: average accuracy, precision, recall, and $F1$ score (Sokolova and Lapalme, 2009) (Rosa et al., 2011). Average accuracy evaluates the average effectiveness for each category of a classifier. Precision is the fraction of retrieved messages that are relevant to the search, while recall is the percentage of the relevant messages that are successfully retrieved, and F1 measure combines both of recall and precision. For some cases, we also provide the $macro-$ and $micro-$ values. The $micro-$ assigns equal weight to each message, while $macro-$ treats each category equally.

[5] http://trec.nist.gov/data/tweets/
[6] http://www.google.com/trends/

Twitter		Sina Weibo	
Total	16935	Total	15811
Sports	2720	Sports	2602
Entertainment	2816	Movies	2694
Business	2912	Games	2605
Science&Tech	2827	Science&Tech	2647
Politics	2937	Politics	2654
Education	2723	Music	2609

Table 2: The distribution of different categories over two datasets.

Twitter				Sina Weibo			
Category	*Precision*	*Recall*	*F*1	*Category*	*Precision*	*Recall*	*F*1
Sports	**0.9322**	0.9483	**0.9402**	Sports	**0.9318**	0.8747	**0.9023**
Entertainment	0.9000	0.5625	0.6923	Movies	0.8848	0.8207	0.8515
Business	0.8043	0.5323	0.6382	Games	0.8090	0.9283	0.8646
Science&Tech	0.6937	**0.9801**	0.8124	Science&Tech	0.8688	0.8323	0.8502
Politics	0.9096	0.9640	0.9360	Politics	0.8661	**0.9324**	0.8980
Education	0.5000	0.5519	0.5165	Music	0.8819	0.8699	0.8759
Micro-average	0.7979	0.7979	0.7979	Micro-average	0.8798	0.8798	0.8798
Macro-average	0.7934	0.6043	0.6128	Macro-average	0.8737	0.8764	0.8738

Table 3: Performance of SSBN model on two datasets with 5% training data and 95% testing data, respectively.

3.2 On Classification Performance Analysis

We first conducted experiment to evaluate the effectiveness of our proposed SSBN model on two datasets. Table 3 displays the average performance in terms of different metrics. Here the parameters are set as $\alpha = 0.5$, $\beta = 0.9$, $\lambda = 0.4$ for Twitter and $\alpha = 0.9$, $\beta = 0.9$, $\lambda = 0.3$ for Sina Weibo, respectively. The parameters selection will be introduced later.

It is observed that our proposed scheme achieves promising precision, recall and $F1$ scores despite of limited availability of labeled data. For twitter dataset, most of the categories achieve precision score higher than 0.85, and the best precision score is up to 0.93 (sports). Half of the categories obtain good results in terms of recall and F1, higher than 0.94 and 0.83, respectively. Our approach yields significant performance over the dataset with pseudo ground truths. This demonstrates the robustness of our method to noisy data. When it comes to Sina Weibo, all the categories achieve remarkable performance of greater than 0.80 across all evaluating metrics. This observation verifies that our method is more stable in less training data. However, our method fails for certain categories such as the Business and Education categories in Twitter dataset. This poor performance mainly comes from the unreliable pseudo ground truths. "Business" and "Education" frequently broaden to various sub-topics. Therefore, the messages retrieved by these types of queries are not internal coherent, at least not as strong as others' categories, even they are assumed to belong to the same category. The unreliable pseudo ground truths bring unpredictable noise to our model.

3.3 On Classification Performance Comparison

To demonstrate the effectiveness of our proposed approach, we compare it against the following the state-of-the-art classifying methods (Phyu, 2009) (Kotsiantis, 2007):

Classifier	*Accuracy*	*MicroP*	*MicroR*	*MicroF1*	*MacroP*	*MacroR*	*MacroF1*
SSBN	**0.8875**	**0.8875**	**0.8875**	**0.8875**	0.8282	0.7627	0.7845
SVM	0.8670	0.8670	0.8670	0.8670	0.8768	0.7611	**0.7860**
NB	0.8722	0.8696	0.8722	0.8722	**0.8879**	0.7329	0.7587
KNN	0.7268	0.7268	0.7268	0.7268	0.6721	0.6471	0.6516
Rocchio	0.8180	0.8204	0.8180	0.8192	0.7361	**0.8384**	0.7605
L-LDA	0.8605	0.8605	0.8605	0.8605	0.8467	0.7223	0.7532

Table 4: Performance comparison among SSBN and other supervised baseline methods on twitter with 90% training data.

Classifier	*Accuracy*	*MicroP*	*MicroR*	*MicroF1*	*MacroP*	*MacroR*	*MacroF1*
SSBN	**0.9020**	**0.9020**	**0.9020**	**0.9020**	0.8976	**0.9045**	**0.9004**
SVM	0.8991	0.8991	0.8991	0.8991	**0.9017**	0.8971	0.8991
NB	0.9015	0.9015	0.9015	0.9015	0.8990	0.9024	0.9003
KNN	0.8565	0.8565	0.8565	0.8565	0.8589	0.8486	0.8526
Rocchio	0.8802	0.8803	0.8802	0.8802	0.8769	0.8832	0.8781
L-LDA	0.8905	0.8905	0.8905	0.8905	0.8876	0.8989	0.8932

Table 5: Performance comparison among SSBN and other supervised baseline methods on Sina Weibo with 90% training data.

- **SVM** (Cortes and Vapnik, 1995) is a supervised learning method. In our experiment, we use an open source package LIBSVM[7] with linear kernel function as baseline.
- **Naive Bayesian** (NB) is a simple probabilistic classifier by applying Bayesian theorem with strong independence assumptions. We use a multi-nomial naive bayesian classifier in our experiment (Yang and Pederson, 1997).
- **K Nearest Neighbors** (KNN) clusters objects based on the closest training examples in the feature space (Creecy et al., 1992). An unlabeled message is assigning the label which is most frequent among the K training samples nearest to the message.
- **Rocchio** (Schapire et al., 1998) is a variant of the Vector Space Model. The average of the relevant documents is viewed as the centroid of the "class".
- **Labeled LDA** (L-LDA) incorporates supervision by constraining LDA model to use only those topics that correspond to an observed label set (Ramage et al., 2009).
- **Transductive SVM** (Trans-SVM) is a semi-supervised SVM method. We extend the binary Transductive SVM in svm-light (Joachims, 1999) to multi-class classifier by incorporating one-against-all strategy.
- **Semi-Naive Bayesian classifiers** (Semi-NB) is a famous semi-supervised text classification method (Nigam et al., 2000). We employ it by using only unlabeled microblogging messages as a prior.

For each aforementioned approaches, the involved parameters are carefully tuned, and the parameters with best performance are used to report the final comparison results. In addition, the same underlying features are utilized for approaches learning. To be fair, our proposed SSBN model was trained with up to 90% data compared with supervised methods, while only 5% training data when compared with semi-supervised approaches. Here, the values of the parameters in SSBN model are set as $\alpha = 0.5, \beta = 0.9$, $\lambda = 0.4$ for Twitter dataset and $\alpha = 0.9$, $\beta = 0.9$, $\lambda = 0.3$ for Sina Weibo dataset.

[7] `http://www.csie.ntu.edu.tw/~cjlin/libsvm/`

$Classifier$	$Accuracy$	$MicroP$	$MicroR$	$MicroF1$	$MacroP$	$MacroR$	$MacroF1$
SSBN	**0.7979**	**0.7979**	**0.7979**	**0.7979**	**0.7934**	**0.6043**	**0.6128**
Trans-SVM	0.6707	0.6707	0.6707	0.6707	0.6602	0.5108	0.4491
Semi-NB	0.7156	0.7156	0.7156	0.7156	0.7308	0.5653	0.549

Table 6: Performance comparison among SSBN and other semi-supervised baseline methods on Twitter with 5% training data.

$Classifier$	$Accuracy$	$MicroP$	$MicroR$	$MicroF1$	$MacroP$	$MacroR$	$MacroF1$
SSBN	**0.8798**	**0.8798**	**0.8798**	**0.8798**	**0.8737**	**0.8764**	**0.8738**
Trans-SVM	0.8084	0.8084	0.8084	0.8084	0.8049	0.8085	0.8052
Semi-NB	0.8198	0.8198	0.8198	0.8198	0.8225	0.8217	0.8204

Table 7: Performance comparison among SSBN and other semi-supervised baseline methods on Sina Weibo with 5% training data.

The comparison results with supervised methods on two datasets are illustrated in Table 4 and Table 5, respectively. It is observed from the tables that our proposed model in general performs better than SVM, NB and L-LDA, and remarkably better than KNN and Rocchio. Even the performance of our method for $MacroP$, $MacroR$ and $MacroF1$ on Twitter and $MacroP$ on Sina Weibo does not achieve the best results, they are still comparable and convincing. Table 6 and Table 7 respectively display the comparison results with semi-supervised methods on two datasets, using 5% as training data. It can be observed that our proposed approach are consistently and significantly better than the current publicly disclosed the state-of-the-art semi-supervised algorithms, across various evaluating metrics. This comprehensive improvements are due to the facts that the integrated external knowledge enriches the message representation and the leveraging intrinsic information detected from abundant unlabeled data enhances the prediction accuracy.

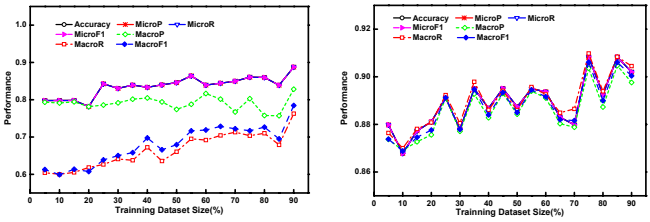
3.4 On the Sensitivity of Training Data Size and Convergence Analysis

In this section, we conduct experiments to investigate the influence of training data size on the overall performance. We progressively increase the size of training corpus at step size of 10%. The experimental results on Twitter and Sina Weibo are respectively illustrated in Figures 3a and 3b. It is observed that the overall trend is upwards along with increasing training set. This is coherent and consistent with our common sense. Also, it is observed that a smaller training set size still produces a robust model on less noisy dataset, with greater than 87% on Sina Weibo.

Perplexity, which is widely used in the topic modeling fields to analyze the convergence of a model (Blei et al., 2003) (Zhao et al., 2010). We do perplexity comparison of SSBN and L-LDA on the testing data when parameters in SSBN model are set as $\alpha = 0.5, \beta = 0.9$, $\lambda = 0.4$ for Twitter and $\alpha = 0.9$, $\beta = 0.9$, $\lambda = 0.3$ for Sina Weibo dataset. Compared with L-LDA model, SSBN model has a lower perplexity value, which means that the words are less surprising to SSBN model, and SSBN model has a powerful predication than L-LDA model.

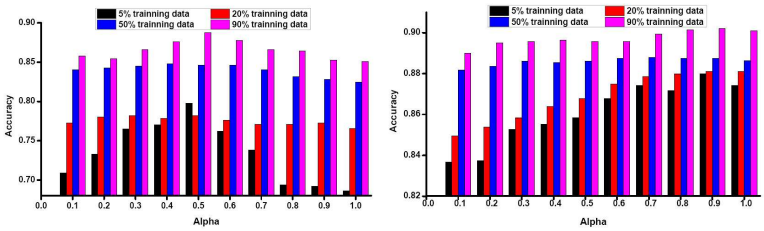
3.5 On the Sensitivity of Parameters

Parameters of α, β and λ are important in our method. In this subsection, we further conduct experiments to study the effect of these parameters. A grid search is performed to select the optimal parameter values.



(a) Twitter (b) Sina Weibo

Figure 3: Performance sensitivity of training set size on Twitter and Sina Weibo.



(a) Twitter (b) Sina Weibo

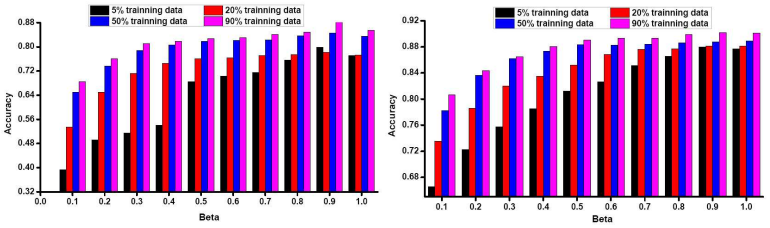
Figure 4: The Performance with varying α and training data size when other parameters are fixed.

3.5.1 Effect of Parameter α

The trade-off parameter α is used to balance the effects of two kinds of prior knowledge at category level: microblogging data collection and external resources. A larger α indicates that more information is preserved from our data collection into the category distribution. A smaller α means that the cues mined from external resources play a dominant role in our model. Figure 4 illustrates the average performance with various α and training collection size on two different datasets. It is observed that the performance increases with the gradual increase of α, and arrives at a peak at certain α, then the performance decreases. This result reflects that an optimal performance comes from an appropriate combination of external and internal resources, rather than pure individual knowledge. Also it verifies that the incorporation of Google resources has been proven useful. Empirical optimal value of α is within $[0.5, 1]$.

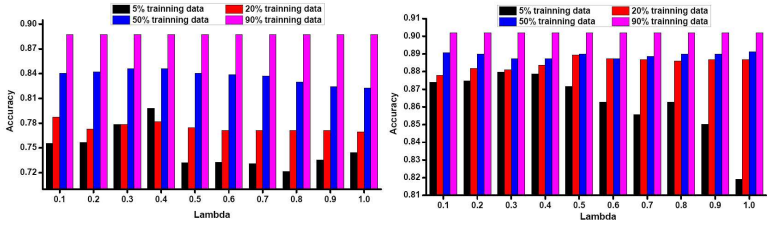
3.5.2 Effect of Parameter β

There are two category-word distributions, θ' and ϕ', which are respectively generated from our data collection and google search results; and parameter β is utilized to adjust the contribution between these two different resources in category-word level. Larger β implies larger likelihood a word is generated from θ'. The effects of parameter β on Twitter and Sina Weibo are shown in Figure 5. It is clearly observed that larger values of β frequently lead to higher accuracies with different training set sizes, and the accuracy reaches peak value when β locates at 0.9. However, when β trends to 1, the performance slightly decreases. Empirical optimal value of β is within $[0.5, 1]$.



(a) Twitter (b) Sina Weibo

Figure 5: The Performance with varying β and training data size when other parameters are fixed.



(a) Twitter (b) Sina Weibo

Figure 6: The Performance with varying λ and training data size when other parameters are fixed.

3.5.3 Effect of Parameter λ

λ indicates the contribution from unlabeled data points, between 0 and 1. When λ is close to 1, knowledge from unlabeled data is considered as important as labeled data. On the other hand, when λ at near-zero value, our model approaches a supervised learning algorithm. The results are illustrated in Figure 6, from which we observe some insights: (1) varying λ has little impact on average accuracy for a large training set, such as 50 percent as training set, especially for 90 percent as training set; (2) the best accuracy occurs at $\lambda = 0.4$ and $\lambda = 0.3$ respectively for Twitter and Sina Weibo, and then drops down quickly, which illustrates unlabeled data could give some feedback to improve classification performance. Empirical optimal value of λ is within $[0.3, 0.5]$.

4 Related Work

The task of topic classification of microblogging messages is to assign the pre-defined class labels to unlabeled messages given a collection of messages. It has been demonstrated to be a fundamental task for many applications, such as query disambiguation (Teevan et al., 2011), location prediction (Gao et al., 2012) and hot topic tracking (Weng and Lee, 2011), *etc*. To the best of our knowledge, our work is the first attempt to utilize semi-supervised learning methods to classify microblogging messages. There are, however, several lines of related work.

The significance of topic models has been exploited in microblog clustering and classification. A representative work was proposed in 2010 (Hong and Davison, 2010), where latent dirichlet allocation (LDA) (Blei et al., 2003) and author-topic model (Rosen-Zvi et al., 2010) were deeply investigated to automatically find hidden topic structures on Twitter. Following that, Zhao et al. (2011) performed content analysis through Twitter-LDA modeling on a Twitter corpus collected within a three month span. Several variants of LDA to incorporate supervision have been proposed

by Ramage et al. (2009, 2010), and have been shown to be competitive with strong baselines in the microblogging environment. Although these LDA-based topic model greatly save cognitive and physical effort required from user interaction, their performances are usually not very satisfactory. The main reason is due to the sparsity of short informal messages that makes similarity comparison difficult. Different from previous models, we employed a two-step pre-processing: detecting informal words using dictionary and correcting the words into formal ones. This helps to alleviate the negative effects brought by short message sparsity to some extent.

Lee et al. (2011) classified tweets into pre-defined categories such as sports, technology, politics, *etc*. Instead of topic models, they constructed word vectors with tf-idf weights and utilized a Naive Bayesian Multinomial classifier to classify tweets. Further, Support Vector Machines achieved good performance to classify Twitter messages, as reported by Zubiaga et al. (2011). Sriram et al. (2010) proposed to use a small set of domain-specific features extracted from the author's profile and text to represent short messages. Their method, however, requires extensive pre-processing to conduct effectively feature analysis, which was impractical to as a general solution for classification of microblogging messages. The performance improvement of the supervised methods mainly depend on a large scale of labeled training data, which is laborious and time consuming. Further, the sparsity problem hinders significant performance improvement. To break the current impasse between annotation cost and effectiveness, we proposed to utilize semi-supervised learning methods. We trained a semi-supervised classifier by using the large amount of unlabeled data, together with labeled data. In addition, our work is novel in that we mined the information cues from Google Search Engine and seamlessly fused them with informal microblogging messages.

5 Conclusion and Future Work

In this paper, we proposed a novel scheme to classify microblogging messages, which addresses three concerns in microblog classifications. First, the incorporation of external resources to supplement the short microblogs well compensates the data sparseness issue. Second, the semi-supervised classifier seamlessly fuse labeled data structure and external resources into the training process, which reduced the requirement for manually labeling to a certain degree. Third, we model the category probability of a given message based on the category-word distribution, and this successfully avoided the difficulty brought about by the spelling errors that are common in microblogging messages. We proposed a semi-supervised learning approach to classify microblogging messages, and the experimental results demonstrated its effectiveness as compared to existing the state-of-the-art methods, as well as practically extension to large-scale dataset.

This work suggests some interesting directions for further exploration. It is interesting to explore whether: (1) the incorporation of social network structure can improve the performance of microblogging classification (Hu and Liu, 2012a); (2) the use of external resources such as Wikipedia and WordNet might be valuable for understanding microblogging messages; and (3) the provision of category summarization can help to organize microblogging messages.

Acknowledgments

This research was supported by the National Natural Science Foundation of China (grant number 61170189, 60973105), the National Natural Science Fund for Young Scholar (grant number 61202239), the Research Fund for the Doctoral Program of Higher Education (grant number 20111102130003), and the Fund of the State Key Laboratory of Software Development Environment (grant number SKLSDE-2011ZX-03),

References

Blei, D. M., Ng, A. Y., and Jordan, M. I. (2003). Latent dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022.

Cortes, C. and Vapnik, V. (1995). Support-vector networks. *Machine Learning*, 20:273–297.

Creecy, R. H., Masand, B. M., Smith, S. J., and Waltz, D. L. (1992). Trading mips and memory for knowledge engineering. In *Communication of the ACM*, volume 35, pages 48–64.

Diao, Q., Jiang, J., Zhu, F., and Lim, E.-P. (2012). Finding bursty topics from microblogs. In *Proceedings of Association for Computational Linguistics*.

Gao, H., Tang, J., and Liu, H. (2012). Exploring social-historical ties on location-based social networks. In *Proceedings of International AAAI Conference on Weblogs and Social Media*.

Gruber, A., Rosen-Zvi, M., and Weiss, Y. (2007). Hidden topic markov model. In *Proceedings of International Conference on Artificial Intelligence and Statistics*.

Hong, L. and Davison, B. D. (2010). Empirical study of topic modeling in twitter. In *Proceedings of KDD Workshop on Social Media Analytics*.

Hu, X. and Liu, H. (2012a). Social status and role analysis of palin's email network. In *Proceedings of the international conference companion on World Wide Web*.

Hu, X. and Liu, H. (2012b). Text analytics in social media. *Mining Text Data*, pages 385–414.

Hu, X., Sun, N., Zhang, C., and Chua, T.-S. (2009). Exploiting internal and external semantics for the clustering of short texts using world knowledge. In *Proceedings of the ACM conference on Information and knowledge management*.

Joachims, T. (1999). Transductive inference for text classification using support vector machines. In *Proceedings of International Conference on Machine Learning*.

Kotsiantis, S. B. (2007). Supervised machine learning: A review of classification techniques. *Informatica*, 31:249–268.

Lee, K., Palsetia, D., Narayanan, R., Patwary, M. M. A., Agrawal, A., and Choudhary, A. (2011). Twitter trending topic classification. In *Proceedings of ICDM Workshop on Optimization Based Methods for Emerging Data Mining Problems*.

Nie, L., Wang, M., Zha, Z.-j., Li, G., and Chua, T.-S. (2011). Multimedia answering: enriching text qa with media information. In *Proceedings of Annual ACM Conference on Special Interest Group on Information Retrieval*.

Nigam, K., Mccallum, A. K., Thrun, S., and Mitchell, T. (2000). Text classification from labeled and unlabeled documents using EM. In *Machine Learning - Special issue on information retrieval*, volume 39, pages 103–134.

Phyu, N. P. (2009). Survey of classification techniques in data mining. In *Proceedings of International MultiConference of Engineers and Computer Scientists*.

Ramage, D., Dumais, S., and Liebling, D. (2010). Charaterizing microblog with topic models. In *Proceedings of International AAAI Conference on Weblogs and Social Media*.

Ramage, D., Hall, D., Nallapati, R., and Manning, C. D. (2009). Labeled LDA: a supervised topic model for credit attribution in multi-labeled corpora. In *Proceedings of International Conference on Empirical Methods in Natural Language Processing*.

Rosa, K. D., Shah, R., Lin, B., Gershman, A., and Frederking, R. (2011). Topical clustering of tweets. In *Proceedings of SIGIR Workshop on Social Web Search and Mining*.

Rosen-Zvi, M., Chemudugunta, C., Griffiths, T., Smyth, P., and Steyvers, M. (2010). Learning author-topic models from text corpora. *ACM Transactions on Information Systems*, 28:1–38.

Schapire, R. E., Singer, Y., and Singhal, A. (1998). Boosting and Rocchio applied to text filtering. In *Proceedings of Annual ACM Conference on Research and Development in Information Retrieval*, pages 215–223.

Sokolova, M. and Lapalme, G. (2009). A systematic analysis of performance measures for classification tasks. *Information Processing and Management*, 45:427–437.

Sriram, B., Fuhry, D., Demir, E., Ferhatosmanoglu, H., and Demirbas, M. (2010). Short text classification in twitter to improve information filtering. In *Proceedings of Annual ACM Conference on Research and Development in Information Retrieval*.

Tang, J., Wang, X., Gao, H., Hu, X., and Liu, H. (2012). Enriching short text representation in microblog for clustering. *Frontiers of Computer Science in China*, 6(1):88–101.

Teevan, J., Ramage, D., and Morris, M. R. (2011). #twittersearch: a comparison of microblog search and web search. In *Proceedings of ACM Conference on Web Search and Data Mining*.

Weng, J. and Lee, B.-S. (2011). Event detection in twitter. In *Proceedings of Association for the Advancement of Artificial Intelligence*.

Yang, Y. and Pederson, J. (1997). Feature selection in statistical learning of text categorization. In *Proceedings of International Conference on Machine Learning*.

Zhao, T., Li, C., Ding, Q., and Li, L. (2010). User-sentiment topic model: refining user's topics with sentiment information. In *Proceedings of ACM SIGKDD Workshop on Mining Data Semantics*.

Zhao, W. X., Jiang, J., Weng, J., He, J., Lim, E.-P., Yan, H., and Li, X. (2011). Comparing twitter and traditional media using topic models. In *Proceedings of European Conference on IR Research*.

Zubiaga, A., Spina, D., Fresno, V., and Martinez, R. (2011). Classfying trending topics: A typology of conversation triggers on twitter. In *Proceedings of ACM Conference on Information and Knowledge Management*.

A System For Multilingual Sentiment Learning On Large Data Sets

Alex CHENG[1] *Oles ZHULYN*[1]
(1) Department of Computer Science, University of Toronto, Canada
hyc@cs.toronto.edu, oles@cs.toronto.edu

Abstract
Classifying documents according to the sentiment they convey (whether positive or negative) is an important problem in computational linguistics. There has not been much work done in this area on general techniques that can be applied effectively to multiple languages, nor have very large data sets been used in empirical studies of sentiment classifiers.
We present an empirical study of the effectiveness of several sentiment classification algorithms when applied to nine languages (including Germanic, Romance, and East Asian languages). The algorithms are implemented as part of a system that can be applied to multilingual data. We trained and tested the system on a data set that is substantially larger than that typically encountered in the literature. We also consider a generalization of the n-gram model and a variant that reduces memory consumption, and evaluate their effectiveness.

Keywords: sentiment, classification, multilingual, empirical verification.

Proceedings of COLING 2012: Technical Papers, pages 577–592,
COLING 2012, Mumbai, December 2012.

1 Introduction

Classifying text documents according to the sentiment they convey is an important problem in computational linguistics. Sentiment reflects the emotional content in the document or the attitude of the speaker to the subject matter in the document, and can be positive or negative. For example, "Thank you for the pleasant time we spent together" conveys a positive sentiment, while "I was devastated when you left" conveys a negative sentiment.

Sentiment classifiers that can process massive amounts of data quickly and accurately have applications in many segments of society. Marketing and brand management firms that are interested in how consumers generally feel about particular companies and their products can apply sentiment classifiers to social media documents containing relevant keywords. Government agencies that monitor electronic communications in order to identify and locate dissidents can use sentiment classifiers to find subversive messages.

To the best of our knowledge, there has not been much work done in this area on general techniques that can be applied effectively to multiple languages, nor have very large data sets been used in empirical studies of sentiment classifiers. In this paper, we present an empirical study of two sentiment classification algorithms applied to nine languages (including Germanic, Romance, and East Asian languages). One of these algorithms is a naive Bayes classifier, and the other is an algorithm that boosts a naive Bayes classifier with a logistic regression classifier, using majority vote. These algorithms are implemented as part of a system that can be applied to multilingual data. Our implementation is fast, allowing a large number of documents to be classified in a short amount of time, with high accuracy.

Automatic sentiment classification of text documents requires that the documents be modeled in a way that is amenable to the algorithm being used. The typical approach is to model the documents using n-grams. In this paper, we consider a generalization of the n-gram model that is more suitable for languages with a flexible word order, and a variant of this generalized n-gram model that helps reduce memory consumption. These models are built into our system.

For the empirical study, we trained and tested our system on a data set that is substantially larger than that typically encountered in the literature. To generate this data set, we wrote custom crawlers, and mined various web sites for reviews of products and services. The reviews were annotated by their authors with star ratings, which we used to automatically label the reviews as conveying either a positive or a negative sentiment. For each experiment in the study, we sampled disjoint training and testing sets uniformly at random from this large data set. Unlike the usual approach in the literature, the testing sets were much larger than the training sets (at least four times larger), and the experiments were repeated many times. We did this to ensure that our results were statistically significant.

The paper is organized as follows. In Section 2, we provide a brief overview of related work done in this area. In Section 3, we describe our large data set and how we acquired it. In Section 4, we discuss the generalization of the n-gram model and its variant. In Section 5, we describe the sentiment classification algorithms that we considered. In Section 6, we describe our experimental setup, and present the results. We then conclude and suggest future directions for this work.

2 Related Work

Pang and Lee (Pang and Lee, 2008) have written an excellent survey on the work done in the area of sentiment classification.

Pang et al. (Pang et al., 2002) undertook an empirical study that resembles our own. They evaluated the effectiveness of several machine learning methods (naive Bayes (Domingos and Pazzani, 1997; Lewis, 1998), maximum entropy (Csiszár, 1996; Nigam et al., 1999), and support vector machines (Cortes and Vapnik, 1995; Joachims, 1998)) for sentiment classification of English-language documents. They generated their data set by mining movie reviews from the Internet Movie Database (IMDb)[1] and classifying them as positive or negative based on the author ratings expressed with stars or numerical values. They modeled the movie reviews as n-grams.

Bespalov et al. (Bespalov et al., 2011) presented a method for classifying the sentiment of English-language documents modeled as high-order n-grams that are projected into a low-dimensional latent semantic space using a multi-layered "deep" neural network (Bengio et al., 2003; Collobert and Weston, 2008). They evaluated the effectiveness of this method by comparing it to ones based on perceptrons (Rosenblatt, 1957) and support vector machines. Their data set was derived from reviews on Amazon[2] and TripAdvisor[3], which were labeled as positive or negative based on their star ratings.

3 Large Data Set

Our large data set consists of reviews of products and services mined from various web sites. We wrote custom crawlers for each of these web sites. The domain for the reviews is quite diverse, including such things as books, hotels, restaurants, electronic equipment, and baby care products. We only looked at web sites where the reviews were accompanied by star ratings (which we normalized to a scale between 1- and 5-stars). This enabled us to automatically assign a sentiment to each review.

We considered reviews accompanied by a rating of 1- or 2-stars as having a negative sentiment, and those accompanied by 5-stars as having a positive sentiment. For some of the web sites (e.g. Ciao![4]), along with the star ratings, the reviews were also accompanied by a binary (recommended or not-recommended) rating. In this case, we assigned a negative sentiment to reviews accompanied by a rating of 1- or 2-stars, and a not-recommended rating, and a positive sentiment to reviews accompanied by a rating of 5-stars, and a recommended rating.

The approach of automatically assigning sentiment to reviews based on accompanying author ratings has precedents in the literature (Pang et al., 2002; Bespalov et al., 2011). Although it is likely that there is some noise in the data with this kind of approach, an automated approach is nevertheless essential for generating a large data set.

The data for English, French, Spanish, Italian, and German was mined from Amazon, Ciao!,

[1] http://reviews.imdb.com/Reviews/
[2] http://www.amazon.com
[3] http://www.tripadvisor.com
[4] http://www.ciao.com

Language	Negative	Positive
Japanese	1111584	8497266
English	459837	2442952
German	297028	1654456
Chinese	155221	1332076
French	146016	685136
Italian	115744	432726
Spanish	69065	272788
Dutch	34596	215586
Portuguese	20507	97759

Table 1: Number of negative and positive documents for each language in our data set

and TripAdvisor. The Portuguese data was mined from Walmart[5], Opinaki[6], Buscapé[7], and TripAdvisor. The Dutch data was mined from bol.com[8], Ciao!, and TripAdvisor. The Chinese data was mined from Amazon, dangdang.com[9], and TripAdvisor. The Japanese data was mined from Amazon, Rakuten[10], and Kakaku.com[11]. Across these web sites, these languages are not equally well-represented. As a consequence, for some of the languages (e.g. Japanese) we were able to mine substantially more data than for others (e.g. Portuguese) (Table 1).

4 Document Representation

A text document is a sequence of tokens. Tokens can simply be single characters within the text document. However, in sentiment classification, the tokens of interest are typically n-grams, which are n-length sequences of contiguous whitespace-separated words. For example, if a document is the sequence $(w_1, w_2, \ldots, w_{N-1}, w_N)$, where $w_2, \ldots, w_{N-1}$ are whitespace-separated words, and w_1 and w_N are the special symbol `<BOUNDARY>`, signifying the beginning or the end of the document, then the 2-grams are $(w_1, w_2), (w_2, w_3), (w_3, w_4), \ldots, (w_{N-2}, w_{N-1}), (w_{N-1}, w_N)$.

In Chinese and Japanese, words are not delimited by whitespace in writing. For the results we present in this paper, we used third-party libraries (Taketa, 2012; Lin, 2012) to segment Chinese and Japanese documents into words. These libraries are based on machine learning methods, and do not require large dictionary files. Nie et al. (Nie et al., 2000) considered tokenizing Chinese documents as n-grams. We also experimented with this approach for both Chinese and Japanese documents (i.e. we treated single characters as tokens). Although we do not present them here, the results we achieved in these experiments were comparable to (though not quite as good as) the results we achieved with the third-party libraries.

[5] http://www.walmart.com.br
[6] http://www.opinaki.com.br
[7] http://www.buscape.com.br
[8] http://www.bol.com
[9] http://www.dangdang.com
[10] http://www.rakuten.co.jp
[11] http://www.kakaku.com

Figure 1: English 2-grams most indicative of positive sentiment.

4.1 Generalized N-gram

We can generalize the n-gram model by introducing a window size $k \geq n$. To iterate over all the tokens in a sequence, we first consider every window in the sequence (that is, every contiguous subsequence of length k). The tokens are all the (not necessarily contiguous) subsequences of length n within each window. When $k = n$, this is just the standard n-gram model. Guthrie et al. (Guthrie et al., 2006) refer to this as the skip-gram model.

This model is suitable for languages with a flexible word order (e.g. German). With a flexible word order, the co-occurrence of several specific words in proximity may be indicative of a particular sentiment irrespective of any intermediary words. In the standard n-gram model, the relevant words can only be captured in a token along with the intermediary words. Due to the potential variety in the intermediary words, a single document may contain many tokens that are different, but that all correspond to the co-occurrence of these relevant words. In contrast, the generalized n-gram model enables these relevant words to be captured in a single token. This helps to mitigate against noise. However, for a given document, the generalized n-gram model requires that more tokens are processed than does the standard n-gram model.

4.2 Hitting N-gram

The hitting n-gram model is a variation on the generalized n-gram model. In the hitting n-gram model, only the windows that are centered around (i.e. "hit") words from a predefined lexicon are considered. We can specify where inside a window we would like the hit to occur by giving the window size in terms of the number of words preceding a word from the lexicon and the number of words following that word from the lexicon.

不错，很好，很喜欢，值得，喜欢，很不错，很有，非常好，实用，适合，孩子，学习，满意，很，好书，赞，非常，挺好，很快，帮助，实惠，舒服，了解，很满，呵呵，方便，本书，生活，哈哈，儿子，这本，老师，爱，推荐，很漂亮，受益匪浅，精美，很精，可爱，全面，划算，经典，详细，感动，超值，很棒，值得一看，丰富，力，慢慢，漂亮，不过，支持，很给，一本，世界，有趣，她，拥有，合适，知识，阅读，好用，挺，收藏，感谢，幸福，更好，爱不释手，小巧，最喜欢，成长，强烈推荐，通俗易懂，每天，好看，推荐给，历史，就到，挺不错，棒，开心，常值，一口气，思考，对于，朋友，快乐，物超所值	失望，没有，退货，根本，不好，太，差，不是，了，很不，太差，一般，不要，就，很差，结果，不知道，不，不能，卓越，不如，怎么，客服，没，都没有，只能，发现，后悔，垃圾，怀疑，郁闷，坏了，换货，页，是不是，居然，不值，麻烦，吗，盗版，打开，粗糙，买，问题，个，而且，没什么，什么，为什么，打电话，电话，严重，可是，才，竟然，块，无语，建议，不行，你们，我，就不，浪费，换，实在，都不，完全，算了，只有，烂，不满意，也没有，不了，不值得，坏，次，联系，两，啊，掉了，本来，说，不舒服，明显，钱，几，不喜欢，无法，本就，不到，售后，商品，换了，一点，不够，点，产品，上，房间

Table 2: Top Chinese automatically segmented words most indicative of positive and negative sentiment.

great, love, easy, highly, best, perfect, excellent, amazing, loves, wonderful, favorite, awesome, fantastic, recommend, book, beautiful, perfectly, pleased, sturdy, fits, works, recommended, fun, definitely, life, price, album, comfortable, superb, happy, helps, gives, family, beautifully, brilliant, incredible, loved, classic, makes, glad, fast, delicious, outstanding, allows, easily, little, always, cd, heart, durable, easier, enjoy, unique, provides, truly, beat, favorites, solid, simple, handy, songs, collection, powerful, ease, size, super, greatest, keeps, song, smooth, books, thank, bonus, nicely, brings, friends, amazed, pleasantly, holds, terrific, gift, wonderfully, hooked, read, quick, enjoyed, skeptical, fabulous, thanks, compact, stores, favourite, albums, refreshing, learning, addictive, penny, guitar, gorgeous, sharp, journey, enjoys, lives, colors, joy, compliments, worry, job, versatile, must, every, informative, soft, everyone, daughter, comes, everyday, masterpiece, satisfied, crisp, affordable, fascinating	poor, bad, waste, worst, money, customer, return, disappointed, service, but, refund, told, terrible, returned, nothing, did, unfortunately, hotel, didn't, back, horrible, worse, problem, sent, useless, ok, company, awful, disappointing, off, tried, why, stay, pay, asked, send, should, returning, do, disappointment, poorly, don't, phone, boring, again, staff, said, call, trying, support, guess, maybe, rude, unless, instead, get, seemed, supposed, contacted, paid, wouldn't, fix, went, stopped, thought, avoid, beware, defective, customers, received, sorry, booked, <NUMBER>, broke, manager, wrong, warranty, junk, mistake, wasted, rooms, contact, left, never, doesn't, me, broken, replacement, failed, happened, crap, email, stupid, garbage, annoying, wasn't, least, star, cheap, reviews, months, properly, apparently, weeks, response, checked, working, got, frustrating, stayed, slow, going, hoping, waiting, error, ridiculous, completely, reason, try, either, credit, ended, please, half

Table 3: Top English words most indicative of positive and negative sentiment.

excellent, permet, plaisir, magnifique, livre, découvrir, bonheur, recommande, facile, parfait, très, merveille, superbe, excellente, petit, parfaitement, grâce, agréable, the, régal, of, indispensable, également, grands, petits, facilement, doux, j'adore, délicieux, chansons, conseille, rock, l'album, bémol, idéal, simple, vivement, pouvez, voix, cd, parfaite, meilleur, douceur, n'hésitez, adoré, délice, enfants, rapide, couleurs, bonne, magnifiques, grande, famille, toutes, génial, titres, découvert, pratique, to, pourrez, parfum, belle, adore, must, and, incontournable, aime, recommander, sublime, beauté, superbes, petite, guitare, ouvrage, différentes, mélange, trouverez, bijou, lait, complet, sucre, remarquable, recette, univers, chanson, sel, modération, déguster, super	pas, ne, rien, service, client, me, réponse, disant, mauvaise, j'ai, je, commande, pire, clients, mauvais, demande, aucune, déception, payer, mois, remboursement, impossible, déconseille, téléphone, n'est, dit, qu'ils, sav, mail, été, suis, déc ue, mal, n'a, bref, bout, arnaque, demandé, n'ai, envoyé, eux, décevant, éviter, eu, n'y, problème, commandé, semaines, rec u, aucun, site, rembourser, payé, compte, personne, tard, contrat, chez, erreur, jours, n'était, mails, nul, courrier, déc u, euros, responsable, là, aurait, avons, avoir, commercial, mon, rec ois, médiocre, panne, désagréable, ma, sommes, vente, heureusement, chambre, c a, colis, dû, j'avais, dommage, m'a, d'attente, j'appelle, semaine, retard, répond, n'ont, dossier, voulu, lendemain, pourtant, manque, étaient

Table 4: Top French words most indicative of positive and negative sentiment.

In contrast to the generalized n-gram model, the hitting n-gram model can drastically reduce the number of tokens that need to be processed, depending on the lexicon that is chosen. For this project, we processed our large data set using Pearson's chi-squared test to find the words that are most indicative of positive and negative sentiment to build a lexicon for each language. We discuss this in more detail in Section 5.

5 Classifiers

For our experiments, we modeled documents using the 2-gram model, the generalized 2-gram model with window size 3, the generalized 2-gram model with window size 5, and the hitting 2-gram model with (preceding) window size 1. For each of these models, we trained a naive Bayes classifier and a logistic regression classifier. During testing, we considered the results from the naive Bayes classifier, and the naive Bayes classifier boosted with the logistic regression classifier using majority vote. We repeated this for each language.

5.1 Hitting 2-gram Model

Yang and Pedersen (Yang and Pedersen, 1997) evaluated several automatic methods for selecting features that were useful for categorizing text. Pearson's chi-squared test proved

to be the most effective. We used Pearson's chi-squared test to find, for each language, the top 200 words most indicative of positive sentiment and the top 200 words most indicative of negative sentiment, without filtering for stop words (e.g. Table 2, Table 3, and Table 4). We used these words as the lexicon for the hitting 2-gram model.

Following Yang and Pedersen, we computed, for each word w and each sentiment s, the goodness of fit measure:

$$\chi^2(w,s) = \frac{N \times (AD - CB)^2}{(A+C) \times (B+D) \times (A+B) \times (C+D)}$$

where A is the number of documents with sentiment s in which w occurs, B is the number of documents without sentiment s in which w occurs, C is the number of documents with sentiment s in which w does not occur, D is the number of documents without sentiment s in which w does not occur, and N is the total number of documents. We did this once over our entire data set, and took the words that scored highest according to this measure.

In our experiments, we set the window size to be 1 preceding word. We also tried other window sizes, but they did not produce substantially better results. We do not report these other results.

The technique we used to build the lexicon can be applied to other kinds of tokens. For example, Figure 1 is a word cloud of the English 2-grams most indicative of positive sentiment in our data set. We generated the word cloud using Wordle (Feinberg, 2012).

5.2 Naive Bayes Classifier

For the 2-gram model, we used the training data to compute for each 2-gram, $(\mathscr{W}, \mathscr{W}')$, the probability that it belongs to a document with a positive sentiment, $\mathscr{P}_{pos}(\mathscr{W}, \mathscr{W}')$, and the probability that it belongs to a document with a negative sentiment, $\mathscr{P}_{neg}(\mathscr{W}, \mathscr{W}')$. Given a document $(\mathscr{W}_1, \mathscr{W}_2, \ldots, \mathscr{W}_{N-1}, \mathscr{W}_N)$[12] to classify, we apply a decision rule based on the ratio

$$\prod_i \frac{\mathscr{P}_{pos}(\mathscr{W}_i, \mathscr{W}_{i-1})}{\mathscr{P}_{neg}(\mathscr{W}_i, \mathscr{W}_{i-1})}$$

computed using the probabilities determined from our training data. If this ratio is greater than 1, then we classify the document as positive. Otherwise, we classify the document as negative.

The following derivation show what this ratio means.

$$\prod_i \frac{\mathscr{P}_{pos}(\mathscr{W}_i, \mathscr{W}_{i-1})}{\mathscr{P}_{neg}(\mathscr{W}_i, \mathscr{W}_{i-1})} = \prod_i \frac{\mathscr{P}_{pos}(\mathscr{W}_i | \mathscr{W}_{i-1})}{\mathscr{P}_{neg}(\mathscr{W}_i | \mathscr{W}_{i-1})} \times \frac{\mathscr{P}_{pos}(\mathscr{W}_{i-1})}{\mathscr{P}_{neg}(\mathscr{W}_{i-1})} \tag{1}$$

$$= \prod_i \frac{\mathscr{P}_{pos}(\mathscr{W}_i | \mathscr{W}_{i-1})}{\mathscr{P}_{neg}(\mathscr{W}_i | \mathscr{W}_{i-1})} \times \prod_i \frac{\mathscr{P}_{pos}(\mathscr{W}_{i-1})}{\mathscr{P}_{neg}(\mathscr{W}_{i-1})} \tag{2}$$

$$= \prod_i \frac{\mathscr{P}_{pos}(\mathscr{W}_i | \mathscr{W}_{i-1})}{\mathscr{P}_{neg}(\mathscr{W}_i | \mathscr{W}_{i-1})} \times \prod_i \frac{\mathscr{P}_{pos}(\mathscr{W}_i)}{\mathscr{P}_{neg}(\mathscr{W}_i)} \tag{3}$$

[12] $\mathscr{W}_2, \ldots, \mathscr{W}_{N-1}$ are whitespace-separated words, and $\mathscr{W}_1$ and $\mathscr{W}_N$ are the special symbol `<BOUNDARY>`, signifying the beginning or the end of the document.

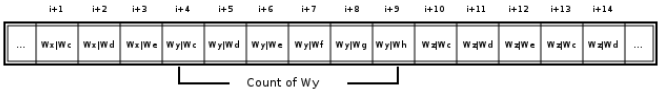


Figure 2: Sum over the range to get the count for W_y.

Line (1) follows from the definition of conditional probability. Line (2) follows from commutativity and associativity of multiplication. Line (3) follows from the fact that the missing term

$$\frac{\mathscr{P}_{pos}(\mathscr{W}_N)}{\mathscr{P}_{neg}(\mathscr{W}_N)} = 1$$

since the occurrence of $\mathscr{W}_N$, the special symbol <BOUNDARY>, in a document with a positive sentiment is equally likely to its occurrence in a document with a negative sentiment. The numerator in the expression in line (3) is the probability that the given document has a positive sentiment according to both the 2-gram model and the 1-gram model. The denominator is the probability that the document has a negative sentiment according to both models. Our decision rule classifies the document according to which of these two probabilities is the greater. Notice that our confidence that the sentiment of the document was classified correctly can be increased using a threshold parameter. For example, if the ratio between the numerator and the denominator is very high, then we have high confidence that the document has a positive sentiment. At the cost of leaving some documents unclassified, the threshold parameter can be used to achieve arbitrarily high classification accuracies.

Our implementation allows these values to be computed quickly. We represent each distinct word that we encounter in the training data with a nonnegative 32-bit integer, and use a hash map to store this representation. We represent each 2-gram that we encounter in the training data by packing the two integers corresponding to the two words in the 2-gram in a 64-bit integer. After processing the training data, we sort all the 64-bit integers representing the 2-grams, and store the sorted list in an array. We use the index of each 2-gram in this array as an index into two other arrays: one representing the number of occurrences of the 2-grams in positive documents, and the other representing the number of occurrences of the 2-grams in negative documents. This approach gives us a minimal perfect hash function from 2-grams to their counts in positive and negative documents. Looking up a count for a given 2-gram is fast: binary search on the sorted array gives us the index to the counts for occurrences in positive and negative documents. Our minimal use of pointers also keeps memory consumption low.

One might be interested in computing the probabilities for a document under the 1-gram and 2-gram models. Our implementation allows this to be computed quickly. Given a word, one can perform binary search on the sorted list of 2-grams to find the first occurrence of a 2-gram whose first word is the given word. After this 2-gram is found, one needs only to sum up all the values in the list up to the last occurrence of a 2-gram whose first word is the given word (Figure 2), and divide by the total sum of all the values in the list (which can be computed once, when the list is built). This is the probability for a 1-gram. The

probability for a 2-gram can be evaluated directly from this using Bayes' rule.

The approach we took for the generalized 2-gram models, and the hitting 2-gram model is the same. However, the derivation for the value in our decision rule does not work out exactly, and only gives us a rough approximation of the probabilities. The results of the experiments reflect this fact: although classification speed is very fast, the accuracies are somewhat less impressive than what one might expect.

5.3 Logistic Regression Classifier

We used a logistic regression classifier provided by the `LIBLINEAR` software (Fan et al., 2008). For logistic regression, it is necessary to represent documents as feature vectors. We tried three representations. In all three cases, we had a feature for each token encountered in the training data. For the first representation, the value we used for each feature was the frequency of occurrence of the corresponding token, in the document. We normalized each feature to fall in the range $[0,1]$ (details in the following paragraph). The second representation was like the first, except we normalized the whole vector to the unit vector, instead of normalizing per feature. For the third representation, the value we used for each feature was 1 or 0, depending on whether the corresponding token was present in the document or not. We normalized the whole vector to the unit vector. All three approaches produced similar results. We only report the results for the first representation.

The normalization that we used for the first representation is the following. Suppose $\mathscr{D}$ is the total set of training documents, and $\mathscr{T}$ is the total set of tokens encountered across all documents in $\mathscr{D}$. For each document $d \in \mathscr{D}$ and each token $t \in \mathscr{T}$, let $freq_d(t)$ be the frequency of occurrence of token t in document d (e.g. if d contains 10 tokens and t occurs 5 times in d, then $freq_d(t) = 5/10 = 0.5$). The normalized value $freq'_d(t)$ of $freq_d(t)$ is

$$freq'_d(t) = \frac{freq_d(t) - \min_{d' \in \mathscr{D}}(freq_{d'}(t))}{\max_{d' \in \mathscr{D}}(freq_{d'}(t)) - \min_{d' \in \mathscr{D}}(freq_{d'}(t))}.$$

Notice that if d is a document from the testing set, then $freq'_d(t)$ can fall outside the range $[0,1]$. This is the intended behavior (Fan et al., 2008; Hsu et al., 2010).

For our experiments, we boosted the naive Bayes classifier with the logistic regression classifier using majority vote. If both classifiers agreed, then we returned the value they agreed on. Otherwise, we returned no answer.

6 Experiments

6.1 Experimental Setup

For the empirical study, we evaluated two algorithms: a naive Bayes classifier, and a naive Bayes classifier boosted with a logistic regression classifier, using majority vote. In evaluating each algorithm, we considered four ways of modeling text documents: the 2-gram model (`2g`), the generalized 2-gram model with window size 3 (`2g-w3`), the generalized 2-gram model with window size 5 (`2g-w5`), and the hitting 2-gram model with (preceding) window size 1 (`2g-h`). We repeated this for nine languages: French (`fr`), Spanish (`es`), Italian (`it`), Portuguese (`pt`), Traditional and Simplified Chinese (`zh`), Japanese (`ja`), German (`de`), English (`en`), and Dutch (`nl`). In total, this constitutes 72 different experiments. We ran each experiment 10 times to validate the results.

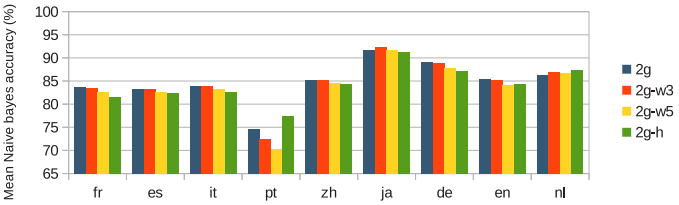


Figure 3: Mean accuracy (in percent, over ten runs) of naive Bayes classifier for each model and each language.

For each of the ten runs and each language, we sampled disjoint training and testing sets uniformly at random from the large data set. We ensured that the testing set was always at least four times larger than the training set. For each way of modeling text documents, we trained each algorithm using the training set, and tested it using the testing set. In Table 8, we report the mean and standard deviation, over ten runs, for the number of positive and negative documents in the training and testing sets for each language.

We performed our experiments using commodity hardware consisting of a quad-core Core 2 (Q9650) processor running at 3.0GHz, 16GB DDR2 memory running at 800MHz, and a 64-bit operating system with Linux kernel version 3.0. Our sentiment classification system was implemented using Java, and we ran it using Oracle Java SE Runtime Environment (build 1.6.0_30-b12). Our system makes use of several third-party libraries. The versions of these that we used are Java `LIBLINEAR` version 1.8 (Waldvogel, 2012), Apache Lucene Core version 3.6.0 (The Apache Software Foundation, 2012), cMeCab-Java version 2.0.1 (Taketa, 2012), and IK Analyzer 2012 upgrade 5 (Lin, 2012).

6.2 Results

Our multilingual sentiment classification system achieved very high accuracy (Table 6 and Table 7), without resorting to ad hoc NLP techniques, like parts-of-speech tagging and regular expression matching. It was also very fast (Table 5), because it did not rely on these techniques, which tend to be slow. The no answer rate for the naive Bayes classifier boosted with the logistic regression classifier is the rate at which documents were left unclassified because the two classifiers did not agree. Despite some documents being left unclassified, the two classifiers boosted together achieved a significantly higher accuracy than the naive Bayes classifier alone.

Recall from 5.2 that, in our implementation, the probability ratio in the decision rule of the naive Bayes classifier is only a rough approximation of the true value for the generalized 2-gram model and the hitting 2-gram model. The consequence of this is that we do not see a substantial improvement in classification accuracy for these models (Figure 3).

The less impressive performance overall for the Portuguese language is due to the quality of the data. For Portuguese, we had fewer documents to train on (Table 8), and the testing documents were, on average, quite short in length (Table 10). Notice that while we also had fewer training documents for the Dutch language, the average testing document length

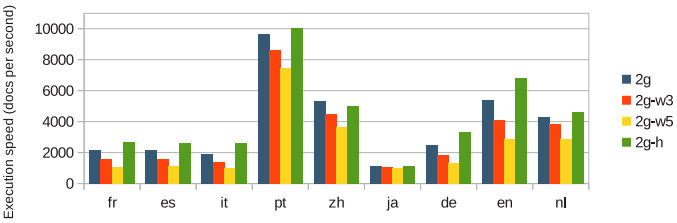


Figure 4: Mean classification speed (in documents per second per CPU core, over ten runs) of naive Bayes classifier boosted with logistic regression classifier for each model and each language.

	Classification speed (documents/second)			
	2g	2g-w3	2g-w5	2g-h
fr	2140±39	1589±41	1072±14	2662±63
es	2142±16	1589±63	1116±56	2642±26
it	1877±73	1388±73	985±32	2593±87
pt	9656±354	8653±272	7480±305	10073±408
zh	5355±64	4483±109	3663±90	4983±657
ja	1136±12	1067±32	1025±13	1134±14
de	2516±110	1859±107	1306±69	3328±151
en	5367±53	4086±20	2840±25	6795±101
nl	4277±703	3874±73	2857±82	4634±382

Table 5: Classification speed (mean and standard deviation, in documents per second, over ten runs) of naive Bayes classifier boosted with logistic regression classifier for each model and each language.

	Accuracy			
	2g	2g-w3	2g-w5	2g-h
fr	83.6±0.1	83.5±0.1	82.6±0.1	81.6±0.1
es	83.3±0.1	83.2±0.1	82.6±0.1	82.5±0.1
it	84.0±0.1	84.0±0.1	83.3±0.1	82.5±0.2
pt	74.7±0.5	72.4±0.7	70.4±0.7	77.3±0.3
zh	85.3±0.1	85.3±0.1	84.5±0.1	84.4±0.1
ja	91.6±0.1	92.3±0.1	91.6±0.1	91.2±0.1
de	89.1±0.1	88.9±0.1	87.8±0.1	87.1±0.0
en	85.5±0.0	85.2±0.0	84.2±0.0	84.3±0.0
nl	86.2±0.4	87.0±0.3	86.7±0.3	87.3±0.4

Table 6: Accuracy (mean and standard deviation, in percent, over ten runs) of naive Bayes classifier for each model and each language.

for Dutch is substantially greater than that for Portuguese. This is why the classification accuracy for Dutch did not suffer as much as it did for Portuguese. On the other hand, while the average testing document length for Chinese and Japanese is very short, we trained the algorithms with far more documents for these languages, and so the classification accuracies did not suffer. Thus, we can see a tradeoff between the amount of training data and the average length of the documents being classified. In our experiments with data from Twitter[13] (which we do not report in this paper), we found the same tradeoff: more training data is needed to achieve higher classification accuracies with documents that are so short in length.

As expected, more unique tokens need to be processed during training as the window size for the generalized 2-gram model is increased (Table 9 and Figure 5). These are the unique tokens that are used to compute the probabilities, and construct the data structure discussed in 5.2. When the number of unique tokens encountered during training is greater, the amount of memory that is consumed during classification is also greater. The classification speed also decreases as the number of unique tokens increases. The hitting 2-gram model drastically reduces the number of unique tokens, and, unsurprisingly, has a faster classification speed than the other models. The hitting 2-gram model also achieves

[13]http://www.twitter.com

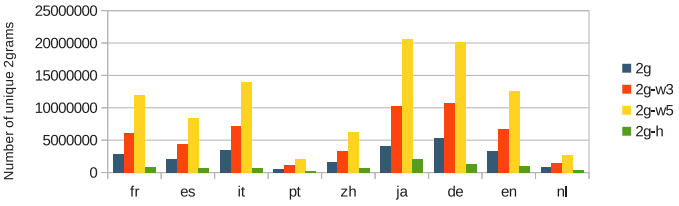


Figure 5: Mean number of unique tokens after training (over ten runs) for each model and each language.

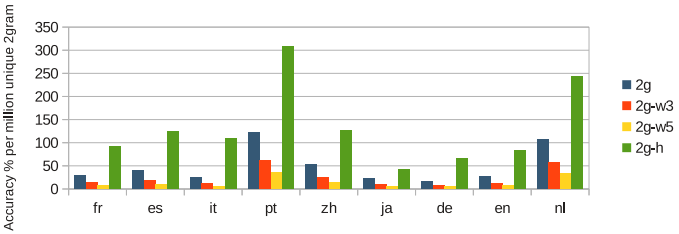


Figure 6: Mean accuracy per million unique tokens after training (in percent, over ten runs) of naive Bayes classifier for each model and each language.

greater accuracy than the other models when the amount of training data is less (i.e. for Portuguese and Dutch). In Figure 6, we see that when we normalize for the number of unique tokens, the hitting 2-gram model achieves far greater accuracy than the other models. Thus, for faster classification speed, reduced memory consumption, and lower quality training data, the hitting 2-gram is the way to go.

Our sentiment classification system was aggressively optimized for high speed and reduced memory consumption. The data for each language was aggregated in one flat file for ease of processing. Running the full set of ten runs of all experiments took less than an hour. Loading everything into memory consumed less than 3.5GB of the heap, which is unprecedented. When we ran the same set of experiments using LingPipe (Alias-i, 2012) for only the Spanish language and using only the 2-gram model, we found that more than 12GB of heap memory were required to even finish training.

Our results show that a simple and straightforward statistical approach with a large amount of training data rivals the many complex, ad hoc NLP approaches that are optimized for small amounts of training data. Important advantages of our approach are increased training and classification speeds, and reduced memory consumption. These are practical concerns that are not generally adequately addressed in the literature, particularly for the NLP approaches, which place a great emphasis on classification accuracy at the cost of speed and memory consumption. Our sentiment classification system achieves a good balance between these concerns.

	Accuracy				No answer rate			
	2g	2g-w3	2g-w5	2g-h	2g	2g-w3	2g-w5	2g-h
fr	91.3±0.1	91.0±0.1	90.6±0.1	90.7±0.1	14.2±0.1	13.9±0.1	14.1±0.0	15.6±0.1
es	90.4±0.1	90.3±0.1	90.1±0.1	89.8±0.1	14.5±0.1	14.5±0.2	14.7±0.2	14.2±0.2
it	91.7±0.1	91.6±0.1	91.3±0.1	91.1±0.1	14.9±0.1	14.8±0.2	15.1±0.2	15.5±0.1
pt	84.7±0.2	84.1±0.2	83.6±0.2	85.2±0.2	18.7±0.8	20.5±0.9	22.4±1.1	16.9±0.6
zh	91.0±0.0	90.8±0.1	90.3±0.0	90.6±0.1	11.7±0.1	11.2±0.1	11.0±0.1	12.3±0.1
ja	95.4±0.0	95.5±0.0	95.2±0.0	95.1±0.0	7.8±0.1	7.2±0.1	7.3±0.1	7.7±0.1
de	94.1±0.0	93.8±0.0	93.3±0.0	93.5±0.0	10.9±0.1	10.6±0.0	10.8±0.1	12.1±0.0
en	90.6±0.0	90.2±0.0	89.5±0.0	90.7±0.0	13.9±0.0	13.4±0.0	13.2±0.0	18.9±0.0
nl	92.0±0.1	91.7±0.1	91.0±0.1	91.7±0.2	17.1±0.2	15.3±0.1	14.3±0.1	16.2±0.2

Table 7: Accuracy and no answer rate (mean and standard deviation, in percent, over ten runs) of naive Bayes classifier boosted with logistic regression classifier for each model and each language.

	Positive documents		Negative documents	
	# trained	# tested	# trained	# tested
fr	26455±79	116704±93	26556±66	116704±93
es	12234±158	55267±92	12061±102	55267±92
it	21175±140	92502±70	20272±89	92502±70
pt	3593±78	16349±43	2931±41	16349±43
zh	30914±194	124232±48	30989±48	124232±48
ja	218278±526	889453±391	219019±396	889453±391
de	54351±255	237839±206	54578±142	237839±206
en	87833±400	367812±224	85626±207	367812±224
nl	6907±83	27691±65	6765±67	27691±65

Table 8: Number of positive and negative documents in the training and testing sets (mean and standard deviation, over ten runs) for each language.

	Number of unique tokens			
	2g	2g-w3	2g-w5	2g-h
fr	2914884±9360	6172487±20129	11914397±39474	882036±2352
es	2140713±12048	4406273±26340	8373245±51332	664764±3752
it	3425993±7458	7223768±16530	13926488±33039	749045±1400
pt	608836±1041	1165038±1953	2033085±3804	251139±499
zh	1640449±5753	3321802±11719	6219173±23299	667135±1756
ja	4142506±6055	10307368±14345	20572997±26398	2140987±2610
de	5372769±14153	10754868±29605	20217297±56647	1306724±2862
en	3290897±3804	6713768±7239	12604478±13434	1021480±1497
nl	809873±2663	1508605±5762	2634343±11110	359049±1642

Table 9: Number of unique tokens after training (mean and standard deviation, over ten runs) for each model and each language.

Conclusion and Future Work

In this paper, we presented an empirical study of two sentiment classification algorithms applied to nine languages (including Germanic, Romance, and East Asian languages). One of these algorithms is a naive Bayes classifier, and the other is an algorithm that boosts a naive Bayes classifier with a logistic regression classifier, using majority vote. We implemented these algorithms as part of a system for classifying the sentiment of multilingual text data. Our implementation is fast, and has high classification accuracy.

We also considered a generalization of the n-gram model for representing text data, and

	Document length in test data
fr	1941±2
es	2116±2
it	2373±2
pt	210±1
zh	112±0
ja	132±0
de	1769±2
en	746±1
nl	859±1

Table 10: Mean document length, with standard deviation, over ten runs, in test data.

a variant of this generalization that helps reduce memory consumption. Along with the standard n-gram model, these two models are built into our system. We evaluated all of these models in the empirical study that we presented in this paper.

For the empirical study, we trained and tested our system on a data set that is substantially larger than that typically encountered in the literature. We generated this data set by crawling and mining various web sites for reviews of products and services. For each experiment in the study, we sampled disjoint training and testing sets uniformly at random from this large data set. Unlike the usual approach in the literature, the testing sets were much larger than the training sets (at least four times larger), and the experiments were repeated many times. We did this to ensure that our results were statistically significant.

As we have shown in this paper, statistical methods applied to large amounts of data are effective for the sentiment classification problem. It would be interesting to investigate the application of this approach to the problem of relevance (i.e. determining whether a document conveys any sentiment at all). Previous efforts have been overly complicated (Pang and Lee, 2004). One approach that we are considering is to take a list of n-grams that are most indicative of sentiment (determined using Pearson's chi-squared test, as discussed in 5.1), and computing the mean and standard deviation for the frequency of occurrence of these words in the training documents. During testing, the frequency of occurrence for these words in the test documents can be compared to the mean we computed. If the frequency of occurrence is not less than one standard deviation below the mean, then a document can be deemed relevant.

We are also interested in commercializing our sentiment classification system by selling it to social media analytics firms, such as Sysomos[14] and BrandWatch[15]. The existing players in the sentiment classification field (e.g. Saplo[16], Lexalytics[17], OpenAmplify[18], and SNTMNT[19]) are not transparent about what they are doing, and it is not clear how robust their offerings are. If commercialization fails, then we intend to make our sentiment classification system freely available under the GPL[20], since one of our great passions is educating the public on the power of machine learning methods.

[14] http://www.sysomos.com
[15] http://www.brandwatch.com
[16] http://saplo.com
[17] http://www.lexalytics.com
[18] http://www.openamplify.com
[19] http://www.sntmnt.com
[20] http://www.gnu.org/copyleft/gpl.html

References

Alias-i (2012). Lingpipe version 4.1.0. http://alias-i.com/lingpipe/index.html.

The Apache Software Foundation (2012). Apache Lucene Core version 3.6.0. http://lucene.apache.org/core.

Bengio, Y., Ducharme, R., Vincent, P., and Janvin, C. (2003). A neural probabilistic language model. *The Journal of Machine Learning Research*, 3(Feb):1137–1155.

Bespalov, D., Bai, B., Qi, Y., and Shokoufandeh, A. (2011). Sentiment classification based on supervised latent n-gram analysis. In *Proceedings of the 20th ACM International Conference on Information and Knowledge Management (CIKM '11)*, pages 375–382.

Chang, C.-C. and Lin, C.-J. (2011). LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2(3):27:1–27:27. Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm.

Collobert, R. and Weston, J. (2008). A unified architecture for natural language processing: deep neural networks with multitask learning. In *Proceedings of the 25th International Conference on Machine Learning (ICML '08)*, pages 160–167.

Cormen, T. H., Leiserson, C. E., Rivest, R. L., and Stein, C. (2001). *Introduction to Algorithms*. MIT Press and McGraw-Hill, 2nd edition.

Cortes, C. and Vapnik, V. (1995). Support-vector networks. *Machine Learning*, 20(3): 273–297.

Csiszár, I. (1996). Maxent, mathematics, and information theory. In Hanson, K. M. and Silver, R. N., editors, *Maximum Entropy and Bayesian Methods: Proceedings of the 15th International Workshop on Maximum Entropy and Bayesian Methods*, pages 35–50.

Domingos, P. and Pazzani, M. (1997). On the optimality of the simple bayesian classifier under zero-one loss. *Machine Learning - Special issue on learning with probabilistic representations*, 29(2-3):103–130.

Fan, R.-E., Chang, K.-W., Hsieh, C.-J., Wang, X.-R., and Lin, C.-J. (2008). LIBLINEAR: A library for large linear classification. *The Journal of Machine Learning Research*, 9(Aug):1871–1874. Software available at http://www.csie.ntu.edu.tw/~cjlin/liblinear.

Feinberg, J. (2012). Wordle. http://www.wordle.net/.

Guthrie, D., Allison, B., Liu, W., Guthrie, L., and Wilks, Y. (2006). A closer look at skip-gram modelling. In *Proceedings of the Fifth International Conference on Language Resources and Evaluation (LREC - 2006)*, pages 1222–1225.

Hsu, C.-W., Chang, C.-C., and Lin, C.-J. (2010). A practical guide to support vector classication. http://www.csie.ntu.edu.tw/~cjlin/papers/guide/guide.pdf.

Joachims, T. (1998). Text categorization with suport vector machines: Learning with many relevant features. In *Proceedings of the 10th European Conference on Machine Learning (ECML '98)*, pages 137–142.

Joachims, T. (1999). Making large-scale support vector machine learning practical. In Schölkopf, B. and Smola, A., editors, *Advances in kernel methods*, pages 169–184. MIT Press Cambridge, MA, USA.

Lewis, D. D. (1998). Naive (bayes) at forty: The independence assumption in information retrieval. In *Proceedings of the 10th European Conference on Machine Learning (ECML '98)*, pages 4–15.

Lin, L. Y. (2012). IK Analyzer 2012 upgrade 5. `http://code.google.com/p/ik-analyzer`.

Nie, J.-Y., Gao, J., Zhang, J., and Zhou, M. (2000). On the use of words and n-grams for chinese information retrieval. In *Proceedings of the Fifth International Workshop on Information Retrieval with Asian Languages (IRAL '00)*, pages 141–148.

Nigam, K., Lafferty, J., and McCallum, A. (1999). Using maximum entropy for text classification. In *Workshop on Machine Learning for Information Filtering (IJCAI '99)*, pages 61–67.

Pang, B. and Lee, L. (2004). A sentimental education: sentiment analysis using subjectivity summarization based on minimum cuts. In *Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics (ACL '04)*.

Pang, B. and Lee, L. (2008). Opinion mining and sentiment analysis. *Foundations and Trends in Information Retrieval*, 2(1-2):1–135.

Pang, B., Lee, L., and Vaithyanathan, S. (2002). Thumbs up?: sentiment classification using machine learning techniques. In *Proceedings of the ACL-02 conference on Empirical methods in natural language processing (EMNLP '02)*, pages 79–86.

Rosenblatt, F. (1957). The perceptron–a perceiving and recognizing automaton. Technical Report 85-460-1, Cornell Aeronautical Laboratory.

Taketa, K. (2012). cMeCab-Java version 2.0.1. `http://code.google.com/p/cmecab-java`.

Waldvogel, B. (2012). Java LIBLINEAR version 1.8. `http://www.bwaldvogel.de/liblinear-java/`.

Yang, Y. and Pedersen, J. O. (1997). A comparative study on feature selection in text categorization. In *Proceedings of the Fourteenth International Conference on Machine Learning (ICML '97)*, pages 412–420.

Extraction of Russian Sentiment Lexicon for Product Meta-Domain

Ilia Chetviorkin[1] *Natalia Loukachevitch*[2]
(1) Faculty of Computational Mathematics and Cybernetics,
Lomonosov Moscow State University,
Moscow, Leninskiye Gory 1, Building 52
(2) Research Computing Center,
Lomonosov Moscow State University,
Moscow, Leninskiye Gory 1, Building 4
ilia.chetviorkin@gmail.com, louk_nat@mail.ru

ABSTRACT

In this paper we consider a new approach for domain-specific sentiment lexicon extraction in Russian. We propose a set of statistical features and algorithm combination that can discriminate sentiment words in a specific domain. The extraction model is trained in the movie domain and then utilized to other domains. We evaluate the quality of obtained sentiment vocabularies intrinsically. Finally we combine the sentiment lexicons from five domains to obtain one general lexicon for the product meta-domain. We demonstrate the robustness of the extracted lexicon in the cross-domain sentiment classification in Russian.

TITLE AND ABSTRACT IN RUSSIAN

Извлечение Словаря Оценочной Лексики на Русском Языке для Мета-Области Товаров

В данной работе рассматривается новый подход к извлечению предметно-ориентированного словаря оценочной лексики на русском языке. Мы предлагаем использовать совокупность статистических и лингвистических признаков, позволяющих выявлять оценочные слова, и комбинировать эти признаки с помощью алгоритмов машинного обучения. Модель извлечения создается для предметной области фильмов, а затем применяется в других предметных областях. Мы оцениваем качество полученных словарей оценочных слов посредством ручной разметки. Наконец, мы собираем из отдельных словарей общий словарь оценочных слов, рассматривая его как оценочный словарь в широкой области товаров. Мы демонстрируем полезность полученного общего лексикона в задаче переноса модели анализа тональности с одной области на другую для отзывов пользователей на русском языке.

KEYWORDS : Sentiment Analysis, Sentiment Lexicon, Domain Adaptation.
KEYWORDS IN RUSSIAN: Анализ Тональности, Оценочные слова, Настройка на Предметную Область

Proceedings of COLING 2012: Technical Papers, pages 593–610,
COLING 2012, Mumbai, December 2012.

В последнее время большие усилия были направлены на решение задачи анализа мнений в различных предметных областях. Автоматизированные подходы к анализу тональности могут быть полезны для государственных органов и политиков, компаний и простых пользователей. Одной из важнейших задач, являющейся основой для анализа мнений в текстах, написанных на различных языках, является создание словарей оценочных слов.

Многие исследователи создают словари общеупотребительных оценочных слов для своих языков. Вместе с тем известно, что в разных предметных областях могут применяться достаточно разные наборы оценочных выражений. Наконец, предметные области могут иметь сходство между собой в используемой оценочной лексике. Так, такие оценочные слова как *негодяй* или *зло* одинаково неприменимы ко всем областям оценки качества товаров.

В данной работе мы исследуем новую идею разработки русского словаря оценочной лексики для широкой области товаров. При этом важно подчеркнуть, что в настоящее время нет общественно доступного русскоязычного словаря оценочной лексики. Наш метод базируется на обучении алгоритма извлечения русской оценочной лексики в одной предметной области, и затем переносе обученной модели на другие предметные области. Мы показываем, что модель извлечения оценочной лексики может быть перенесена на другие предметные области, если имеются все необходимые для работы системы данные. Мы применяем нашу модель к нескольким предметным областям и затем из оценочных словарей отдельных предметных областей собираем единый словарь оценочной лексики, рассматривая его как словарь оценочной лексики в широкой области товаров.

Извлечение оценочных слов в заданной предметной области основано на нескольких текстовых коллекциях: коллекции отзывов о продуктах с оценками пользователей, коллекции описаний продуктов и контрастной коллекции (например, новостная коллекция). Такие коллекции могут быть автоматически сформированы для разных предметных областей. Кроме того, мы предположили, что можно выделить некоторые части корпуса мнений (например, о фильмах), в которых концентрация оценочных слов выше: предложения, заканчивающиеся на «!» или «…»; короткие предложения не более чем из 7 слов; предложения, содержащие слово «фильм» без других существительных. Условно назовем этот корпус – малый корпус.

Для каждого слова в коллекции отзывов мы вычисляем набор статистических и лингвистических признаков.

Для обучения алгоритмов нам необходимо размеченное множество слов. Для этого мы вручную разметили множество всех слов с частотой выше трех из предметной области о фильмах (18362 слова). Мы относили слово к категории оценочных в случае если могли представить его в каком-либо оценочном контексте.

Мы решали задачу классификации на два класса: разделение всех слов на оценочные и неоценочные. Для этих целей использовались следующие алгоритмы: *Logistic Regression*, *LogitBoost* и *Random Forest*. Все параметры алгоритмов были выставлены в соответствии с их значениями по умолчанию.

Используя данные алгоритмы, мы получили списки слов, упорядоченные по вероятности оценочности слов. Для оценки качества этих списков использовалась мера *Precision@n*. Для сравнения качества работы системы в разных предметных областях мы использовали значение n = 1000.

Мы заметили, что извлеченные списки оценочных слов существенно различаются в зависимости от алгоритма. Поэтому мы решили вычислить среднее от значений вероятностей в каждом из списков. В результате качество автоматического извлечения оценочных слов в области фильмов *Precision@1000* составило 81.5%.

Для использования системы в новой предметной области необходимо собрать аналогичный набор коллекций, как и предметной области о фильмах. Мы применили модель извлечения оценочных слов в таких областях, как книги, игры, цифровые камеры, мобильные телефоны.

Для того чтобы собрать обобщенный список оценочной лексики в области товаров, мы применили формулу, поощряющую нахождение оценочного слова в начале наибольшего количества полученных списков оценочной лексики в разных предметных областях. Качество полученного списка составило P@1000 = 91.4%.

Для проверки полезности полученного обобщенного списка оценочных слов в мета-области товаров мы протестировали его в задаче переноса системы анализа тональности с одной области на другую.

Для тестирования мы взяли по 1000 положительных и 1000 отрицательных отзывов в четырех предметных областях. Мы обучали классификатор тональности в одной области на трех разных наборах признаков: всех словах, извлеченному списку оценочных слов этой предметной области и обобщенному списку оценочных слов. Далее мы применяли обученный классификатор на другой предметной области. Всего было рассмотрено 9 пар предметных областей. Было показано, что в среднем классификатор, обученный на обобщенном списке предметных областей, лучше переносится на новую предметную область.

Таким образом, в нашей работе мы создали русскоязычный список оценочных слов для широкой области товаров и показали его полезность в задачах, связанных с настройкой систем анализа тональности на новую предметную область. Мы планируем опубликовать полученный список оценочных слов, и это будет первый общественно доступный список оценочной лексики для русского языка.

1 Introduction

Over the last few years a lot of efforts were made to solve sentiment analysis tasks in different domains. Automated approaches to sentiment analysis can be useful for state bodies and politicians, companies, and ordinary users. Most of these efforts concern English, where a lot of resources and tools for natural language processing and especially for sentiment analysis exist.

One of the important tasks, considered as a basis for sentiment analysis of documents written in a specific language, is a creation of its sentiment lexicon (Abdul-Mageed et al., 2011; Peres-Rosas et al., 2012).

Usually authors try to gather general sentiment lexicons for their languages. However a lot of researchers stress the differences between sentiment lexicons in specific domains. For example, "must-see" is a strongly opinionated word in the movie domain, but neutral in the digital camera domain (Blitzer et al., 2007). For these reasons, supervised learning algorithms trained in one domain and applied to other domains demonstrate considerable decrease in the performance (Ponomareva & Thelwall, 2012; Read & Carroll, 2009; Taboada et al., 2011).

To overcome this issue various adaptation methods are proposed, like ensembles of classifiers (Aue & Gamon, 2005) or graph-based approaches (Wu et al., 2009). Nevertheless such approaches usually do not work well for domains whose lexicons differ significantly and recent studies are focused on bridging the gap between domain-specific words (Pan et al, 2010). Indeed, sentiment lexicons adapted to a particular domain or topic have been shown to improve task performance in a number of applications, including opinion retrieval (Jijkoun et al., 2010), and expression-level sentiment classification (Choi & Cardie, 2009). In addition sentiment word extraction from a text collection enables to find slang and non-vocabulary words, which can be strong sentiment predictors.

Stressing the differences in sentiment lexicons between domains, one should understand that domains can form clusters of similar domains. So a lot of sentiment words relevant to various product domains are not relevant to the political domain or the general news domain and vice versa. For example, such words as *evil* or *villain* are not applicable to all product domains. Therefore we suppose that gathering a specialized sentiment lexicon for the product meta-domain can be useful for researchers and practitioners.

In the current study we focus on the novel idea of construction of Russian sentiment lexicon for the product meta-domain. At this moment we should also emphasize that no publicly available Russian sentiment lexicon exists. Our method is based on training of the supervised algorithm for sentiment lexicon extraction in one domain and further transfer of the model to other domains. We show that in comparison with supervised sentiment classifiers, our sentiment lexicon extractor can be transferred to other domains if all necessary data are available. The trained sentiment lexicon extraction model is applied to an extensive number of domains and then extracted lexicons are summed up to the single list of sentiment words. So we obtain the generalized sentiment lexicon for the group of domains.

We opt to focus on recognizing sentiment words without any polarity scores. It is pointed in the research papers that the two-stage approach is often beneficial, in which on the first stage we determine main sentiment bearers in a text and on the second stage classify them according to the polarity (Pang and Lee, 2008). Thus such sentiment lexicons can be very useful for more accurate processing of user opinions.

We evaluate the extracted general lexicon intrinsically, by manually labelling of word lists, and extrinsically, by transferring of sentiment classifiers based on our general lexicon to domains without any labelled data. The results demonstrate the effectiveness of our constructed general sentiment lexicon.

The reminder of this article is organized as follows. In Section 2 we observe state-of-the-art methods for the sentiment lexicon generation, Section 3 describes the data collections and features involved in the model, in Section 4 we utilize our approach for four other domains and combine sentiment word vocabularies from all of them in Section 5. Finally, in Section 6 we conduct the experiments on the cross-domain sentiment classification involving extracted sentiment words.

2 Related work

The related works can be divided into two categories: the creation of a sentiment lexicon for a specific language, and the creation of a sentiment lexicon for a specific domain.

2.1 Creation of sentiment lexicons for specific languages

There are four main methods that are exploited by researchers to develop the sentiment lexicons for their languages: use of translated English sentiment resources, use of language-specific wordnets aligned to Princeton WordNet, use of corpora-based techniques similar to the techniques proposed for English sentiment lexicon extraction, use of electronic dictionaries of specific languages.

In (Mihalcea et al., 2007) two methods for translating sentiment lexicons to Romanian are proposed. The first method uses bilingual dictionaries to translate an English sentiment lexicon gathered using OpinionFinder (Wiebe & Riloff, 2005) and obtain 4,983 Romanian sentiment words. The evaluation of randomly chosen units shows the percentage of the sentiment words in the list is around 50%; besides, the low coverage of existing Romanian sentiment expressions is revealed. The second method is based on parallel corpora. The corpus on the source language is annotated with sentiment information, and the information is then projected to the target language. The problems arise due to mistranslations, e.g. because irony is not recognized.

Researchers in (Banea et al., 2008) propose to use a monolingual dictionary to acquire a sentiment lexicon from 60 manually selected seeds, equally sampled from verbs, nouns, adjectives and adverbs. To filter erroneous entries the LSA similarity measure is used.

In (Perez-Rosas et al., 2012) a method to derive Spanish lexicons by using manually or automatically annotated data available in English is presented. The multilingual sense-level aligned WordNet structure is used to generate a highly accurate (90%) polarity lexicon comprising 1,347 entries, and one with accuracy (74%) encompassing 2,496 words.

(Clematide & Klenner, 2010) begin their work with German polarity lexicon from 8000 polarity words obtained from GermaNet, a WordNet-like lexical database. Revealing rather low coverage of German novels by polarity-bearing adjectives from this list, they expand the set of 2899 German sentiment adjectives extracting coordinated adjectives pairs similar to (Hatzivassiloglou & McKeown, 1997).

To enhance the quality of dictionary-based methods for the general sentiment vocabulary generation in other languages, (Steinberger et al., 2011) create two source sentiment vocabularies: English (2400 entries) and Spanish (1737 entries). Both lists are translated by Google translator to the target language. Only overlapping entries from each translation are taken into further consideration. The set of target languages comprises six languages including Russian. The extracted Russian list of sentiment words contained 966 entries with accuracy of 94.9%.

In comparison with these approaches we create a Russian lexicon for a very broad domain - meta-domain of products and services, for which we do not use any dictionaries - only users' reviews, and in this paper we show usefulness of this general lexicon.

2.2 Development of sentiment lexicons for specific domains

In many studies domain-specific sentiment lexicons are created using various types of propagation from a seed set of words, usually a general sentiment lexicon (Kanayama & Nasukawa, 2007; Lau et al., 2011; Qiu et al., 2011). In such approaches an important problem is to determine an appropriate seed lexicon for propagation, which can heavily influence the quality of the results. Besides, the propagation often lead to unclear for a human sentiment lists. So, for example, in (Lau et al., 2011) only 100 first obtained sentiment words were evaluated by experts, *precision@100* was around 80%, what means that the intrinsic quality of the extracted 4000 lexicon (as announced in the paper) can be quite low.

Another approaches apply statistical measures based on domain-specific corpora to extract domain-specific sentiment words: $\chi 2$ (Jijkoun et el., 2010), divergence from randomness (DFR), which measures the divergence between a term's probability distribution in a set of relevant and opinionated documents and its probability distribution in a set of relevant documents (He et al., 2009) etc.

The sentiment lexicon extraction method proposed in this paper exploits a set of statistical and linguistic measures, which can characterize domain-specific sentiment words from different sides. We combine these features into a single model using machine learning methods. Then we train it on one domain and show that such a model can be effectively transferred to other domains for extraction of their sentiment lexicons.

3 Extraction of sentiment lexicon in a specific domain

In the current study a new supervised method for domain-specific sentiment lexicon extraction is presented. We train our model in one domain and then apply it to several others. Finally, we combine the extracted word lists to construct a general lexicon of sentiment words typical for products and services.

Our approach is based on several text collections, which can be automatically formed for many domains, such as: a collection of product reviews with authors' evaluation scores, a text collection of product descriptions and a contrast corpus (for example, a general news collection). For each word in the review collection we calculate a set of linguistic and statistical features using the aforementioned collections and then apply machine learning algorithms for term classification.

Our method does not require any seed words, and is rather language-independent, however, lemmatization (or stemming) and part-of speech tagging are desirable. Working with Russian language, we use a dictionary-based morphological processor, including unknown word processing. Below in the text we will speak only about lemmatized words.

3.1 Data preparation

We collected 28, 773 movie reviews of various genres from the online recommendation service *www.imhonet.ru*. For each review, user's score on a ten-point scale was extracted. We called this collection the **review collection**.

Example of the movie review:

Nice and light comedy. There is something to laugh - exactly over the humour, rather than over the stupidity… Allows you to relax and gives rest to your head.

We also required a contrast collection of texts for our experiments. In this collection the concentration of opinions should be as little as possible. For this purpose, we collected 17, 680 movie descriptions. This collection was named the **description collection**.

One more contrast corpus was a collection of two million news documents. We had calculated a document frequency of each word in this collection and used only this frequency list further. This list was named the **news corpus**.

3.2 Collections with higher concentration of opinions

We suggested that it was possible to extract some fragments of reviews from the review collection that had higher concentration of sentiment words. These fragments may include:

- Sentences ending with a "!";
- Sentences ending with a "…";
- Short sentences, no more than seven word length;
- Sentences containing the word «movie» without any other nouns.

We called this collection the **small collection**.

3.3 Statistical features

Our aim is to create a high quality list of sentiment words based on the combination of various discriminative features. We propose the following set of features for each word:

- Frequency-based

- Collection frequency $f(w)$ (i.e. number of occurrences in all documents in the collection)
- Document frequency
- Frequency of capitalized words
- Weirdness
- TFIDF

- Rating-based
  - Deviation from the average score
  - Word score variance
  - Sentiment category likelihood for each *(word, category)* pair

We will consider some of them in more detail.

Frequency of capitalized words. The meaning of this feature is the frequency (in the review corpus) of each word starting with the capital letter and not located at the beginning of the sentence. With this feature we are trying to identify potential proper names, which are always neutral.

Weirdness. To calculate this feature two collections are required: one with high concentration of sentiment words and the other – contrast one. The main idea of this feature is that sentiment words will be «strange» in the contexts of the contrast collection. This feature is calculated as follows (Ahmad et al., 1999):

$$Weirdness = \frac{P_s(w)}{P_g(w)}$$

where $P_s(w)$ – probability of the word in a special corpus, $P_g(w)$ – probability of the word in a general corpus. Here and further we consider maximum likelihood estimation of the probabilities. Instead of the collection frequency one can use the document frequency for the probability calculation.

Weirdness was calculated using the following collection pairs: *opinion-news*, *opinion-description, description-news* with document frequency and *small-description*, *opinion-description* with collection frequency.

TFIDF. We use TFIDF variant described in (Callan et al., 1992), based on BM25 function. We calculate TFIDF using the collection pairs: *small-news*, *small-description*, *opinion-news*, *opinion-description, description-news.*

3.4 Rating-based features

As we mentioned above we had collected user's numerical score (on a ten point scale) for each review. Let $C = \{1...10\}$ to be the set of rating categories in the review collection. First, we want to give some definitions, which we will use further.

Definition 1.

i. The probability of a rating category **c** given a word ***w***:

$$P(c \mid w) = \frac{f(w,c)}{\sum_{c_i \in C} f(w,c_i)}$$

ii. The probability of a word ***w*** given a rating category **c**:

$$P(w \mid c) = \frac{f(w,c)}{\sum_{w_i \in C} f(w_i,c)}$$

Definition 2.

i. An expected category for a given word:

$$E(c \mid w) = \sum_{c_i \in C} c_i \cdot P(c_i \mid w)$$

ii. An expected category in the review collection:

$$E(c) = \sum_{c_i \in C} c_i \cdot P(c_i)$$

Using our definitions we suggest the following features:

Deviation from the average score.

$$Dev(w) = \left| E(c \mid w) - E(c) \right|$$

This feature can discriminate words appearing in a wide range of rating categories.

Word score variance. One more useful predictor is word score variance. If a word has small variance then it might be used in reviews with similar scores and has high probability to be a sentiment word.

$$Var(w) = E(c^2 \mid w) - E(c \mid w)^2$$

Scaled likelihood. To get some intuition about how likely a word is to appear in each sentiment class we define a scaled log-likelihood:

$$Lhc(w) = \log \frac{P(w \mid c)}{P(w)}$$

Scalability is required to be comparable between words. We have also added some features aggregating *Lhc values* like maximum and average.

3.5 Morphological Features

Some linguistic features were also added to our system because they can play crucial role in improving the sentiment lexicon extraction.

- Four binary features indicating the word part of speech (noun, verb, adjective and adverb)

- Two binary features reflecting POS ambiguity (i.e. word can have various parts of speech depending on a context) and the feature indicating if this word is recognized by the POS tagger.
- Predefined list of prefixes of a word (for example, Russian prefixes "*ne*", "*bes*", "*bez*" etc. similar to English "*un*", "*in*", "*im*" etc.)

The last feature is a strong predictor for words starting with negation.

3.6 Algorithms and evaluation

To train supervised machine learning algorithms we needed a set of labeled sentiment words. For our experiments we manually labeled words with the frequency greater than three in the movie review collection (18362 words). We marked up a word as a sentiment one in case we could imagine it in any opinion context in the movie domain. All words were tagged by two assessors. If there was a disagreement about the sentiment of a specific word, the collective judgment after discussion was used as a final ground truth. As a result of our assessment procedure we had obtained the list of 4079 sentiment words in the movie domain.

We solved the two class classification problem: to separate all words into sentiment and neutral categories. For this purpose Weka[1] data mining tool was used. We considered the following algorithms: *Logistic Regression*, *LogitBoost* and *Random Forest*. All parameters in the algorithms were set to their default values. For each experiment 10 fold cross-validation was used.

Using this algorithms we obtained word lists, ordered by the predicted probability of their opinion orientation. To measure the quality of these lists the *Precision@n* metric was used. This metric was very convenient for measuring the quality of list combinations and it could be used with different thresholds. To compare quality of the algorithms in different domains we chose n = 1000. This level was not too large for the manual labeling and demonstrated the quality in an appropriate way.

The results of classification are in Table 1.

Logistic Regression	**LogitBoost**	**Random Forest**	**Average**
75.7%	75.3%	72.4%	81.5%

TABLE 1 – Precision@1000 of word classification

We noticed that the lists of sentiment words extracted by the algorithms differ significantly. So we decided to average word probability values in these three lists. The result of this summation can be found in the last column of the Table 1.

As the baseline for our experiments we used the lists ordered by frequency in the review collection and deviation from the average score. *Precision@1000* in these lists was 26.9% and 35.5% accordingly. Thus our algorithms gave significant improvements over the baselines. All the other features can be found in Table 2.

[1] http://www.cs.waikato.ac.nz/ml/weka/

Let us look at some examples of sentiment words with the high probability value in the sum list: *Trogatel'nyi (affective), otstoi (trash), fignia (crap), otvratitel'no (disgustingly), posredstvenniy (satisfactory), predskazuemyi (predictable), ljubimyj (love) etc.*

Feature	Collection	Precision @1000
TFIDF	small – news	38.5%
TFIDF	small – descr	36.4%
TFIDF	review – news	30.5%
TFIDF	review – descr	39.8%
Weirdness	review – news (doc. count)	31.7%
Weirdness	review – descr (doc. count)	48.1%
Weirdness	small – descr (frequency)	49.1%
Weirdness	review – descr (frequency)	46.6%
Dev	review	35.5%
Var	review	21.5%
Lhc	review	33.0%
Frequency	review	26.9%
Frequency	small	31.9%
Document Frequency	review	27.8%

TABLE 2 – Precision@1000 for different features

4 Model adaptation

In the previous section we described the construction of the sentiment lexicon extraction model for the movie domain. The next step of the current research is utilizing this model in four other domains and combining obtained results to form a general sentiment lexicon for the product meta-domain.

	Review Collection	Description Collection	Source
Books	23, 883	22, 321	Imhonet
Games	7, 928	1, 853	Imhonet
Digital Cameras	10, 208	920	Yandex Market
Mobile Phones	30, 620	890	Yandex Market

TABLE 3 – The characteristics of the data collections

4.1 Additional datasets

We collected[2] data in the four domains: books, computer games, mobile phones and digital cameras. The structure of the datasets is the same as for movie domain. Data collection characteristics for each domain can be found in Table 3.

In further experiments we use the same **news corpus** as for movie domain.

4.2 Model utilization and evaluation

For all words in a particular field (excluding low frequent ones) we computed feature vectors (see Sections 3.3-3.5) and constructed a domain word-feature matrix. We applied our classification model, which was trained in the movie domain, to these word-feature matrixes and manually evaluated the first thousand of the most probable sentiment words in each domain. The results of the evaluation are in Table 4.

	Average
Books	86.0%
Games	72.2%
Digital Cameras	62.0%
Mobile Phones	73.2%

TABLE 4 – The results of domain adaptation

Despite the drop in some other domains the quality of sentiment word extraction continues to be much higher than the quality level of single features (Table 2). So we can conclude that the sentiment lexicon extraction model is robust enough to be transferred to other domains.

5 Developing the Russian lexicon for product meta-domain

To construct the general sentiment lexicon for products and services we combine sentiment word lists from five domains. We want to boost words that occur in many different domains and have high weights in each of them. We propose the following function for the word weight in the resulting list:

$$R(w) = \max_{d \in D}(prob_d(w)) \cdot \sum_{d \in D} \frac{1}{|D|} \cdot \left(1 - \frac{pos_d(w)}{|d|}\right)$$

where D – is the domain set with five domains, d is the sentiment word list for a particular domain and $|d|$ is the total number of words in this list. Functions $prob_d(w)$ and $pos_d(w)$ are the sentiment probability and position of the word in the list d .

The *Precision@1000* of the obtained sentiment word list is **91.4%.** The inter-rater agreement between the two Russian annotators is measured at 0.84 (κ = 0.63).

[2] Review data collections in the book and digital camera domains are obtained from Russian Seminar of Information Retrieval Methods (www.romip.ru)

As a baseline for our method of construction of the general sentiment lexicon for product meta-domain, we take the combined *weirdness* list (review – descr) as rather simple, but high quality one. We construct it from weirdness lists in the same manner as described in the beginning of the section. The Precision@n plots of the extracted lexicon and weirdness list combination are depicted on Figure 1.

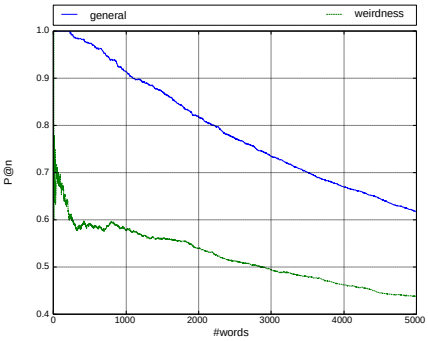


FIGURE 1 – Precision@n depending on #words

The first ten most probable sentiment words are: *bespodobniy (matchless), kleviy (cool), obaldenniy (astounding), neponiatniy (incomprehensible), neprivichniy (unusual), srednenkiy (mediocre), posredstvenniy (moderate), neploho (not bad), otlichneishiy (splendiferous), nenuzhniy (unnecessary).* This sentiment lexicon is clean enough to be used in various sentiment analysis tasks.

This meta-domain list of sentiment words consists of words really used in users' reviews and its creation does not require any dictionary resources. We plan to make it available for further research in sentiment analysis of Russian texts.

6 Lexicon evaluation on the cross-domain sentiment classification task

6.1 Experimental setup

To evaluate usefulness of our meta-domain sentiment list we test it in the cross-domain sentiment classification task as described for example in (Blitzer et al., 2007; Bollegala et al., 2011; Pan et al., 2010). In these studies the dataset consisting of Amazon product reviews for four different product types (books (B), DVDs (D), electronics (E) and kitchen appliances (K)) is used. There are 1000 positive and 1000 negative reviews selected randomly and labeled for each domain. Domain-adaptation algorithms are trained on the one domain (source domain) and tested on the other domain (target domain).

We do not compare our approach with these approaches because we do not make any efforts to adapt a classifier to a new domain. We use the similar setup to show the

generalization abilities of the sentiment word lists. In these experiments we try to demonstrate the influence of our meta-domain list on the sentiment classification quality in a new domain without any labeled data.

So we randomly take 1000 positive and 1000 negative labeled Russian reviews from four domains: movies (**M**), books (**B**), mobile phones (**P**) and digital cameras (**C**). The reviews with user's score 9-10 are considered as positive and reviews with authors' score 1-4 are considered as negative.

Taking pairs of the domains, we train a sentiment classifier in one domain (source domain) and then transfer the classifier to the other domain (target domain). We treat a review text as a bag-of-words and use the following features for classification:

- All frequent words of the source domain (**F**ull **L**ist),
- Sentiment words from the generated sentiment lexicon of the source domain (**S**ource **D**omain **L**exicon),
- Words from the meta-domain sentiment lexicon, excluding the sentiment vocabulary of the target domain during the extraction (**G**eneral **L**exicon).

In this task we utilize the LIBLINEAR realization of the support vector machine (SVM) classification algorithm with the default parameter values.

Additionally we include TFIDF weights for each feature, as it is pointed to give higher quality of the classification in comparison with the binary weights and we also take into account the polarity influencers, which can revert or magnify the polarity of the following words. The specific details can be found in (Chetviorkin & Loukachevitch, 2011).

We performed experiments with the proposed feature sets on the 9 domain pairs: ***B→C, M→C, P→C, B→P, M→P, C→P, M→B, P→B, C→B*** where the letter before an arrow corresponds with the source domain and the letter after an arrow corresponds with the target domain. We do not consider cross-domain sentiment classification with the movie domain as a target one, because we manually labeled and trained the sentiment word extraction model in it, and the results of the classification can be unclear.

For domain specific and general sentiment lexicons we explored different word quantity thresholds: {1000, 2000, 3000, 4000, 5000, 6000, 7000, 8000, 9000, 10000} and report the results with each of them (see Figure 2 and 3).

6.2 Metrics

We denote by $A(S,T,L)$ the accuracy obtained during the transfer from source domain ***S*** to target domain ***T*** of the sentiment classifier trained using the lexicon ***L***. The main point of comparison in the current research is the accuracy $A(S,T,FL)$, which corresponds to the accuracy obtained by the *baseline* lexicon, i.e. all frequent words from the source domain.

Thus we can define the main measure in the current experiment:

$$\Delta(S,T,L) = A(S,T,L) - A(S,T,FL)$$

This is the difference between the accuracy obtained with the lexicon L and baseline lexicon FL, during the transfer from source domain S to target domain T. We also use the averaged variant of this measure:

$$\overline{\Delta}(L) = \frac{1}{|D|} \sum_{(S,T) \in D} \Delta(S,T,L)$$

In our case $|D| = 9$.

6.3 Main results

We report all results in this section using first 4000 words in the general lexicon and domain specific lexicons. This is the maximum amount of words with rather reliable intrinsic precision values ~70% in the general lexicon (see Section 5). We also provide the results of cross-domain sentiment classification quality with the other threshold values in general and domain specific lexicons on the Figure 2 and 3.

On all tasks the general sentiment lexicon performs on bar or better than the other feature sets. In Table 5 and Table 6, we summarize the comparison results of cross-domain classification using different feature sets.

A	**B->C**	**M->C**	**P->C**	**B->P**	**M->P**	**C->P**	**M->B**	**P->B**	**C->B**
FL	74.0	72.55	78.65	70.05	70.5	79.9	78.15	65.1	66.5
SDL	76.1	75.2	75.55	73.45	71.15	78.85	79.0	64.3	66.9
GL	76.1	75.7	81.9	73.35	72.55	79.8	78.05	66.6	67.2

TABLE 5 – The accuracy of cross-domain classification

Δ	**B->C**	**M->C**	**P->C**	**B->P**	**M->P**	**C->P**	**M->B**	**P->B**	**C->B**	$\overline{\Delta}$
SDL	2.1	2.65	-3.1	3.4	0.65	-1.05	0.85	-0.8	0.4	0.57
GL	2.1	3.15	3.25	3.3	2.05	-0.1	-0.1	1.5	0.7	1.76

TABLE 6 – The difference with baseline of cross-domain classification

The results demonstrate the effectiveness of the general meta-domain sentiment lexicon. In the Table 6 one can see that for some domain pairs our lexicons show significantly better results than the baseline. The average difference over all domain pairs between **FL** (baseline) and **GL** is 1.76%.

In some domain pairs the difference is very small or even negative. We connect this issue with the similarity of the domain lexicons in general (Ponomareva & Thelwall, 2012) and sentiment lexicons in particular. Sometimes sentiment words from one domain can be utilized in the other one, but not vice versa.

We suppose that such a general lexicon for the product meta-domain can serve as a good source of sentiment seed words to generate domain-specific vocabularies in a lot of specific domains.

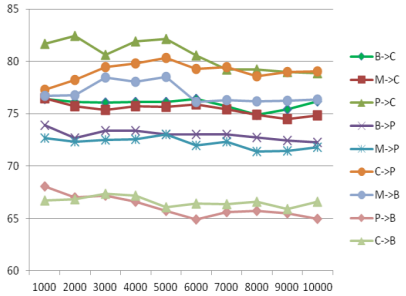


FIGURE 2 – The dependence of the classification quality on the threshold in the general lexicon

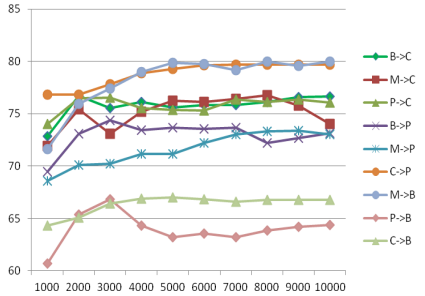


FIGURE 3 – The dependence of the classification quality on the threshold in the domain specific lexicons

Conclusion and perspectives

In this paper, we described a method for sentiment lexicon extraction for any domain on the basis of several domain-specific text collections. We utilized our algorithm in different domains and showed that it had good generalization abilities. We combined sentiment lexicons from various domains and constructed the general meta-domain sentiment lexicon for products and services. This lexicon was evaluated intrinsically, with P@1000 = 91.4% and extrinsically in the cross-domain classification task. The sentiment classification algorithm based on the meta-domain sentiment lexicon outperformed all baselines and proved usefulness of the constructed resource. Besides, this meta-lexicon can be a useful source of sentiment seeds for sentiment lexicon extraction in new domains of products and services.

We extracted such a general lexicon for Russian language, for which sentiment analysis resources practically do not exist. We plan to make our general lexicon for the product meta-domain publicly available.

Acknowledgments

This work is partially supported by RFBR grant N11-07-00588-a.

References

Abdul-Mageed M., Diab M., Korayem M. (2011). Subjectivity and Sentiment Analysis of Modern Standard Arabic. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics*, number 3, pp. 587-591.

Ahmad K., Gillam L., Tostevin L. (1999). University of Surrey participation in Trec8: Weirdness indexing for logical documents extrapolation and retrieval *In the Proceedings of Eigth Text Retrieval Conference (Trec-8)*.

Aue A. and Gamon M. (2005). Customizing sentiment classifiers to new domains: A case study. In *International Conference on Recent Advances in Natural Language Processing*, Borovets, BG.

Banea C., Mihalcea R., Wiebe J. and Hassan S. (2008). Multilingual subjectivity analysis using machine translation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Blitzer J., Dredze M., Pereira F. (2007) Biographies, bollywood, boom-boxes and blenders: Domain adaptation for sentiment classification. In *Proceedings of ACL 2007*, pp. 440–447.

Bollegala D., Weir D. and Carroll J. (2011) Using multiple sources to construct a sentiment sensitive thesaurus for cross-domain sentiment classification. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, Portland, Oregon. pp. 132-141.

Callan J.P., Croft W.B., Harding S.M. (1992). The INQUERY Retrieval System. In Proceedings of *3rd International Conference on Database and Expert Systems Applications* / A.M. Tjoa and I. Ramos (eds.). – Springer Verlag, New York, pp.78-93.

Chetviorkin I. and Loukachevitch N. (2011). Three-way movie review classification. In *Proceedings of the International Conference on Computational Linguistics Dialog*, pp 177-186.

Choi Y. and Cardie C. (2009). Adapting a polarity lexicon using integer linear programming for domain-specific sentiment classification. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pp. 590–598.

Clematide S., Klenner S. (2010) Evaluation and extension of a polarity lexicon for German. In *WASSA-workshop held in conjunction with ECAI-2010*, pp 7-13.

Hatzivassiloglou V. and McKeown K. R. (1997). Predicting the semantic orientation of adjectives. In *Proceedings of ACL-97*, pp. 174–181, Madrid, ES.

He B., Macdonald C., He J., and Ounis I. (2009). An effective statistical approach to blog post opinion retrieval. In *Proceedings of the 17th ACM CIKM*, pp. 1063-1072.

Jijkoun V., de Rijke M. and Weerkamp W. (2010). Generating focused topic-specific sentiment lexicons. In *Proceedings of ACL '10*, pp. 585–594.

Kanayama H. and Nasukawa T. (2006). Fully automatic lexicon expansion for domain-oriented sentiment analysis. In *EMNLP '06*, pp. 355–363, Morristown, NJ, USA.

Lau R., Lai C., Bruza P. and Wong K. (2011). Pseudo Labeling for Scalable Semi-supervised Learning of Domain-specific Sentiment Lexicons. In *20th ACM Conference on Information and Knowledge Management*.

Mihalcea R., Banea C. and Wiebe J. (2007). Learning multilingual subjective language via cross-lingual projections. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pp. 976–983, Prague, Czech Republic.

Pan S. J., Ni X., Sun J-T, Yang Q. and Chen Z. (2010). Cross-Domain Sentiment Classification via Spectral Feature Alignment. In *Proceedings of the World Wide Web*

Conference. pp. 751-760, New York, USA.

Pang B., Lee L. (2008). Opinion mining and sentiment analysis. Foundations and Trends® in Information Retrieval. *Now Publishers*.

Perez-Rosas V., Banea C. and Mihalcea R. (2012). Learning Sentiment Lexicons in Spanish. In *Proceedings of the Eight International Conference on Language Resources and Evaluation (LREC'12)*.

Ponomareva N. and Thelwall M. (2012): Bibliographies or blenders: Which resource is best for cross-domain sentiment analysis? In *Proceedings of the 13th Conference on Intelligent Text Processing and Computational Linguistics*.

Qiu G., Liu B., Bu J. and Chen C. (2011). Opinion word expansion and target extraction through double propagation. *Computational Linguistics*, 37(1).

Read J., Carroll J. (2009). Weakly Supervised techniques for domain independent sentiment classification. In *Proceedings of the first International CIKM Workshop on Topic-Sentiment Analysis for Mass Opinion Measurement*, pp. 45-52.

Steinberger J., Lenkova P., Ebrahim M., Ehrmann M., Hurriyetogly A., Kabadjov M., Steinberger R., Tanev H., Zavarella V. and Vazquez S. (2011). Creating Sentiment Dictionaries via Triangulation. In *Proceedings of the 2nd Workshop on Computational Approaches to Subjectivity and Sentiment Analysis, ACL-HLT 2011*, pp. 28–36,

Taboada M., Brooke J., Tofiloski M., Voll K. and Stede M. (2011). Lexicon-based methods for Sentiment Analysis. *Computational linguistics*, 37(2).

Wiebe J. and Riloff E. (2005). Creating subjective and objective sentence classifiers from unannotated texts. In *Proceedings of CICLing 2005*. pp. 486-497.

Wu Q., Tan S. and Cheng X. (2009). Graph ranking for sentiment transfer. In *Proceedings of ACL-IJCNLP 2009*, pp. 317–320.

Problems in Evaluating Grammatical Error Detection Systems

Martin CHODOROW[1] *Markus DICKINSON*[2]
Ross ISRAEL[2] *Joel TETREAULT*[3]
(1) Hunter College and the Graduate Center, City University of New York, New York, NY 10065, USA
(2) Indiana University, Department of Linguistics, Bloomington, IN 47405, USA
(3) Educational Testing Service, Rosedale Road, Princeton, NJ 08540, USA
mchodoro@hunter.cuny.edu, md7@indiana.edu, raisrael@indiana.edu,
jtetreault@ets.org

ABSTRACT
Many evaluation issues for grammatical error detection have previously been overlooked, making it hard to draw meaningful comparisons between different approaches, even when they are evaluated on the same corpus. To begin with, the three-way contingency between a writer's sentence, the annotator's correction, and the system's output makes evaluation more complex than in some other NLP tasks, which we address by presenting an intuitive evaluation scheme. Of particular importance to error detection is the skew of the data – the low frequency of errors as compared to non-errors – which distorts some traditional measures of performance and limits their usefulness, leading us to recommend the reporting of raw measurements (true positives, false negatives, false positives, true negatives). Other issues that are particularly vexing for error detection focus on defining these raw measurements: specifying the size or scope of an error, properly treating errors as graded rather than discrete phenomena, and counting non-errors. We discuss recommendations for best practices with regard to reporting the results of system evaluation for these cases, recommendations which depend upon making clear one's assumptions and applications for error detection. By highlighting the problems with current error detection evaluation, the field will be better able to move forward.

KEYWORDS: grammatical error detection, system evaluation, evaluation metrics.

Proceedings of COLING 2012: Technical Papers, pages 611–628,
COLING 2012, Mumbai, December 2012.

1 Introduction

With hundreds of millions of people worldwide learning second, or even third, languages (Leacock et al., 2010), there is a large and growing need for NLP systems that automatically detect and correct the grammar and word usage errors that learners make. In response to this need, NLP researchers have developed tools to target errors involving articles (Han et al., 2006), prepositions (Tetreault and Chodorow, 2008b), particles (Dickinson et al., 2011), verb forms (Lee and Seneff, 2008), and collocations (Dahlmeier and Ng, 2011). Research in this field has surged over the last few years, culminating notably in two recent "Helping Our Own" (HOO) Shared Tasks (Dale and Kilgarriff, 2010) and (Dale et al., 2012), which are concerned with automated error correction in texts authored by non-native speakers of English working in NLP. However, despite this high level of activity and interest, there is relatively little consensus on how best to evaluate grammatical error detection/correction systems. This makes it hard to measure performance and compare systems, which is essential for progress in the field.

The goal of this paper is to draw attention to the many evaluation issues in error detection that have largely been overlooked in the past and which make it hard to draw meaningful comparisons between different approaches, even when they are evaluated on the same corpus. Of particular importance is the skew of the data – the low frequency of errors as compared to non-errors – which distorts some traditional measures of performance and limits their usefulness. Many issues in evaluation remain outside the scope of this paper, and we will not have recommendations for every problem that we discuss. However, we feel that highlighting the problems will help to move the field forward.

The lack of consensus in evaluation is due, in large part, to the nature of the error detection task. Consider (1a), a sentence that a learner of English might write.[1] It could be corrected by a human annotator or a system as (1b), (1c), (1d), or any number of other grammatical variants.

(1) a. Book of my class inpired to me.
    b. A book in my class inspired me.
    c. Books for my class inspired me.
    d. The books of my class were inspiring to me.

How we count the number, type, and scope of errors in the learner's sentence depends on the relation between what was written and the annotator's correction. However, on the task of error detection, how we score the performance of an NLP system depends on the relation between the system's output and **both** the learner's sentence and the annotator's correction. This three-way contingency makes the evaluation inherently more complex than in some other NLP tasks where the only comparison is between system and annotator. In Section 3.1, we describe the standard measures of system performance (Accuracy, Precision, Recall, F-measure), and in Section 3.2, we map a three-way contingency table of matches and mismatches among writer, annotator, and system onto the standard evaluation measures. Section 3.3 examines general problems that arise in using these measures, and Section 3.4 describes the variety of

[1] For the sake of simplicity, we use constructed examples of grammatical errors to illustrate the evaluation issues discussed in this paper. One example of a sentence with grammatical errors, taken from the first Helping Our Own Shared Task (Dale and Narroway, 2011), is: *First of all, we focus on an analysis on sentences in product reviews regarding the two views: personal and impersonal views.* where *analysis on* can be corrected to *analysis of*, *regarding the two* can be corrected to *regarding two* and *impersonal views* can be corrected to *impersonal*.

metrics found in published reports of system performance. Sections 3.5 and 3.6 deal with three issues that are particularly vexing for error detection: how to specify the size or scope of an error, how to properly treat errors as graded rather than discrete phenomena, and how to count non-errors. Each subsection contains recommendations for best practices with regard to reporting the results of system evaluation. Certain metrics are more or less appropriate depending on the type of task the system is used for, so we begin with a brief overview of applications of grammatical error detection.

2 Applications

It is not surprising that most applications of grammatical error detection are in education, where it has been used for student assessment and to support language learning. A more recent set of applications focuses on improving systems within the domain of NLP itself.

Automatically scoring essays Error detection is a fundamental component in most systems which perform automated essay scoring (Yannakoudakis et al., 2011; Attali and Burstein, 2006; Burstein et al., 2003; Lonsdale and Strong-Krause, 2003). The goal here is to find aspects of grammar and word usage related to overall text quality so that a holistic score, usually on a 5- or 6-point scale, can be generated. Essay scoring systems also measure the range of vocabulary, discourse structure, and the mechanics of writing (e.g., spelling) as predictors of the writing score. These systems are used for large scale high-stakes tests taken by native and non-native speakers, such as the Graduate Record Exam (GRE), and for tests of non-native proficiency, such as the Test of English as a Foreign Language (TOEFL).

Improving writing quality To date, most work that focuses on writing assistance tools has targeted non-native English writers. Grammatical error detection in this context is used to assist one in producing better essays or other documents (Chodorow et al., 2010; Hirst and Budanitsky, 2005; Kukich, 1992). In the short term, the goal of providing this assistance is to improve the current document by highlighting errors and suggesting corrections. In the longer term, the goal is to help writers learn more about the language they are studying so that they can produce higher quality writing, as described next.

Assisting language learning An indicative example of assisting language learners can be found in various Intelligent Tutoring Systems, which provide learners with immediate feedback on their language constructions (Meurers, 2012; Heift and Schulze, 2007). Learners not only need to observe their errors but also to understand the meta-linguistic properties of such errors. Furthermore, systems must track not just what the learner is doing incorrectly, but also what the learner is doing correctly, in order to properly model behavior (Amaral and Meurers, 2007). For both purposes, error detection is needed to detect errors, suggest corrections, and provide information about the linguistic properties of the writer's mistakes.

Applications within NLP Grammatical error detection can be a useful component in correcting and evaluating text generated in various NLP applications. Among other applications, it can be used to detect errors in machine translation (MT) output (such as in Knight and Chander (1994) and Peng and Araki (2005)) and can even be incorporated in quality metrics to assess MT (Parton et al., 2011). In these contexts, an NLP system takes the place of the writer, but the goal is similar, namely to produce more error-free language.

3 Measuring Performance

3.1 Traditional Evaluation Measures

To illustrate the traditional measures of NLP system evaluation, consider as an example the task of comma restoration (see, e.g. Shieber and Tao, 2003), in which the commas are removed from a well-edited text (the gold standard) and a system attempts to restore them by predicting their locations. For evaluations involving a binary distinction such as this one (the presence vs. absence of a comma), a comparison between the system's output and the annotator's judgments (the gold standard) can be organized as a two-by-two contingency table, shown in Figure 1. Presence of a comma is the target or positive class, and absence is the negative class. Positions in the text where both the system and the gold standard indicate that there should be a comma are true positives (TP); those where both indicate no comma are true negatives (TN); those where the system predicts a comma but the gold standard shows no comma are false positives (FP); and those positions where the system predicts no comma but the gold standard shows a comma are false negatives (FN). Given counts for these four contingencies, it is straightforward to calculate measures such as Accuracy (A), Precision (P), Recall (R), true-negative rate (TNR), and F-score (F_1), as shown in Figure 2.

		Annotation (Gold Standard)	
		Comma (+)	No comma (-)
System prediction	Comma (+)	TP	FP
	No comma (-)	FN	TN

Figure 1: The basis for typical NLP system evaluation

Accuracy (A) $= \frac{TP+TN}{TP+TN+FP+FN}$
Precision (P)$= \frac{TP}{TP+FP}$
Recall (R)$= \frac{TP}{TP+FN}$
True Negative Rate (TNR) $= \frac{TN}{TN+FP}$
F-measure (F_1) $= 2 \cdot \frac{P \cdot R}{P+R}$

Figure 2: Evaluation metrics

3.2 Error Detection and the Three-Way Contingency Table

Now consider a task that is similar to comma restoration, the task of comma error detection (Israel et al., 2012), in which a system seeks to find and correct errors in the writer's usage of commas. For this task, the positive class is not the presence of a comma but rather an error of the writer's that involves a comma. Therefore, it is necessary to compare what the writer has written to an annotator's judgment, and only if there is a mismatch between the two do we have an error (the positive class); when writer and annotator agree, the case is a non-error (the negative class). The traditional 2x2 table is no longer sufficient to represent all of the contingencies, which must instead be laid out in the more complex three-way table of Figure 3.

Figure 3 has the virtue of being complete in the sense that it shows all 2x2x2 (=8) possible combinations of binary values for the writer's, annotator's, and system's output. However, we believe that the scheme in Figure 4, which we refer to as the Writer-Annotator-System (WAS)

		Annotation (Gold Standard)			
		Comma		No comma	
Writer's form		Comma	No comma	Comma	No comma
Pos (+) / Neg (-) class		No Error (-)	Error (+)	Error (+)	No error (-)
System prediction	Comma	TN	TP	FN	FP
	No comma	FP	FN	TP	TN

Figure 3: The basis for error detection evaluation

evaluation scheme, is a simpler and more intuitive way of characterizing the relationships. To determine whether a case is a TN, TP, FN, or FP, the three sources of input are compared to see which, if any, agree. The X, Y, and Z in the figure are variables that can be replaced with any type of token; what is important is whether or not they match.

	Written	Annotated	System
TN	X	X	X
FP	X	X	Y
FN	X	Y	X
TP	X	Y	Y
*	X	Y	Z

Figure 4: *WAS* evaluation scheme

Note that the first row in Figure 4 (with answers of X from each input) can apply to multiple outcomes, as in preposition selection, or to a binary classification task, such as the presence or absence of a comma; we focus on the binary task, to be consistent with the previous figures. If all three inputs agree, the case is tallied as a TN. If the System agrees with the Annotator but the Writer does not, then it is a TP. For binary classification tasks, the first four rows are sufficient for both detection and correction, as detecting an error naturally suggests the correct answer. For tasks that involve more than two classes, further distinctions are needed to distinguish detection and correction. The final row in the figure represents a situation where all three sources provide different answers. This is marked with a * because it belongs in different categories depending on whether the evaluation is for detection or correction. For detection, this is a TP because the system is flagging an error, even if it does not produce the correct answer according to the annotator or gold standard (essentially this reduces to X Y Y). From the TP, FP, FN, and TN counts, all the measures in Figure 2 can be calculated.[2]

As far as we are aware, evaluation has not previously been schematized as in Figure 4, and we recommend setting up one's evaluation in such a way whenever possible. For the rest of the paper, we will assume TP, FP, FN, and TN as clarified by this scheme.

3.3 Traditional Metrics and the Problem of Skewed Data

At first glance, A (accuracy) seems to be the most straightforward and easily interpreted measure of system performance, but when the distribution of positive and negative classes is highly skewed, as is most often the case with writing errors, accuracy can be quite misleading. For example, Han et al. (2006) report a rate of usage errors of 13% for prepositions in a

[2]For error correction, X Y Z is both a FP (system $\neq$ writer) and FN (system $\neq$ annotator).

corpus of English essays written by native speakers of Chinese, Japanese, and Russian. For that corpus, a baseline system that always predicts "no error" will have an A of 87%, reflecting the overwhelming proportion of negative cases. Similar or even greater levels of skew are commonly found in the error detection literature (Leacock et al., 2010). With high baselines such as these, it is often difficult to see, from a single summary measure such as A, just how a system is performing, especially on the non-majority class, in this case, the errors.

The metrics most frequently reported in NLP research are intended to address this problem. R compares the number of errors the System correctly detects (TP) to the total number of errors in the Annotated gold standard (TP+FN); P compares TP to the total number of errors that the System reports (TP+FP); and F_1 is the harmonic mean of R and P. Unfortunately, all three measures are affected by the proportion of cases that are annotated as errors in the gold standard (referred to as the **prevalence** of the errors, which is equal to (TP+FN)/N, where N is the total number of cases, i.e., N = TP+TN+FP+FN) and by the proportion of cases that are reported by the System as errors (referred to as the **bias** of the System, which is equal to (TP+FP)/N). Powers (2012) demonstrates how a system that performs no better than chance will nonetheless show an increase in R when prevalence increases and an increase in P when bias increases. To understand this behavior, we must consider what it means to perform at chance.

If the class labels Error and No Error are assigned to cases independently by the Annotator and the System, then these labels are expected to match a proportion of the time by chance alone - a proportion equal to the product of their probabilities. For example, the expected proportion of TP matches is equal to the product of the proportion of cases assigned the Error label by the Annotator (i.e., the prevalence) and the proportion of cases assigned the Error label by the System (i.e., the bias). This is illustrated in Figure 5, with the value in each cell of the table equal to the product of the marginal proportions in the cell's row and column (e.g., expected proportion of TP = .20 x .20 = .04; expected proportion of FP = .20 x .80 = .16; etc.).

		Annotation (Gold Standard)		Marginal
		Error (+)	No Error (-)	proportion
System prediction	Error (+)	TP .04	FP .16	.20 (bias)
	No Error (-)	FN .16	TN .64	.80
Marginal proportion		.20 (prevalence)	.80	

Figure 5: Performance of a hypothetical system with Accuracy = .68 and Kappa = .00

The A which is expected by chance, E(A), is .04 + .64 = .68, the sum of the expected proportions of TP and TN. Cohen's kappa (κ) statistic (Cohen, 1960), shown in Figure 6, uses E(A) to correct the observed A found between Annotator and System. This provides a measure of performance over and above chance.

$$\kappa = \frac{A - E(A)}{1 - E(A)}$$

Figure 6: Formula for Cohen's kappa

To illustrate the use of κ, consider the hypothetical data in Figure 7, which has the same marginal proportions, and therefore the same E(A), as Figure 5. It shows a System that has an observed A of .10 + .70 = .80, but, when corrected for chance agreement, the Accuracy value (i.e., κ) is (.80 - .68)/(1.00 - .68) = .38. This says that after removing the proportion of cases

expected to show agreement by chance alone, the System is correct (agrees with the Annotator) on 38% of the remaining cases.

		Annotation (Gold Standard)		Marginal
		Error (+)	No Error (-)	proportion
System prediction	Error (+)	TP .10	FP .10	.20 (bias)
	No Error (-)	FN .10	TN .70	.80
Marginal proportion		.20 (prevalence)	.80	

Figure 7: Performance of a hypothetical system with Accuracy = .80 and Kappa = .38

Now consider the hypothetical data in Figure 8, which show a system that is performing no better than expected by chance. Its κ of .00 is the same as the κ of Figure 5, but the A values for the two sets of data are quite different, .68 vs. .54. This illustrates the earlier observation that A can be misleading as a measure of system performance, but A is not alone in this regard. R, P, and F_1 are also affected by changes in the marginal proportions.

		Annotation (Gold Standard)		Marginal
		Error (+)	No Error (-)	proportion
System prediction	Error (+)	TP .12	FP .18	.30 (bias)
	No Error (-)	FN .28	TN .42	.70
Marginal proportion		.40 (prevalence)	.60	

Figure 8: Performance of a hypothetical system with Accuracy = .54 and Kappa = .00

For the data in Figure 5, R is .04/(.04 + .16) = .20, and P and F_1 also have values of .20. These differ from those of Figure 8, where R is .12/(.12 + .28) = .30, P is .12/(.12 + .18) = .40, and F_1 is .34. A comparison of the marginal proportions in the two figures shows that an increase in bias can be expected to increase R, while an increase in prevalence can be expected to increase P, even when systems are performing at chance levels. These are not desirable properties for system performance measures. As Powers (2012, p. 345) points out, "traditional evaluation measures used in Computational Linguistics (including Error Rates, Accuracy, Recall, Precision, and F-measure) are of limited value for unbiased evaluation of systems, and are not meaningful for comparison of algorithms unless both the dataset and algorithm parameters are strictly controlled for skew (Prevalence and Bias)".

Sensitivity to skew may be especially problematic for evaluation of error detection/correction systems. Until recently (Dale et al., 2012), error-annotated learner corpora were not generally available for use in system testing. As a result, systems that were developed by different researchers were tested on different datasets having different error rates (Leacock et al., 2010). The variability in error rates (prevalence) could be due in part to different populations of learners with different native languages and different levels of proficiency in their second language. In addition to the variability in prevalence, there is also variability in bias when parameters of error detection systems are tuned, as they typically are, on development data to optimize performance. For example, if a system's decision to output an Error label is based on a probability that it computes for each case, then a probability threshold can be set which will increase or decrease the number of Error labels that will be output. Lower thresholds correspond to higher bias, and higher thresholds to lower bias.

Because of sensitivity to bias, system evaluation is limited if it only examines performance at one setting of a threshold. By varying the threshold through the full range of values, P-R curves can be generated to give a better picture of performance. In a similar manner, other types of curves can be produced, such as the Receiver Operating Characteristic (ROC), which plots the true positive rate (TP/(TP+FN)) against the false positive rate (FP/(FP+TN)). Area under the ROC curve has been used for evaluating and ranking machine learning models (Bradley, 1997), but Kaymak et al. (2010) argue that the area under the κ curve (κ plotted against the false positive rate) is superior to the area under the ROC curve because it accounts for skew in the data while the area under ROC does not.

While κ has advantages over R and P, it too has come under increased scrutiny. Although it properly takes changes in prevalence and bias into account, κ is known to favor correct classifications of the minority class (for error detection, Errors) over those of the majority class (Non-errors). Kaymak, et al.(2010, p. 9) have argued that "this property of Kappa is actually a virtue for many real world classification problems in which it is more important to correctly classify the minority rather than the majority class." Error detection is arguably such a problem. Another issue is that there are several kappa-like reliability measures which take chance agreement into account, but it is not clear which one or ones are best. For example, Krippendorff (2004) criticizes the use of κ for measuring the reliability of annotators in content analysis because, for a given level of agreement, the value will actually increase when there are greater differences in the annotators' preferences for using the various category labels. Krippendorff's alpha measure is designed, in part, to address this problem. Recently, Powers (2012) has shown that in some situations, Cohen's kappa is inferior to another kappa-like variant formed from a different combination of the values of the contingency table (Powers, 2003). Finally, despite its attractive features, κ is not informative about system performance in the terms that an end user cares about because, as shown above, it does not reflect A, P, or R. This limits its value for decisions about whether a system is ready for use in real world applications.

3.4 Variety in Reporting Evaluation Measures

As a result of the many issues involved in evaluating grammatical error correction systems, it is perhaps unsurprising that there is little consensus about which evaluation metric to use. For example, Leacock et al. (2010) describe how, in 2008, there were three papers on preposition error detection and each used slightly different evaluation metrics making comparison impossible. More recently, Tetreault et al. (2010) and Dickinson et al. (2011) report P and R for work done on prepositions and particles, respectively. Gamon (2010) also used P and R, while Rozovskaya and Roth (2010b) and Rozovskaya and Roth (2011) reported A of usage in a test corpus before and after corrections were made by their system. Sometimes, different metrics have been used for two aspects of the same task, as in Han et al. (2010), where A was reported for determining the presence vs. absence of a preposition, but P and R were used to evaluate performance in detecting when an incorrect preposition was used by the writer. Some researchers have even looked towards more holistic, sentence-level metrics like BLEU and METEOR (see, e.g. Park and Levy, 2011), or modifying measures like P and R to consider entire sentences, as in Gamon (2011). In the two HOO Shared Tasks, P, R and F_1 were all reported, but no preference was given to which one to use to rank the systems in the end.

Recommendations It is clear that no single measure of performance is best for all purposes; each one has its own set of advantages and disadvantages. However, all of the measures, aside from BLEU and METEOR, are based on the same four values, the counts for TP, FP, FN, and TN. We recommend reporting these four in addition to any metrics derived from them. This will enable readers to calculate other measures that the authors of a particular paper did not choose to include (e.g., Matthews Correlation), and will provide the flexibility needed to accommodate new developments in evaluation as debates over how best to measure performance are resolved. In other words, it will make it possible for future readers to go back to any publications that report these four values and retrofit new measures to old data. We also recommend κ (and the κ curve) for its ability to take the skew of the class and the label distributions into account. When reporting P and R, prevalence and bias should also be reported. For a given test corpus, prevalence will be constant, but bias will vary when threshold values change, so bias should be represented, perhaps along the horizontal axis, in displaying P, R, and F_1 curves.

3.5 Positives

In error detection, the target, or positive, class consists of an error in the writer's text. Defining a positive is complicated both by the variability in the unit size and scope of errors and by the variability in human judgments of errors.

3.5.1 Unit Size

Often there is no simple 1:1:1 correspondence between the writer's erroneous string, the annotator's correction, and the type of error the system was designed to detect. Looking back at (1a), for example, *Book* may be treated as an article error corrected by (1b), a number error corrected by (1c), or both an article error and a number error corrected by (1d). Each diagnosis has implications for how the error will be counted and mapped onto a correction.

Even when there is no ambiguity about the type of error, decisions must still be made about the size of the units over which the error is defined. In (2a), the noun-verb number disagreement involves two discontinuous substrings (e.g., *The book . . . inspire*), either one of which could be changed to correct the error, as in (2b) and (2c). This highlights the question of what size of unit is most appropriate for measuring errors and corrections: is it the morpheme, the word, the phrase, or the string? In this case, a strictly token-based definition may identify only a single word as the error (either *book* or *inspire*) or identify two separate errors, whereas a string-based definition can refer to both words as a single error.

(2) a. The **book** in my class **inspire** me.
b. The book in my class **inspires** me.
c. The **books** in my class inspire me.

There are several issues involved in choosing the unit size. First, the definition of unit size affects whether a system is given credit for finding all and only the errors, as these examples illustrate. Identifying *inspire* as an error may or may not be sufficient; identifying both words may be overkill. Defining error detection metrics in terms of edit distance mitigates this problem for evaluation comparisons (Dahlmeier and Ng, 2012), as correcting *inspire* to *inspires* is handled the same as *book ... inspire* corrected to *book ... inspires*. In essence, edit distance measures (EDMs) compare the system output to the correct *string*, ignoring exactly how it was derived.

Moreover, EDMs naturally handle multiple, overlapping errors, a problem for systems that target only a specific error type (Gamon, 2010; Rozovskaya and Roth, 2010a). Taking an example from Dahlmeier and Ng (2012), the sequence ... *development set similar with test set* ... can be corrected as a preposition selection error *with* → *to* and an adjacent article omission error $\epsilon \rightarrow$ *the*, or it can also be corrected as a single error (*with* → *to the*). Because EDMs can calculate the best match between the system's edits and those of the gold standard, they make it possible to define errors over multiple units, not just words but sequences of words as well.

Despite the benefits of basing metrics on edit distance, issues remain, depending upon the specific application of error detection or correction (see section 2). The selection of unit size is consequential for the specificity of feedback that a system can give to the writer. If *book* and *inspire* are not linked, for instance, it is harder to provide feedback on subject-verb agreement. Or if *with* → *to* and $\epsilon \rightarrow$ *the* really are different error types, then feedback should deal with them separately. Focusing on matching the string, but not on how it was derived, works well when the task is to produce a final corrected string, but it comes with a cost for applications such as intelligent tutoring systems, where the goal is to determine which types of errors the writer has made.

Additionally, errors can be defined over multiple units, but are still constrained by what the base units are. Consider if the learner had written *inpire* in (2a), thereby adding a misspelling on top of an agreement error. If the base units are words, then there is one correction to make, mapping *inpire* to *inspires*, for what are clearly two distinct errors. Character-level edits would handle this particular problem, but are difficult to work with as soon as the error is on the level of the morpheme (e.g., the two-character, one-morpheme *-es*), the word, or the sentence. The issue is that errors are defined across types of linguistic units; some errors are morpheme-level, some phrase-level, and it is difficult to group them all together.[3] EDMs can only work from the smallest unit they define, and in general the units seem to be variable.

Finally, one benefit of EDMs is that they easily fit into calculations of P and R. However, by defining matches in terms of string distance, there is no way to talk about true negatives. Errors can be of variable sizes, but this means that the size of non-errors is, in a sense, undefined. Thus, alternative measures like κ cannot be calculated.

The issue of unit size becomes more complicated when error types interact. In (3), for instance, if a learner's preposition *in* should be corrected to *to*, then the article can be dropped. If, however, the preposition is not changed, then dropping the article makes the sentence worse. This is because one correction leads to another, as noted by many doing error annotation (Boyd, 2010; Hana et al., 2010; Lee et al., 2012; Dickinson and Ledbetter, 2012). As far as we aware, no system evaluation accounts for this problem, and we do not solve it, either.

(3) Every day we go { in the school → to school }

Recommendations How one treats positives depends upon one's purpose(s). If matching to a correct string and P/R are all that is required, EDMs have several excellent properties, including being able to account for errors of variable type and size (above the level of the base unit). If feedback is desired, EDMs are not a good choice because each error type must be taken into account separately (which is largely done in Dale and Narroway (2011)). EDMs should not

[3]While the typical base syntactic unit varies for different languages, e.g., morphemes for Hungarian (Dickinson and Ledbetter, 2012), the principles discussed here apply across languages.

be used if the goal is to control for skew in the data by calculating measures such as κ, which require counts of true negatives (see section 3.6).

3.5.2 Reliability: Clear-Cut vs. Contentious Errors

For some errors, such as the extraneous preposition *to* in (1a), the judgment is unequivocal, but for others, such as the writer's choice to use *of* instead of another preposition, judgments are likely to be contentious. Arguably, an error detection system which fails to mark *to* as an error is performing more poorly than one which fails to mark *of*, but in current practice, all errors are generally assigned equal weight (though, see the "with bonus" correction in Dale and Narroway, 2012). Tetreault and Chodorow (2008a) examined some problems posed by disagreements in judgment. Using a cloze task, they asked two trained professional annotators to fill in a blank in well-formed sentences where a preposition had been removed. The results showed only 75% agreement on the prepositions that they filled in. When the same experts were asked to mark preposition usage errors in non-native writing, the system's evaluation differed by as much as 10% in P depending on which expert's annotations were used as the gold standard. Madnani et al. (2011) have argued that a better approach to system evaluation when errors are not clear-cut, as in preposition selection, is to treat them as graded and to assign them weights based on the distribution of judgments for each error obtained through crowdsourcing. Using this approach for system evaluation, if 80% of judgments for a case are Error and the system labels it as an Error, then the TP count is incremented by .80 and the FP count by .20. If, instead, the system outputs Non-error for this case, the FN count is incremented by .80 and the TN count by .20. Madnani et al. (2011) present weighted versions of the P and R formulas, and they argue that these measures are more stable than their unweighted counterparts, especially when test sets are small.

Recommendations For error types that are not judged with high reliability, authors should consider evaluation metrics based on weighted counts where the weights reflect the distributions of human judgments. The standard metrics otherwise apply.

3.6 Negatives

The non-target, or negative, class in error detection consists of the non-errors in the writer's text. At first glance, this would simply seem to be the complement of the positive class. However, an appropriate set of non-errors for system evaluation is not that easily specified, even for errors of only a single token (see discussion in section 3.5 for variable-length errors). Consider the task of enumerating the article non-errors in (1a). No article should be inserted before *of*, *my*, *class*, *inspired*, *to*, *me*, and the punctuation at the end of the sentence. Should all 7 of these positions be treated as negatives? Should only the noun phrases be counted? Most can be trivially ruled out as sites for articles (e.g., **a my*, **a me*). Similarly, when detecting missing prepositions, as in (4), performance measures can be affected not just by the system's ability to insert the proper preposition (*of*) in the correct position (after *fond*) but also by how we count the positions where a preposition is not inserted by the system. In (4), are there four positions where prepositions were not inserted or are there zero positions?

(4) He is fond beer .

Decisions of this sort have consequences for measures of system performance. When positions are included that can always be correctly identified as non-errors by trivial means, then the result will be an inflation in the count of TN, but no change in the other counts of the contingency table. This will not affect R, P, or F_1, but it will cause both A and κ to increase. This is illustrated in Figure 9 and Figure 10. Figure 9 is the same as Figure 5 except that the counts, based on an N of 100, have been included in the cells of the table. Figure 10 shows the effect of increasing the TN count by 100. This changes the proportions and, as a result, both A and κ increase.

		Annotation (Gold Standard)		Marginal
		Error (+)	No Error (-)	proportion
System prediction	Error (+)	TP 12(.12)	FP 18(.18)	.30 (bias)
	No Error (-)	FN 28(.28)	TN 42(.42)	.70
Marginal proportion		.40 (prevalence)	.60	

Figure 9: Cell counts (proportions) of a hypothetical system: Accuracy = .54 and Kappa = .00

		Annotation (Gold Standard)		marginal
		Error (+)	No Error (-)	proportion
System prediction	Error (+)	TP 12(.06)	FP 18(.09)	.15 (bias)
	No Error (-)	FN 28(.14)	TN 142(.71)	.85
Marginal proportion		.20 (prevalence)	.80	

Figure 10: Cell counts (proportions) of a hypothetical system: Accuracy = .77 and Kappa = .21

Recommendations When using κ or other measures that rely on true negatives, authors should be very explicit about what constructions or parts of speech were included in the negative class. If the TN count was based, in part, on a set of heuristics that identified obvious negative cases even before a statistical classifier was used, then the number of TNs identified in this way should also be reported so that the metrics can be adjusted appropriately.

Conclusion

We have presented the *WAS* evaluation scheme for mapping the writer's, annotator's, and system's output onto traditional NLP evaluation measures, and we have argued that the choice of metric should take into account factors such as the skew of the data and the type of application that the system will be used for. κ can provide a way to evaluate systems across differences in skew, and new kappa-related measures are being developed. Our recommendations with regard to reporting results of system performance can be summed up simply: Wherever possible, provide the reader with the counts for the four cells of the contingency table in addition to metrics that are derived from them. They are the basis for current and future measures of performance. Along with these values, describe what N, the total number of cases, consists of by listing the criteria that determined which constructions the system has analyzed. As noted earlier, no single metric is best for all purposes. A measure that is useful for system comparisons may not be the best to determine if a system is good enough to be deployed operationally, and measures of overall sentence quality, such as edit distance, may not adequately support language learning. It is our hope that, by following these guidelines, reporting in the field of grammatical error detection will be more informative and more consistent.

Acknowledgments

We would like to thank Michael Heilman, Aoife Cahill and the anonymous reviewers for their helpful comments.

References

Amaral, L. and Meurers, D. (2007). Conceptualizing student models for ICALL. In *Proceedings of the 11th International Conference on User Modeling*, Corfu, Greece.

Attali, Y. and Burstein, J. (2006). Automated essay scoring with e-rater v.2. *Journal of Technology, Learning, and Assessment*, 4(3).

Boyd, A. (2010). EAGLE: an error-annotated corpus of beginning learner German. In *Proceedings of LREC-10*, Malta.

Bradley, A. P. (1997). The use of the area under the ROC curve in the evaluation of machine learning algorithms. *Pattern Recognition*, 30:1145–1159.

Burstein, J., Chodorow, M., and Leacock, C. (2003). Criterion online essay evaluation: An application for automated evaluation of student essays. In *Proceedings of the Fifteenth Annual Conference on Innovative Applications of Artificial Intelligence*.

Chodorow, M., Gamon, M., and Tetreault, J. (2010). The utility of article and preposition error correction systems for English language learners: Feedback and assessment. *Language Testing*, 27(3):419–436.

Cohen, J. (1960). A coefficient of agreement for nominal scales. *Educational and Psychological Measurement*, 20(1):37–46.

Dahlmeier, D. and Ng, H. T. (2011). Correcting semantic collocation errors with L1-induced paraphrases. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 107–117, Edinburgh, Scotland, UK. Association for Computational Linguistics.

Dahlmeier, D. and Ng, H. T. (2012). Better evaluation for grammatical error correction. In *Proceedings of NAACL-HLT 2012*.

Dale, R., Anisimoff, I., and Narroway, G. (2012). HOO 2012: A report on the preposition and determiner error correction shared task. In *Proceedings of the Seventh Workshop on Building Educational Applications Using NLP*, pages 54–62, Montréal, Canada. Association for Computational Linguistics.

Dale, R. and Kilgarriff, A. (2010). Helping our own: Text massaging for computational linguistics as a new shared task. In *International Conference on Natural Language Generation*.

Dale, R. and Narroway, G. (2011). The HOO pilot data set: Notes on release 2.0. Technical report, Center for Language Technology, Macquarie University.

Dale, R. and Narroway, G. (2012). A framework for evaluating text correction. In Chair), N. C. C., Choukri, K., Declerck, T., Doğan, M. U., Maegaard, B., Mariani, J., Odijk, J., and Piperidis, S., editors, *Proceedings of the Eight International Conference on Language Resources and Evaluation (LREC'12)*, Istanbul, Turkey. European Language Resources Association (ELRA).

Dickinson, M., Israel, R., and Lee, S.-H. (2011). Developing methodology for Korean particle error detection. In *Proceedings of the Sixth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 81–86, Portland, OR.

Dickinson, M. and Ledbetter, S. (2012). Annotating errors in a Hungarian learner corpus. In *Proceedings of the 8th Language Resources and Evaluation Conference (LREC 2012)*, Istanbul, Turkey.

Gamon, M. (2010). Using mostly native data to correct errors in learners' writing: A meta-classifier approach. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*.

Gamon, M. (2011). High-order sequence modeling for language learner error detection. In *Proceedings of the 6th Workshop on Innovative Use of NLP for Building Educational Applications*.

Han, N.-R., Chodorow, M., and Leacock, C. (2006). Detecting errors in English article usage by non-native speakers. *Natural Language Engineering*, 12(2):115–129.

Han, N.-R., Tetreault, J., Lee, S.-h., and Ha, J.-y. (2010). Using an error-annotated learner corpus to develop and ESL/EFL error correction system. In *Language Resources and Evaluation Conference*.

Hana, J., Rosen, A., Škodová, S., and Štindlová, B. (2010). Error-tagged learner corpus of Czech. In *Proceedings of the Fourth Linguistic Annotation Workshop*, pages 11–19, Uppsala, Sweden.

Heift, T. and Schulze, M. (2007). *Errors and Intelligence in Computer-Assisted Language Learning: Parsers and Pedagogues*. Routledge.

Hirst, G. and Budanitsky, A. (2005). Correcting real-word spelling errors by restoring lexical cohesion. *Natural Language Engineering*, 11(1):87–111.

Israel, R., Tetreault, J., and Chodorow, M. (2012). Correcting comma errors in learner essays, and restoring commas in newswire text. In *Proceedings of NAACL-HLT 2012*.

Kaymak, U., Ben-David, A., and Potharst, R. (2010). AUK: A simple altenative to the AUC. Technical report, Erasmus Institute of Management, Erasmus School of Economics, Rotterdam, NL.

Knight, K. and Chander, I. (1994). Automated postediting of documents. In *Proceedings of the Twelfth National Conference on Artificial Intelligence (AAAI)*, pages 779–784, Seattle.

Krippendorff, K. (2004). Reliability in content analysis. *Human Communication Research*, 30(3):411–433.

Kukich, K. (1992). Techniques for automatically correcting words in text. *ACM Computing Surveys*, 24(4):377–439.

Leacock, C., Chodorow, M., Gamon, M., and Tetreault, J. (2010). *Automated Grammatical Error Detection for Language Learners*. Synthesis Lectures on Human Language Technologies. Morgan Claypool.

Lee, J. and Seneff, S. (2008). Correcting misuse of verb forms. In *Proceedings of ACL-08: HLT*, pages 174–182, Columbus, Ohio.

Lee, S.-H., Dickinson, M., and Israel, R. (2012). Developing learner corpus annotation for Korean particle errors. In *Proceedings of the Sixth Linguistic Annotation Workshop (LAW VI)*, pages 129–133, Jeju, Republic of Korea.

Lonsdale, D. and Strong-Krause, D. (2003). Automated rating of ESL essays. In *Proceedings of the HLT-NAACL 03 Workshop on Building Educational Applications Using Natural Language Processing*, pages 61–67.

Madnani, N., Tetreault, J., Chodorow, M., and Rozovskaya, A. (2011). They can help: Using crowdsourcing to improve the evaluation of grammatical error detection systems. In *Proceedings of the 49th Annual ACL Conference (Short Papers)*, Portland, USA.

Meurers, D. (2012). Natural language processing and language learning. In Chapelle, C. A., editor, *Encyclopedia of Applied Linguistics*. Blackwell. to appear.

Park, Y. A. and Levy, R. (2011). Automated whole sentence grammar correction using a noisy channel model. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1*, pages 934–944.

Parton, K., Tetreault, J., Madnani, N., and Chodorow, M. (2011). E-rating machine translation. In *Proceedings of the Sixth Workshop on Statistical Machine Translation*, pages 108–115, Edinburgh, Scotland. Association for Computational Linguistics.

Peng, J. and Araki, K. (2005). Correction of article errors in machine translation using web-based model. In *IEEE NLP-KE*, pages 393–397, Wuhan, China.

Powers, D. M. W. (2003). Recall and precision versus the Bookmaker. In *Proceedings of the International Conference on Cognitive Science*, pages 529–534, Sydney, Australia.

Powers, D. M. W. (2012). The problem with kappa. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 345–365, Avignon, France. Association for Computational Linguistics.

Rozovskaya, A. and Roth, D. (2010a). Annotating ESL errors: Challenges and rewards. In *Proceedings of the NAACL HLT 2010 Fifth Workshop on Innovative Use of NLP for Building Educational Applications*.

Rozovskaya, A. and Roth, D. (2010b). Training paradigms for correcting errors in grammar and usage. In *Proceedings of HLT-NAACL-10*.

Rozovskaya, A. and Roth, D. (2011). Algorithm selection and model adaptation for ESL correction tasks. In *Proceedings of ACL-2011*.

Shieber, S. M. and Tao, X. (2003). Comma restoration using constituency information. In *Proceedings of the 2003 Human Language Technology Conference and Conference of the North American Chapter of the Association for Computational Linguistics*.

Tetreault, J. and Chodorow, M. (2008a). Native judgments of non-native usage: Experiments in preposition error detection. In *Coling 2008: Proceedings of the workshop on Human Judgements in Computational Linguistics*, pages 24–32, Manchester, UK. Coling 2008 Organizing Committee.

Tetreault, J. and Chodorow, M. (2008b). The ups and downs of preposition error detection in ESL writing. In *Proceedings of COLING-08*, Manchester.

Tetreault, J., Foster, J., and Chodorow, M. (2010). Using parse features for preposition selection and error detection. In *Proceedings of the ACL 2010 Conference Short Papers*.

Yannakoudakis, H., Briscoe, T., and Medlock, B. (2011). A new dataset and method for automatically grading ESOL texts. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 180–189, Portland, OR.

Unsupervised and semi-supervised morphological analysis for Information Retrieval in the biomedical domain

Vincent CLAVEAU
IRISA-CNRS, Campus de Beaulieu
F35042 Rennes, France
Vincent.Claveau@irisa.fr

Abstract
In the biomedical field, the key to access information is the use of specialized terms. However, in most of Indo-European languages, these terms are complex morphological structures. The aim of the presented work is to identify the various meaningful components of these terms and use this analysis to improve biomedical Information Retrieval. We present an approach combining an automatic alignment using a pivot language, and an analogical learning that allows an accurate morphological analysis of terms. These morphological analysis are used to improve the indexing of medical documents. The experiments reported in this paper show the validity of this approach with a 10% improvement in MAP over a standard IR system.

Keywords: Morphology, biomedical terminology, alignment, analogical learning, morpho-semantic indexing, biomedical information retrieval.

Proceedings of COLING 2012: Technical Papers, pages 629–646,
COLING 2012, Mumbai, December 2012.

1 Introduction

In the biomedical domain, terminologies are the keystone of many applications. They are used for structuring the knowledge as well as retrieving and formalizing information contained in documents. For instance, the well-known MeSH®(Medical Subject Headings) `www.nlm.nih.gov/mesh` terminology is developed to index the very popular PubMED database (`www.pubmed.gov`). In most Indo-European languages, biomedical terms also have interesting inner characteristics in that they tend to be complex morphological constructions. Indeed, they are often resulting from the composition of several Greek or Latin roots, prefixes, and suffixes. This morphological complexity is an important point to take into account for basic operations like handling, understanding, translating or building semantic relationships between these terms, and furthermore for higher level applications like machine translation or, as we demonstrate in this paper, Information Retrieval (IR).

In this paper, we investigate the development of morphological resources and show how a biomedical Information Retrieval task can benefit from such resources. More precisely, we present several techniques aiming at breaking up a term into its morphological components, namely morphs[1], while labeling these morphs with some semantic information. To the contrary of existing studies (Deléger et al., 2008; Markó et al., 2005a, for example) which are chiefly based on human expertise, the techniques proposed here rely on unsupervised or semi-supervised approaches.

The original idea at the heart of our approach is to use the multilingualism of existing terminological databases. We exploit Japanese as a pivot language, or more precisely terms written in Kanjis, to help decompose the terms of other languages into morphs and associate them with the corresponding Kanjis, in a fully automatic way. Thus, Kanjis play the role of a semantic representation for morphs. The main advantage of Kanjis in this respect is that Japanese terms can be seen as a concatenation of elementary independent words that may even be found in general language dictionaries. For example, the term photochemotherapy can be translated in Japanese by 光化学法; splitting and aligning these two terms gives: photo ↔ 光 ('light'), chemo ↔ 化学 ('chemistry', 'medicine'), therapy ↔ 法 ('therapy'). As it is shown here, each morph is associated with Kanjis that may be used as descriptors more convenient to index a document than the term itself. In particular, we demonstrate here how such correspondences between morphs and Kanjis can be exploited, in different ways, to improve the results of an IR system.

The morphological analysis, and the document indexing that it allows, thus chiefly relies on an alignment step between morphs and Kanjis. This alignment is performed with an original technique, suited to the biomedical domain and based on a *Forward-Backward* algorithm and on analogy learning. In this paper, different versions of this alignment approach are proposed, either fully unsupervised, or semi-supervised.

The paper is structured as follows. After a review of the related studies in Section 2, we present the unsupervised alignment technique, its semi-supervised variants and their results respectively in Sections 3, 4 and 5. Then, in Section 6, the use of the obtained morphological decompositions in an IR framework is explained. Evaluations, conducted on a biomedical IR test collection, are detailed in Section 7.

[1]Following Mel'čuk (2006), we distinguish between morphs, elementary linguistic signs (segments), and morphemes, equivalence classes with identical signified and close signifiers.

2 Related work

Many studies have used morphology for terminological analysis. This is more particularly the case in the biomedical domain where terminologies are central to many applications and where terms are constructed by operations like neo-classical composition (e.g. *chemotherapy*, built from the Greek pseudo-word *chemo*, and *therapy*), which are very regular, and very productive. Unfortunately, no comprehensive database of morphs with semantic information is available, and splitting a term into morphs is still an issue. One can distinguish two views of the use of morphology as a tool for term (or word) analysis. In the lexematic view, relations between terms rely on the word form, but without the need to split them into morphs (Grabar and Zweigenbaum, 2002; Claveau and L'Homme, 2005; Hathout, 2009). Beside this implicit use of morphology, the morphemic view chiefly relies on splitting the term into morphs as a first step. Many studies have been made in this framework. They either rely on partially manual approaches in which an expert gives morphs and combination rules (Deléger et al., 2008; Markó et al., 2005a) or heuristics (Baud et al., 1999), or on more automatic approaches. The latter usually try to find recurrent letter patterns in word lists as morph-candidates (Kurimo et al., 2010). But such techniques cannot associate a semantic meaning with these morphs. To our knowledge, our approach is the first to make the most of a pivot language to perform an automatic morphological analysis, as we propose in this study. It can be explained by three peculiarities of the biomedical domain: the morphology of its terms is known to be very regular, with few exceptions, the morphological composition (producing compounds) is very fertile, and there exists many multilingual terminologies.

From a more technical point of view, the use of a bilingual terminology also evokes studies in transliteration, particularly Katakana or Arabic (Tsuji et al., 2002; Knight and Graehl, 1998, for example), or in translation. In this framework, Morin and Daille (2010) propose to map complex terms written in Kanjis with French ones, by using morphological rules. Yet, here again, these rules are to be given by an expert, and this study only concerns a special case of derivation. Moreover such an approach cannot handle neo-classical compounds. In other studies, translation methods for biomedical terms which considers terms as simple sequences of letters have been proposed (Claveau, 2009, inter alia). Such approaches share some similarities with the one presented here: they require aligning the words at the letter level. In most cases, this is performed with 1-1 alignment algorithms (one character, possibly empty, of the source language word is aligned with one another character of the target language word), but in recent work about phonetization (Jiampojamarn et al., 2007), authors have shown that the interest of *many-to-many* alignment.

Concerning the use of morphological processing in Information Retrieval, the literature is more important (Moreau and Sébillot, 2005, for a panorama). Although the results depend on numerous factors (language, morphological tool, size of collection, domain...), there is a broad consensus about the benefit of simple processes like *stemming* (or, rarer in IR, lemmatization): such tools are available in many languages, conceptually simple, and they usually improve the results of IR systems. It is noteworthy that the only morphological phenomena addressed by these tools are inflection and derivation. As they mostly perform simple operations on the prefix and suffix of the words, morphological composition remains out of their scope. Yet, many authors have noted the importance of clever tokenisation based on morpheme for the biomedical indexing (Jiang and Zhai, 2007; Trieschnigg et al., 2007), but without proposing effective solutions. Recently, advanced morphological tools developed in the framework of MorphoChallenge have been applied to IR problems (Kurimo

et al., 2009). Here again, the authors have observed an improvement for some languages, such as Finnish which is highly compositional, but results on English were significantly lower than when using a simple stemmer. In that respect, the good results presented in the next sections confirm the interest of our approach.

3 Analogy for Alignment

As it was previously explained, our morphological decomposition technique relies on the alignment of terms with their translation in a pivot language (Japanese, Kanjis). Thus, this approach makes a strong parallelism assumption: the term in Kanjis must be built in the same way than the one in the studied language. This hypothesis may appear as unrealistic, but the results presented hereafter show that it is reasonable. It is noteworthy that the choice of Kanjis as pivot is not fortuitous. Kanjis do not have morphology, their form is therefore invariable whatever their position in the term. One only needs to test a few combinations when trying to segment a term made of Kanjis, compared with a term written with the latin alphabet. Kanjis are independent of the Greek and Latin roots used in most European languages. This prevents learning irrelevant regularities based on common etymology. Last, a segment of a Kanji term is most of the time a valid Kanji term by itself (to the contrary of morphs). It is thus possible to use dictionaries to access their meaning. These different reasons make Kanji-based Japanese a very good pivot when compared to other alternatives.

Our alignment technique is mainly based on an *Expectation-Maximization* (EM) algorithm that we briefly present in the next sub-section (Jiampojamarn et al., 2007, for more details and examples of its use). The second sub-section explains the modification made to this standard algorithm so that it can naturally and automatically handle morphological variation, which is a phenomenon inherent to our morph splitting problem.

3.1 EM Alignment

The alignment algorithm at the heart of our approach is standard: it is a *Baum-Welch* algorithm, extended to map symbol sub-sequences and not only 1-1 alignments. In our case, it takes as input French terms with their Kanji translations, taken from a multilingual terminology for instance. The maximum length of the sub-sequences of letters and Kanjis considered for alignment are parametrized by $maxX$ and $maxY$.

For each term pair (x^T, y^V) to be aligned (T and V being the lengths of the terms in letters or Kanjis), the EM algorithm (see Algorithm 1) proceeds as follows. It first computes the partial counts of every possible mapping between sub-sequences of Kanjis and letters (*Expectation* step). These counts are stored in table γ, and are then used to estimate the alignment probabilities in table δ (*Maximization* step).

The *Expectation* step relies on a *forward-backward* approach (Algorithm 4): it computes the *forward* probabilities α and *backward* probabilities β. For each position t, v in the terms, $\alpha_{t,v}$ is the sum of the probabilities of all the possible alignments of (x_1^t, y_1^v), that is, from the beginning of the terms to the current position, according to the current alignment probabilities in δ (cf. Algorithm 2). $\beta_{t,v}$ is computed in a similar way by considering (x_t^T, y_v^V). These probabilities are then used to re-estimate the counts in γ. In this version of the EM algorithm, the *Maximization* (Algorithm 3) simply consists in computing the δ alignment probabilities by normalizing the counts in γ.

Algorithm 1 *EM Algorithm*

Input: list of pairs (x^T, y^V), $maxX$, $maxY$
while changes in δ **do**
 initialization of γ to 0
 for all pair (x^T, y^V) **do**
  γ = Expectation(x^T, y^V, $maxX$, $maxY$, γ)
 δ = Maximization(γ)
return δ

Algorithm 2 *Forward-many2many*

Input: (x^T, y^V), $maxX$, $maxY$
$\alpha_{0,0} := 1$
for $t = 0...T$ **do**
 for $v = 0...V$ **do**
  if $(t > 0 \vee v > 0)$ **then**
   $\alpha_{t,v} = 0$
  if $(v > 0 \wedge t > 0)$ **then**
   for $i = 1...maxX$ s.t. $t - i \geq 0$ **do**
    for $j = 1...maxY$ s.t. $v - j \geq 0$ **do**
     $\alpha_{t,v} \mathrel{+}= \delta(x^t_{t-i+1}, y^v_{v-j+1})\alpha_{t-i,v-j}$
return α

Algorithm 3 *Maximization*

Input: γ
for all sub-sequence a s.t. $\gamma(a, \cdot) > 0$ **do**
 for all sub-sequence b s.t. $\gamma(a, b) > 0$ **do**
  $\delta(a, b) = \frac{\gamma(a,b)}{\Sigma_x \gamma(a,x)}$
return δ

Algorithm 4 *Expectation*

Input: (x^T, y^V), $maxX$, $maxY$, γ
α := Forward-many2many(x^T, y^V, $maxX$, $maxY$)
β := Backward-many2many(x^T, y^V, $maxX$, $maxY$)
if $\alpha_{T,V} > 0$ **then**
 for $t = 1...T$ **do**
  for $v = 1...V$ **do**
   for $i = 1...maxX$ s.t. $t - i \geq 0$ **do**
    for $j = 1...maxY$ s.t. $v - j \geq 0$ **do**
     $\gamma(x^t_{t-i+1}, y^v_{v-j+1}) \mathrel{+}= \frac{\alpha_{t-i,v-j}\delta(x^t_{t-i+1}, y^v_{v-j+1})\beta_{t,v}}{\alpha^{T,V}}$
return γ

The EM process is repeated until the probabilities δ are stable. When the convergence is reached, the alignment simply consists in finding the mapping that maximizes $\alpha(T, V)$. In addition to this resulting alignment, we also store the final alignment probabilities δ, which are used to split unseen terms (cf. Section 6.2).

This technique is not very different from the one used in statistical translation. Yet, some particularities are worth noting: this approach allows us to handle *fertility*, that is the capacity to align from or to empty substrings (for lack of space, it does not appear in the above simplified version); conversely, *distortion*, that is reordering of morphs, cannot be handled easily without major changes in this algorithm.

3.2 Automatic morphological normalisation

The maximization step simply compute the translation probabilities of a Kanji sequence into a letter sequence. For example, for the Kanji 菌 ('*bacteria*'), there may exist one entry in δ associating it with bactérie, one with bactério (as in bactério/lyse) and another one with bactéri (in myco/bactéri/ose), each with a certain probability. This dispersion of probabilities, which is of course harmful for the algorithm, is caused by morphemic variation: bactério, bactérie, and bactéri are 3 morphs of the same morpheme, and we would like their probabilities to reinforce each other. The adaptation we propose aims at making the maximization phase able to automatically group the different morphs belonging to a same morpheme. To achieve this goal, we use a simple but well suited technique relying on formal analogical calculus.

3.2.1 Analogy

An analogy is a relation between 4 elements that we note: $a : b :: c : d$ which can be read *a is for b what c is for d* (Lepage, 2000, for more details about analogies). Analogies have been

used in many NLP studies, especially for translation of sentences (Lepage, 2000) or terms (Langlais and Patry, 2007; Langlais et al., 2008). Analogies are also a key component in the previously mentioned work on terminology structuring (Claveau and L'Homme, 2005). We rely on this latter work to formalize our normalization problem. In our framework, one possible analogy may be: dermato : dermo :: hémato : hémo. Knowing that dermato and dermo belong to a same morpheme, one can infer that this is the case for hémato and hémo. Such an analogy, build on the graphemic representation of words, is said a formal analogy.

After Stroppa and Yvon (2005), formal analogies can be defined in terms of factorizations. Let us note $\overleftarrow{\oplus}$ the (non-commutative) concatenation operator at the right ($abc\overleftarrow{\oplus}d = abcd$), and $\overleftarrow{\ominus}$ its associated string subtraction operator ($abc\overleftarrow{\oplus}d\overleftarrow{\ominus}d = abc\overleftarrow{\ominus}c\overleftarrow{\oplus}c = abc$), and similarly for $\overrightarrow{\oplus}$ and $\overrightarrow{\ominus}$ operating at the left of the first argument. Let a be a string (a term in our case) over an alphabet Σ, a factorization of a, noted f_a, is a sequence of n factors $f_a = (f_a^1, ..., f_a^n)$, such that $a = f_a^1 \overleftarrow{\oplus} f_a^2 \overleftarrow{\oplus} ... \overleftarrow{\oplus} f_a^n$. A formal analogy can be defined by as:

Definition 1 $\forall (a,b,c,d) \in \Sigma, [a : b :: c : d]$ *iff there exist factorizations* $(f_a, f_b, f_c, f_d) \in (\Sigma^{*n})^4$ *of* (a,b,c,d) *such that,* $\forall i \in [1,n], (f_b^i, f_c^i) \in \left\{(f_a^i, f_d^i), (f_d^i, f_a^i)\right\}$ *. The smallest n for which this definition holds is called the degree of the analogy.*

For most European languages, as French and English, morphology is mostly concerned with prefixation and suffixation. Thus, we are looking for formal analogies of degree at most 3 (ie, 3 factors: prefix $\oplus$ base $\oplus$ suffix). In our approach, such analogies are searched by trying to build a rule rewriting the prefixes and the suffixes to move from dermato to dermo and to check that this rule also applies to hémato-hémo. The base is considered as the longest common sub-string (lcss) between the 2 words. In the previous example, the rewriting rule r would be:
$r = \text{lcss}(\text{morph}_1,\text{morph}_2) \overleftarrow{\ominus} \text{ato} \overleftarrow{\oplus} \text{o}$.
This rule makes it possible to rewrite dermato into dermo and hémato into hémo; thus, hémato,hémo is in analogy with dermato,dermo.

3.2.2 Using analogy for normalization

The main problem is that we do not have examples of morphs that are known a priori to be related (like dermato and dermo in the previous example). Thus, we use a simple bootstrapping technique: if two morphs are stored in γ as possible translations of the same Kanji sequence, and if these two morphs share a sub-string longer than a certain threshold, then we assume that they both belong to the same morpheme. From these bootstrap pairs, we build the prefixation and suffixation rewriting rules allowing us to detect analogies, and thus to group pairs of morphs (which can be very short, unlike the bootstrapping pairs). The more a rule is found, the more certain it will be. Therefore, we keep all the analogical rules generated at each iteration along with their number of occurrence, and we only apply the most frequently found ones. The whole process is thus completely automatic.

This new *Maximization* step is summarized in Algorithm 5. It ensures that all the morphs supposed to belong to the same morpheme have equal and reinforced alignment probabilities.

4 Semi-supervision and bootstrapping

The approach described above can be considered as unsupervised since no example of alignment or decomposition is provided. Yet, in some cases, expert knowledge is available

Algorithm 5 *Maximization* with analogical normalization

Input: γ
for all sub-sequence a s.t. $\gamma(a, \cdot) > 0$ **do**
 for all m_1, m_2 s.t. $\gamma(a, m_1) > 0 \wedge \gamma(a, m_2) > 0 \wedge$ lcss$(m_1, m_2) >$ threshold **do**
  build the prefixation and suffixation rule r for m_1, m_2
  increment the score of r
 for all sub-sequence b s.t. $\gamma(a, b) > 0$ **do**
  build the set $\mathscr{M}$ of all morphs associated to b with the help of the n most frequent analogical rules from the previous iteration
  $\delta(a, b) = \frac{\sum_{c \in \mathscr{M}} \gamma(a, c)}{\sum_{x} \gamma(a, x)}$
return δ

and can be used to add some supervision to this morpho-semantic alignment task. It is important to note that this human intervention can be more or less costly and requires more or less expertise. While manually building a full morphological resource from scratch is a tedious task, providing light information during the alignment process is more accessible. In the following sub-sections, we propose different strategies to improve the unsupervised alignment process, implementing different trade-off between human cost and performance.

4.1 Active alignment

In analogy with active learning (Settles, 2009), the first semi-supervised strategy that we propose is active alignment. Its principle is the following: a human expert, the oracle, adds information about the pair during the expectation step. Different information can be used: decomposition of the Kanji term, of the English term, partial or full alignment. The interest of these pieces of information is twofold. First, it helps reduce the complexity, by avoiding to consider certain alignments in the pair. Secondly, it possibly improves the final results by reinforcing the probability for this pair on a few realistic alignments.

From an algorithmic point-of-view, the implementation is straightforward. The information provided by the oracle is interpreted as a set of constraints on the possible decomposition and/or alignments used to compute $\alpha_{t,v}$, $\beta_{t,v}$, and $\gamma(x_{t-i+1}^{t}, y_{v-j+1}^{v})$ in the Forward, Backward, and Expectation steps respectively.

As for active learning, one can think of different strategies to choose the pairs to be presented to the oracle (Settles, 2009). The goal is of course to find the strategy that best *helps* the alignment algorithm and thus results in a faster convergence and/or better performance. In this paper, two strategies are experimented and both will ask the oracle for providing full alignment of a pair. The first one is a random strategy and serves as baseline: at each iteration, randomly selected pairs are proposed to the oracle. The second strategy is a difficulty-driven one: at each iteration, pairs with many equi-probable alignments are proposed to the oracle. In practice, the equi-probability is measured with the probability gathered at the previous iteration.

4.2 Bootstrapping the oracle

The previous active alignment approach can also be used, to some extent, without human intervention. Indeed, when processing multiple languages, it is sometimes possible to make the most of the existing probabilities from a language L1 to help estimate the alignment

probabilities for the new language L2. Several ways to use these alignment probabilities of other languages can be imagined. In the experiments presented below, we used a simple approach. We adopt the difficulty-driven presented above, but instead of presenting the pairs (of L2) to a human oracle, they are processed as follows:
– if the Kanji term is known in the L1 alignment pairs, the closest term of L1 (edit distance) among the known translations is chosen; this L1-term and the Kanji term are then aligned and the alignment is propagated to the L2 pair. This step is done by representing alignment as characters in the L1 term and by adding the marks in L2 so that it minimizes the edit distance.
– if the Kanji is not known, the most probable alignment, based on the previous iteration probabilities is proposed.

5 Experiments

5.1 Evaluation Data

The data used in the experiments presented below come from the UMLS MetaThesaurus (Tuttle et al., 1990). The MetaThesaurus groups several terminologies for several languages and associates to each term a concept identifier (CUI). The CUI are language independent and thus make it easy to build lists of terms in the spotted language with their Japanese equivalents. In this paper, we present experiments for English and French. In both cases, we only considered Japanese terms composed of Kanjis, and only simple (one-word) French or English terms. About 14,000 English-Kanjis pairs and 8,000 French-Kanjis ones are formed this way. An ending mark (';') is added to each French or English term.

We randomly selected 1,600 pairs for French and 500 for English in order to evaluate the performance of our alignment technique. These pairs have been aligned manually to serve as gold standard.

5.2 Alignment results

In the different experiments presented below, the performance is evaluated in terms of precision: an alignment is counted as correct only if all the components of the pair are correctly aligned (thus, it is equivalent to the sentence error rate in standard machine translation).

For each pair, the EM algorithm indicates the probability of the proposed alignment. Therefore, it is possible to only consider alignments having a probability greater than a given threshold. By varying this threshold, we can compute a precision according to the number of terms aligned. Figures 1 and 2 respectively present the results obtained on the French and English test pairs. We indicate the curves produced by the EM algorithm with and without our morphemic normalization. For comparison purpose, we also report the results of giza++ (Och and Ney, 2003), a reference tool in machine translation. The different IBM models and sets of parameters available in giza++ were tested; the results reported are the best ones (obtained with IBM model 4 without distortion).

As expected, the interest of the morphemic normalization appears clearly in these two experiments; in the worst case (that is, when all the terms are kept for alignment), it yields a 70% precision for French and 80% for English. Indeed, the normalization brings a 10% improvement whatever the number of aligned pairs. Normalization also has an interest

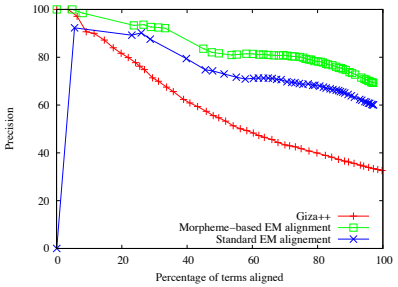


Figure 1: Precision of French-Kanji alignment according to the number of test pairs aligned

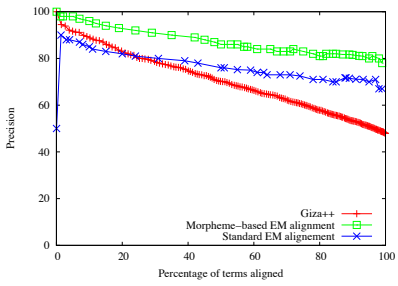


Figure 2: Precision of English-Kanji alignment according to the number of test pairs aligned

in terms of complexity since it reduces the needed iteration number by improving the convergence of the EM estimation.

A manual examination of the results shows that most of the errors are caused by the falsification of our hypothesis: some French-Japanese pairs cannot be decomposed in a similar way. For example, the French term anxiolytiques (anxiolytics) is translated by a sequence of Kanjis meaning literally 'drugs for depression'. Among these errors, some pairs imply terms that are not neo-classical compounds in French, Japanese or both (eg. méninges (meninges) is translated by 膜 'brain membrane'). Other errors are caused by a lack of training data: some morphs or sequences only appear once, or only combined with another morph, which mislead the segmentation.

5.3 Semi-supervised approaches

To compare the two active alignment strategies and the bootstrapped one presented in Section 4, we are interested in the alignment performance and the convergence speed. Thus, Figure 4 presents, for these three semi-supervised approaches together with the original unsupervised version, the precision on the French dataset after different number of iterations of the EM loop. For comparison purposes, we also report the results of the original unsupervised version. For a fair comparison between the unsupervised versions requiring the oracle, the same amount of pairs (20) is presented to the oracle at each iteration. The bootstrapped version is based on the alignment probabilities gathered from the English-Kanji alignment task. Of course, to prevent any bias, none of the pairs processed by the oracle are used as test pairs.

As expected, the three active alignment strategies converge faster than the original one, but also yield better overall results. Adding information at each iteration clearly helps the alignment to produce more relevant association between Kanjis and morphs. The difficulty-driven strategy obtains good results. In particular, it outperforms the random strategy after a few iterations. Before that, when iteration < 5, the probabilities collected are not reliable enough to propose interesting pairs to the oracle, and the oracle even seems to decrease the performance of the alignment. The combination of this strategy with the

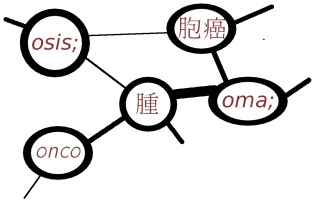


Figure 3: Morpheme-Kanji graph

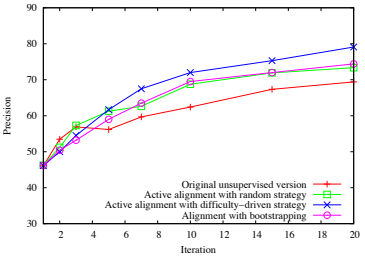


Figure 4: Precision of semi-supervised and unsupervised alignments according to the number of iterations

bootstrapping also performs better than the original version, but not as well as the real oracle one. This result can be explained by the fact that the difficulty-driven strategy asks for pairs that are difficult to align, not only for the studied language (here, French), but also for the one serving as bootstrap (English). Nonetheless, it still remains a good option to improve the results without any human supervision.

6 Morpho-semantic analysis for Information Retrieval

As it was previously said, biomedical Information Retrieval (IR) has some important characteristics due to the use of specialized terms. In that respect, taking into account rich morphological information has already been proved useful, but only with hand-crafted resources (Markó et al., 2005b). Beside the intrinsic evaluation of our approach presented in the previous section, we evaluate its use in a large scale IR experiment in English.

6.1 Morpho-semantic graphs

Once all the terms are aligned, one can study the recurrent correspondences between English morphs and Kanjis. The more a morph is aligned with a sequence of Kanjis, the more they are 'semantically' related. All these links can be represented as a graph: the vertices represent Kanjis and morphemes (i.e a set of morphs grouped during the analogical step of the alignment), and the edges are weighted according to the number of times that a particular morpheme is aligned with a Kanji sequence among the 14,000 training pairs from the umls. Figure 3 shows a toy example of such a graph. The size of the edge lines is proportional to the associated weight.

This representation allows us to shed light on different types of semantic relations between the morphemes. It is done by exploring the neighborhood of each morpheme: each vertex receives an amount of energy which is propagated to the connected vertices proportionally to the edge's weight. For instance, Figure 5 presents the closest morphemes reached, in the form of tag clouds, for the French morpheme ome (oma in English, a suffix for cancer-related terms). The size and color represent the energy that reach the neighboring morpheme vertices. The reached vertices are expected to be conceptually related and to exhibit synonym or quasi-synonym morphemes of the suffix ome. It is interesting to see that other

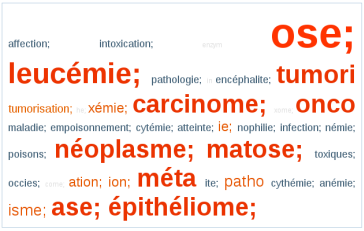


Figure 5: Cloud of 1st order affinities for French morpheme ome

Figure 6: Cloud of 2nd order affinities for French morpheme gastro

related suffixes are found, but also prefixes like onco.

The alignment and the segmentation produced by our algorithm also make it possible to study the co-occurrences of morphemes in English (or French) terms. One can study first-order affinities (which morphemes are frequently associated with other morphemes) and, more interesting, second order affinities (morphemes sharing the same co-occurring morphemes). The second-order affinity allows us to group morpheme according to their paradigm. For instance, the tag cloud in Figure 6 illustrates the morphemes associated with gastro (morpheme for stomach) according to this second order affinity. Most of the morphemes identify organs, and the closest ones are for biologically close organs.

This information of different nature makes it possible to identify relationships between terms, or build synonyms, or explore the termbase using these morphological elements. Yet, to our knowledge, such specialized morpho-semantic resources do not exist. It makes a direct evaluation of these three different uses of the alignment results not possible. But in the remaining of this paper, we propose to evaluate them in an Information Retrieval framework.

6.2 Morphemic representation for Information Retrieval

In order to integrate the morphological information in an IR system, we adopt a simple indexing representation: the documents are considered as bags-of-morphemes and words. The morphemes are those obtained by decomposing biomedical terms, and for some experiments those associated to the former ones as second order affinities. The goal is of course to be able to associate a query containing stomachalgia with a document containing gastrodynia.

Thus, when indexing the collection, terms are decomposed. Two cases may occur: the term is either known as it appears in the alignment pairs, either not. In the first case, we simply use the decomposition produced by the alignment algorithm. In the second case, we make the most of the δ probabilities to generate the most probable translation. To do so, we use a simple approach: the translation probability in δ are used by a Viterbi algorithm to generate the most probable Kanji translation. We do not use language modeling. It is

important to note that this translation process produces at the same time the decomposition of the initial term by associating each morph with its Kanji translation. This translation process is thus equivalent to the desired morpho-semantic analysis for unknown terms (i.e. absent from the pair list used in the alignment step).

In these two cases, another alignment product is also used: the analogical rewriting rules collected at the last alignment iteration. They allow us to detect the morphs belonging to a same morpheme. Such information makes it possible to match a query containing hemo with a document containing haemo, hemato or even emia;.

A *baseline* system and four indexing systems using the morphological information are proposed. They all rely on a standard IR approach, namely a vector space model with an Okapi BM25 weighting scheme (Robertson et al., 1998, for details, see) and a tokenizer similar to Terrier's one (Ounis et al., 2006); the standard values of the BM25 parameters b, k_1, k_3 have been kept. The *baseline* system performs a standard indexing of the documents with a Porter stemming (Porter, 1980).
1 – The first morphologically-enhanced system is morpheme-based. It simply considers the morphemes produced by the decomposition of the term in the documents (and queries) as words to index (the original terms are no longer used for indexing, only its morphemes). The morpheme weights take the decomposition probability into account; it is defined as the product of this probability and the BM-25 weight.
2 – The second system is Kanji-based. Here again, the terms of the documents are decomposed, and the closest Kanjis are used as indexing words. These Kanjis are those identified in the neighborhood of the morphs produced when decomposing the terms (see Section 6.1).
3 – The third system adopts the same morpheme-based representation as the first system, but expands the queries with first-order affinities of their morphemes. The morphemes used as expansion are weighted according to their proximity in the graph and the weight of the morphemes that they expand.
4 – The last system is similar to the third one but uses the second-order affinities to expand the queries.

7 Biomedical IR experiments

7.1 Experimental setting

For the following experiments, we use the dataset built for the filtering track of the TREC-9 conference. This dataset is itself based on the document collection ohsumed, which is composed of about 350,000 medline abstracts. In addition to this, about 4,000 queries and the corresponding relevance judgments were developed for TREC-9. The queries are composed of several fields: the subject, which a MeSH term, and a definition of this term. Although the collection was initially built for evaluating filtering systems, here we use this dataset as a standard IR collection, and only consider the subject field as the query.

7.2 Results

Table 1 presents the results of the baseline IR system and the system based on the morphological analysis. The performance of the systems are evaluated using the standard IR evaluation measures: we compute precision on top 5, 10... 1,000 documents (P@x), mean average precision (MAP), interpolated average precision (IAP) and the R-precision

	baseline (BM-25 + stemming)	System 1 morpheme-based	System 2 Kanji-based
MAP	29.93	33.94 (+13.4 %)	32.76 (+9.5 %)
IAP	31.74	35.55 (+12 %)	34.49 (+8.6 %)
R-prec	35.28	39.64 (+12.3 %)	38.59 (+9.4 %)
P@5	69.87	73.45 (+5.1 %)	71.70 (+2.6 %)
P@10	67.99	71.31 (+4.9 %)	*69.65 (+2.4 %)*
P@50	52.98	56.90 (+7.4 %)	55.24 (+4.3 %)
P@100	40.86	44.56 (+9.1 %)	43.39 (+6.2 %)
P@500	15.11	17.21 (+13.9 %)	16.92 (+12 %)
P@1000	8.72	10.10 (+15.86 %)	9.95 (+14.2 %)

Table 1: Performance of the morpheme and Kanji based IR systems on the OHSUMED collection, with the TREC queries

(R-prec).In order to assess if the differences between the two systems are statistically significant, we run a Wilcoxon test ($p = 0.05$) (Hull, 1993); those differences with the *baseline* that are not judged statistically significant are italicized.

The morpheme-based system, only relying on decomposing term and grouping its morph into morpheme, yields very good results with a 13% MAP gain. As expected, decomposing the terms improves more specifically the performance at the end of the retrieved document list (P@100 and higher), since it makes it possible to retrieve relevant documents even though they do not contain the exact terms of the queries. The Kanji-based system yields very similar results. Although the Kanjis were expected to be a more generic representation, no additional gain is obtained. In practice, in some queries, the Kanjis are too generic to capture the specific meaning expected or bring no additional information compared with the original morphemes. Moreover, no selection is performed on the morpheme to be translated into Kanjis, and some Kanjis have properties (document frequencies) that highly differ from the source morpheme since they can be translation of different morphemes. A weighting scheme taking the initial document frequencies into account seems an important foreseen work.

Table 2 presents the results of the two last systems, based on query expansion. The two expansion-based systems have more contrasted results. On the one hand, expanding the queries with first-order affinities gives good results; although it yields a lower precision at the top of list than system 1, it obtains a slightly better recall. On the other hand, second-order affinities produce bad results compared with morphological decomposition alone. The affinities added to the query, most of the time, break the specificity of the information asked; it makes the system retrieve too much non-related documents.

8 Conclusion and future work

The original idea of making the most of another language like Japanese in order to help the morphological decomposition and analysis of compounds offers many new opportunities to automatically handle biomedical terms. The new alignment approach based on analogy that we propose takes the particularities of the data into account, and also offers different ways to balance quality, convergence speed and human intervention through the semi-supervised approaches proposed. The high quality results obtained allow us to address IR problems

	baseline (BM-25 + stemming)	System with 1st order affinities	System with 2nd order affinities
MAP	29.93	34.40 (+14.9 %)	28.74 (-3.9 %)
IAP	31.74	36.63 (+15.4 %)	30.80 (-2.9 %)
R-prec	35.28	39.92 (+13.2 %)	34.38 (-2.6 %)
P@5	69.87	71.76 (+2.7 %)	*68.65 (-1.7 %)*
P@10	67.99	70.46 (+3.6 %)	*66.20 (-2.6 %)*
P@50	52.98	56.30 (+6.7 %)	50.50 (-4.68 %)
P@100	40.86	44.69 (+9.4 %)	39.07 (-4.38 %)
P@500	15.11	17.98 (+18.9 %)	*15.01 (-0.64 %)*
P@1000	8.72	10.56 (+21.1 %)	*8.77 +0.66 %)*

Table 2: Performance of the expansion based IR systems on the OHSUMED collection, with the TREC queries.

caused by the morphological complexity of the biomedical terminology that could not be addressed with usual IR tools like stemmers. In this respect, our concerns about the role of morphology to access information are similar to existing studies (Markó et al., 2005a; Deléger et al., 2008), but to our knowledge, we are the first to propose an automatic process, directly available for many languages. Of course, our approach chiefly relies on the availability of multilingual terminologies, but such databases like UMLS are now widely developed, on the contrary of usable morphological resources.

Many perspectives are foreseen from this work. First, from a technical point of view, we plan to consider more complex segmentation than the linear one we implemented. Indeed, the syntactic properties of the Kanjis (some of them expect an agent or object), could help to better structure the different morphemes. One could also exploit the semantic relations between Kanjis that can be easily found in general Japanese dictionaries.

Concerning the analysis aspects illustrated in the last section, many possibilities are also under consideration. As the links between morphs that we produce are not typed, the use of heuristics (such as string inclusion used by Grabar and Zweigenbaum (2002)) or techniques from distributional analysis could provide useful additional information to better characterize the relationships. Yet, the problem of evaluating this type of work arises, especially the ground truth construction, since such resources do not exist. The IR setting used in this paper could be used again, possibly with more biomedical collections such as the TREC Genomics ones (Hersch and Voorhees, 2009).

Finally, an adaptation of these principles for complex terms is under study. The main difficulty in this case is to manage the reordering of the words composing these terms, and thus manage the distortion in the alignment algorithm. This issue is important for IR since these multiword terms are known to occurs with many variants and thus prevent to match queries and documents with different variants of the same term (Nenadic et al., 2005).

Acknowledgments

This work was partly funded by OSEO, the French innovation agency, in the framework of the Quæro project.

References

Baud, R., Rassinoux, A.-M., Ruch, P., Lovis, C., and Scherrer, J.-R. (1999). The power and limits of a rule-based morpho-syntactic parser. In *Proceedings of the 1999 Annual Symposium of the American Medical Informatics Association. Transforming Health Care through Informatics. AMIA*, pages 22–26, Washington, DC, USA.

Claveau, V. (2009). Translation of biomedical terms by inferring rewriting rules. In Prince, V. and Roche, M., editors, *Information Retrieval in Biomedicine: Natural Language Processing for Knowledge Integration*. IGI - Global.

Claveau, V. and L'Homme, M.-C. (2005). Structuring terminology by analogy-based machine learning. In *Proc. of the 7th International Conference on Terminology and Knowledge Engineering, TKE'05*, Copenhaguen, Denmark.

Deléger, L., Namer, F., and Zweigenbaum, P. (2008). Morphosemantic parsing of medical compound words: Transferring a french analyzer to english. *International Journal of Medical Informatics*, 78(Supplement 1):48–55.

Grabar, N. and Zweigenbaum, P. (2002). Lexically-based terminology structuring: Some inherent limits. In *Proc. of International Workshop on Computational Terminology, COMPUTERM*, Taipei, Taiwan.

Hathout, N. (2009). Acquisition morphologique à partir d'un dictionnaire informatisé. In *Actes de la 16e Conférence Annuelle sur le Traitement Automatique des Langues Naturelles*, Senlis, France.

Hersch, W. and Voorhees, E. (2009). TREC genomics special issue overview. *Information Retrieval*, 12(1):1–15.

Hull, D. (1993). Using Statistical Testing in the Evaluation of Retrieval Experiments. In *Proceedings of the 16th Annual ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR'93*, Pittsburgh, États-Unis.

Jiampojamarn, S., Kondrak, G., , and Sherif, T. (2007). Applying many-to-many alignments and hidden markov models to letter-to-phoneme conversion. In *Proc. of the conference of the North American Chapter of the Association for Computational Linguistics*, Rochester, New York, USA.

Jiang, J. and Zhai, C. (2007). An empirical study of tokenization strategies for biomedical information retrieval. *Information Retrieval*, 10(4-5):341–363.

Knight, K. and Graehl, J. (1998). Machine transliteration. *Computational Linguistics*, 24(4):599–612.

Kurimo, M., Creutz, M., and Turunen, V. (2009). Morpho challenge evaluation by information retrieval experiments. In *Proceedings of the 9th Cross-language evaluation forum conference on Evaluating systems for multilingual and multimodal information access*, CLEF'08, pages 991–998, Berlin, Heidelberg. Springer-Verlag.

Kurimo, M., Virpioja, S., and Turunen, V. T. (2010). *(Eds), Proceedings of the MorphoChallenge 2010*. Espoo, Finlande.

Langlais, P. and Patry, A. (2007). Translating unknown words by analogical learning. In *Proc. of Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 877–886, Prague, Czech Republic.

Langlais, P., Yvon, F., and Zweigenbaum, P. (2008). Translating medical words by analogy. In *Proc. of the workshop on Intelligent Data Analysis in bioMedicine and Pharmacology (IDAMAP) 2008*, Washington, DC.

Lepage, Y. (2000). Languages of analogical strings. In *Proc. of the 18th conference on Computational linguistics, COLING'00*, Universität des Saarlandes, Saarbrücken, Germany.

Markó, K., Schulz, S., and Han, U. (2005a). Morphosaurus - design and evaluation of an interlingua-based, cross-language document retrieval engine for the medical domain. *Methods of Information in Medicine*, 44(4).

Markó, K., Schulz, S., Medelyan, O., and Hahn, U. (2005b). Bootstrapping dictionaries for cross-language information retrieval. In *Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '05)*, Salvador, Brésil.

Mel'čuk, I. (2006). *Aspects of the Theory of Morphology*. Trends in Linguistics. Studies and Monographs. Mouton de Gruyter, Berlin.

Moreau, F. and Sébillot, P. (2005). Contributions des techniques du traitement automatique des langues à la recherche d'information. Technical Report 1690, IRISA.

Morin, E. and Daille, B. (2010). Compositionality and lexical alignment of multi-word terms. *Language Resources and Evaluation (LRE)*, 44.

Nenadic, G., Spasic, I., , and Ananiadou, S. (2005). Mining biomedical abstracts: What's in a term? In *Proceedings of the IJCNLP 2004*, volume 3248 of *Lecture Notes in Artificial Intelligence*, pages 797–806. Springer-Verlag.

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

Ounis, I., Amati, G., Plachouras, V., He, B., Macdonald, C., and Lioma, C. (2006). Terrier: A High Performance and Scalable Information Retrieval Platform. In *Proceedings of ACM SIGIR'06 Workshop on Open Source Information Retrieval (OSIR 2006)*.

Porter, M. F. (1980). An Algorithm for Suffix Stripping. *Program*, 14:130–137.

Robertson, S. E., Walker, S., and Hancock-Beaulieu, M. (1998). Okapi at TREC-7: Automatic Ad Hoc, Filtering, VLC and Interactive. In *Proceedings of the 7th Text Retrieval Conference, TREC-7*, pages 199–210.

Settles, B. (2009). Active learning literature survey. Technical report, University of Wisconsin–Madison.

Stroppa, N. and Yvon, F. (2005). An analogical learner for morphological analysis. In *Proceeedings of the 9th CoNLL*, pages 120–127, Ann Arbor, MI, USA.

Trieschnigg, D., Kraaij, W., and de Jong, F. (2007). The influence of basic tokenization on biomedical document retrieval. In *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '07, pages 803–804, New York, NY, USA. ACM.

Tsuji, K., Daille, B., and Kageura, K. (2002). Extracting French-Japanese word pairs from bilingual corpora based on transliteration rules. In *Proc. of the 3rd International Conference on Language Resources and Evaluation, LREC'02*, Las Palmas de Gran Canaria, Spain.

Tuttle, M., Sherertz, D., Olson, N., Erlbaum, M., Sperzel, D., Fuller, L., and Neslon, S. (1990). Using meta-1 – the 1st version of the UMLS metathesaurus. In *Proc. of the 14th annual Symposium on Computer Applications in Medical Care (SCAMC)*, pages 131–135, Washington, USA.

A hybrid approach to finding phenotype candidates in genetic texts

Nigel Collier[1,2] *Mai – Vu Tran*[1,3] *Hoang – Quynh Le*[1,3]
Anika Oellrich[4] *Ai Kawazoe*[1] *Martin Hall – May*[5]
Dietrich Rebholz – Schuhmann[2,6]
(1) National Institute of Informatics, Tokyo, Japan
(2) European Bioinformatics Institute, Cambridge, UK
(3) University of Engineering and Technology - VNU, Hanoi, Vietnam
(4) Wellcome Trust Sanger Institute, Cambridge, UK
(5) University of Southampton, Southampton, UK
(6) The University of Zurich, Zurich, Switzerland
collier@nii.ac.jp,vutm@vnu.edu.vn,lhquynh@vnu.edu.vn
ao5@sanger.ac.uk,zoeai@nii.ac.jp
mhm@ecs.soton.ac.uk,rebholz@ebi.ac.uk

ABSTRACT
Named entity recognition (NER) has been extensively studied for the names of genes and gene products but there are few proposed solutions for phenotypes. Phenotype terms are expected to play a key role in inferring gene function in complex heritable diseases but are intrinsically difficult to analyse due to their complex semantics and scale. In contrast to previous approaches we evaluate state-of-the-art techniques involving the fusion of machine learning on a rich feature set with evidence from extant domain knowledge-sources. The techniques are validated on two gold standard collections including a novel annotated collection of 112 abstracts derived from a systematic search of the Online Mendelian Inheritance of Man database for auto-immune diseases. Encouragingly the hybrid model outperforms a HMM, a CRF and a pure knowledge-based method to achieve an F1 of 77.07. Disagreement analysis points to further improvements on this emerging NE task. The annotated corpus and guidelines are available on request.

KEYWORDS: conditional random fields, biomedicine, machine learning, genetic disorders, text mining.

Proceedings of COLING 2012: Technical Papers, pages 647–662,
COLING 2012, Mumbai, December 2012.

1 Introduction

Biomedical named entity recognition (NER) is a computational technique used to identify and classify strings of text (*mentions*) that designate important concepts in biomedicine. Over the last fourteen years there has been considerable interest in this problem with a variety of generic and entity-specific algorithms applied to extract the names of genes, gene products, cells, chemical compounds and diseases (Fukuda et al., 1998; Rindflesch et al., 1999; Collier et al., 2000; Kazama et al., 2002; Zhou et al., 2003; Settles, 2004; Kim et al., 2004; Leaman and Gonzalez, 2008). As the first stage in the integrated semantic linking of knowledge between literature and structured databases it is critically important to maximise the effectiveness of this step.

Despite significant progress in NER there is still no one size fits all solution. Barriers arise because of ambiguity in the text and coding schema. Ambiguity in the text comes in various forms according to the semantic type of the entity but can be caused by a lack of standard nomenclatures, extensive and growing nomenclatures for proteins/genes across multiple organisms or the widespread use of abbreviations and descriptive names. For example, (Krauthammer and Nenadic, 2004) illustrate uncontrolled naming in genes with *bridge of sevenless (boss)* (FlyBase ID FBgn0000206) and Hunter and Bretonnel Cohen (2006) discuss term class ambiguity (e.g. is *group* a chemical entity or an assemblage of organisms?). Such challenges have led to a variety of proposed solutions involving a wide range of resources. Among these, linguistically annotated corpora such as GENIA (Tateisi et al., 2000; Kim et al., 2003) have proven to be central to the NER solution. However due to the size of the vocabularies involved, annotated corpora by themselves do not provide a complete solution. Researchers have therefore also looked at the rich availability of formally structured biomedical knowledge (ontologies) such as the Unified Medical Language System (UMLS) (Bodenreider et al., 2002) and the Gene Ontology (Gene Ontology Consortium, 2000). Nevertheless corpora remain a key part of the solution as they provide the contextual evidence that link mentions to terms through the author's intentions. Creating such resources though is time consuming and expensive, especially when annotating new semantic types and relations.

In this paper we focus on the analysis and identification of a new class of entity: phenotypes. Two thoughts motivate this: (1) The database curation community has expressed a wish for full text entity indexing and the inclusion of phenotypes (Dowell et al., 2009; Hirschman et al., 2012), and (2) Biomedicine is rapidly moving towards full-scale integration of data, opening up the possibility to understand complex heritable diseases caused by genes. Association studies involving phenotypes are considered important to making progress (Lage et al., 2007; Wu et al., 2008). The ultimate goal of the work we present here is to allow relations mined from sentences such as the one we annotated below to feed into novel hypothesis generation procedures. From Ex 1. the reader can easily infer a relation between *IgG1 disorder* and three genes/gene products marked as GGP.

Ex 1. *Among* $[\text{patients}]_{ORG}$ *with* $[\text{systemic lupus erythematosus}]_{DIS}$ *(*$[\text{SLE}]_{DIS}$*), those with the* $[\text{IgG1 disorder}]_{PHEN}$ *have a higher prevalence of high titre* $[\text{rheumatoid factor}]_{GGP}$ *and* $[\text{antinuclear antibody}]_{GGP}$*, but a lower prevalence of* $[\text{anti-double-stranded DNA (anti-dsDNA) antibodies}]_{GGP}$ *above 30 U/ml.* (Source PMCID: PMC1003566).

Whilst other authors have tried similar approaches for other entity types, none have tried both machine learning and external resource lookup for a class as rich and semantically complex as phenotypes. The key contributions of this paper are: (1) To provide an operational semantics for identifying phenotype candidates in text, (2) To introduce a set of guidelines and an anno-

tated corpus based on a selection of 19 clinically significant auto-immune diseases from The Online Mendelian Inheritance of Man (OMIM) (Hamosh et al., 2005), one of the most widely used gene-disease databases, and (3) To mitigate linguistic variation whilst still meeting the conceptual expectations of biologists we propose a new named entity solution that uses statistical inference and external manually crafted resources. This method is tested on the new corpus and one extant corpus (Khordad et al., 2011) that has been used in previously reported experiments.

2 The challenge of phenotypes

Freimer and Sabatti (2003) describe phenotypes as referring to '*any morphologic, biochemical, physiological or behavioral characteristic of an organism. ...All phenotypic characteristics represent the expression of particular genotypes combined with the effects of specific environmental influences.*' Despite recent data integration efforts for phenotypes such as (Robinson and Mundlos, 2010), phenotypic descriptions still tend to be author/study specific and biological results may go undiscovered if the terms used lie outside an author's immediate research area (Bard and Rhee, 2004). Again, unlike genes or anatomic structures, phenotypes and their traits are complex concepts and do not constitute a homogeneous class of objects (i.e. a natural kind).

Traits such as 'eye colour', 'blood group', 'hemoglobin concentration' or 'facial grimacing' describe morphological structures, physiological processes and behaviours. When qualities or quantities of traits are used to describe a specific organism then we have phenotypic descriptions, e.g. 'blue eyes', 'blood group AB', 'not having between 13 and 18 gm/dl hemoglobin concentration'.

Traits and phenotypes can apply at all levels of anatomical granularity from chemical structures to cells and organs making it difficult to know where to draw a boundary. Phenotypes can include quantifications that are either specific (e.g. '18 gm/dl') or relative (e.g. 'normal' or 'increased'). Accordingly the first part of this paper deals with specifying exactly what we mean by the concepts of 'phenotype' with reference to current ontological research.

3 Methods

3.1 Schema

We employed two types of entity in our study: gene/gene product (GGP) and bodily feature (BF). GGP is proposed because (a) a subset of these entities are useful for applications that explore gene-phenotype relations, and (b) it allows us to compare our results against the many biomedical NER studies of the past, e.g. (Kim et al., 2004; Rebholz-Schuhmann et al., 2010). Because of space limitations we will not provide a rigidly formal definition or a taxonomic analysis (Beisswanger et al., 2008). Future work will explore the relationships between these and other entity types.

In line with BioTop (Beisswanger et al., 2008), GGP is relatively straightforward to define by the conjunction of (BioTop ID Nucleic Acid Structure) and (BioTop ID Peptide Structure).

Definition: A GGP (gene/gene product) entity is a mention of one of three major macromolecules DNA, RNA or protein. DNA and RNA are nucleic acid sequences containing the genetic instructions used in the development and function of an organism. Proteins are polypeptide sequences, or parts of polypeptide sequences, folded into structures that facilitate biological function.

Examples include: [cryoglobulins], [anticariolipin antibodies], [AFM044xg3], [chromosome 17q], [CC16 protein] .

Our definition of phenotype was taken from the formal analysis in Scheuermann *et al.* (Scheuermann et al., 2009) who define phenotype as '*A (combination of) bodily features(s) of an organism determined by the interaction of its genetic make-up and environment*'. It is important to recognise that this definition requires us to know the underlying cause. Since causality is often difficult to establish using narrow contextual evidence of the sort used in NER it seems reasonable that we focus here on identifying bodily features themselves, i.e. *phenotype candidates*, and then determine causality in another stage of processing.

Definition: A BF (bodily feature) entity is a mention of a bodily quality in an organism.

Examples include: [lack of kidney], [abnormal cell migration],[absent ankle reflexes] as well as more complex cases such as [no abnormality in his heart], [unfavorable serum lipid levels] and [suceptibility to ulcerative colitis].

Our definition of bodily features require two caveats (a) in contrast to Khordad et al. (2011) we did not apply a granular cut off at the level of cell, and (b) because of the diversity of bodily features across organisms we took a decision to focus our definition of this entity on mouse as a model organism and human as the most important species. Following the discussion of phenotypes as processes in physiology (Hoehndorf et al., 2012) we include some mentions of processes within the scope of our annotation schema.

Linguistic forms of entities require a number of policy decisions to be made about how to annotate mentions in text. For a class as complex as phenotypes this is a particular consideration. Although more complex approaches exist, for simplicity we make the common assumption here that named entities are 'continuous, non-nested and non-overlapping' (Alex et al., 2007). As a basic policy we do not allow embedding of entities within our corpus so annotators have to make a choice of entity class based on the longest matching span even though one entity may contain another entity of the same or a different type. We leave to future work consideration of other approaches, e.g. for handling discontinuous entity mentions. Within our guidelines we describe whether specific, generic, underspecified and negatively quantified mentions qualify. A summary of the rule set (available from the first author) is shown in Table 1. We follow (Magnini et al., 2006) in differentiating between specific, generic and underspecified mentions.

3.2 Annotated data sources

3.2.1 Phenominer

The Phenominer version 1 corpus contains 112 abstracts we selected from PubMed Central (PMC). 19 auto-immune diseases were selected from OMIM and from these records citations were then chosen. Diseases include Type 1 diabetes, Grave's disease, Crohn's disease, autoimmune thyroid disease, multiple sclerosis and inflammatory arthritis. In order to ground the article in discussion about both a disease and a phenotype, citations needed to contain the auto-immune disease term and at least one term from either OMIM's free form clinical synopsis field, the Human Phenotype Ontology (HPO) (Robinson and Mundlos, 2010) or the Mammalian Phenotype Ontology (MP) (Smith and Eppig, 2009).

Despite being small, the number of annotated abstracts is consistent with several previous specialised studies, e.g. (Suakkaphon et al., 2011; Collier et al., 2000). Annotation was carried

	BF	GGP
specific reference	Yes	Yes
generic reference	Yes	Yes
underspecified reference	No	No
modifiers	Yes[1,2]	No
conjunctions	Yes[3]	Yes[3]
processes	Yes[4]	No
negation	Yes[5]	No

Table 1: Referential semantics and scoping of mentions by entity type. Notes on annotation: [1] Quantitative modifiers are included, e.g. [having five fingers] as well as spatial modifiers, e.g. [abnormality in his left hand]. [2] Qualitative modifiers are included such as physical components: [black hair], underspecified ranges: [normal height], locational modifiers: [low set ears], and level modifiers: [quite small fingers].[3] Where there is elision of the head, e.g. [IA/H5 virus], then we annotate the whole expression. Otherwise we annotate each expression separately, e.g. [IA virus] and [H5 virus]. [4] We exclude finite verb forms, infinite verb forms with 'to', verbs in a progressive or perfect aspect, verb phrases, clauses or sentences and any phrase with a relative clause or complement clause. [5] If the negation appears in a noun phrase with an anatomical entity then we generally allow it, e.g. [absent ankle reflexes], [no left kidney].

out by the same highly experienced biomedical annotator who had annotated the GENIA corpus. The total number of tokens (sentences) in the corpus is 26,026 (1976) from which there were 1611 GGP entities and 472 BF entities.

3.2.2 KMR

As a basis of comparison we test our methods on the same corpus and tagging model as Khordad et al. (2011) who used a collection of 3784 tokens (120 sentences) with 110 annotated phenotype mentions. This is designated as the *KMR* corpus. In contrast to the Phenominer corpus sentences in KMR were taken from 4 PubMed papers from the year 2009 in the area of human genetics. Annotation was conducted with reference to the HPO so that a term was tagged as phenotype if it was in the HPO or if it was not in the HPO but its definition showed that it was caused by a genotype (Khordad, 2012). Finally we found that there was no cross over of sentences between the Phenominer and KMR corpora.

3.3 Models

The full system we developed (designated in the Results as *Hybrid*) employs machine learning and knowledge-based approaches, combined together with a rule-based Merge module. This is illustrated in Figure 1. Below we briefly describe its component modules and resources. As a baseline comparison we use Khordad's approach, designated in the Results as *Khordad* and in Figure 1 as the *Rule matching* module - see Khordad et al. (2011).

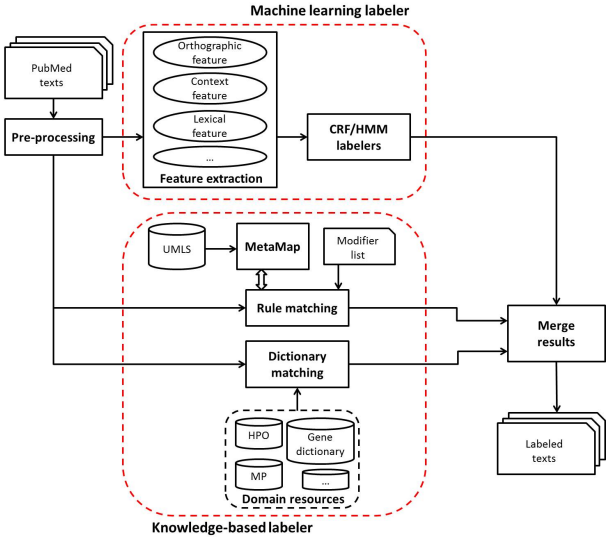


Figure 1: Phenotype tagging architecture

3.3.1 Pre-processing

The text collections are passed into a module which splits texts into sentences and tokens. This was done using the OpenNLP library with a Maximum Entropy model[1]. Abbreviation expansion is then done using BioText (Schwartz and Hearst, 2003) to make a list of local abbreviation occurring in each paper which we then replace with their full form. A similar approach is adopted by Khordad in their staged rules.

3.3.2 Machine learning labeler

Within the machine learning module we compare two widely used sequence labeling models: a second order Hidden Markov Models (HMM) (Rabiner and Juang, 1986; Bikel et al., 1997) with Viterbi decoding and a linear chain Conditional Random Fields (CRF) (Lafferty et al., 2001; McDonald and Pereira, 2005). Both are run as fully supervised models. Class labels for tokens follow the standard BIO system, i.e. each token receives the label O if it is not an NE, B plus the entity name when it starts an entity, and I plus the entity name when it is inside an entity. The main advantage of the CRF over the vanilla HMM is that it estimates the conditional probability distribution over labeled sequences. Both use the freely available Java-based MALLET implementation[2] with default parameters.

Previous research has found that utilizing various features for both the focus word (designated

[1]OpenNLP library: `http://opennlp.apache.org`
[2]Mallet: A machine learning for language toolkit: `http://mallet.cs.umass.edu/`

as w_i) and the surrounding words is crucial to obtain high performance. We take our feature set for BF and GGP labeling from the most typical and effective features used for biomedical NER (Kim et al., 2004). This is summarised in Table 2. Our experiments tested a variety of unigram, bigram and conjoined base features. These were taken from a ±2 window around the focus word for parts of speech, orthography and surface word forms. POS tagging was done using the OpenNLP library with Maximum Entropy model and Genia Corpus + WSJ Corpus (F-score 98.4%), there are 44 Penn Treebank POS tags and all of them are used. The HMM did not use conjoined features due to model memory limitations.

Feature	Description	Example
LX	Current word token	w_i
MM	MetaMap tag of the token[2]	cgab,fndg,neop
OR	Orthography of the token	initCap, isDate, allCap, isDigit
CT	Word token context	
	History context of the token	w_{i-2} , w_{i-1}
	Future context of the token	w_{i+1} , w_{i+2}
	Conjoined context	w_{i-2} . w_{i-1}
POS[1]	Part of speech tag of the token	RB, CD, NN, JJ, NNP

Table 2: Feature sets used in the machine learning labeler. [1]Part of speech tags are assigned by training the GENIA tagger (Tsuruoka et al., 2005). [2] The MetaMap semantic tags are chosen from the same group of 15 semantic types chosen by Khordad which are relevant for phenotypes.

In addition to the Phenominer and Khordad corpora outlined earlier we also make use of the JNLPBA04 corpus (Kim et al., 2004) for training the GGP labeler. The corpus contains 2000 Medline abstracts selected by a search using terms *human, blood cell, transcription factor* and then hand annotated for 5 NE classes including RNA, DNA and protein which we merge to form our GGP class. Table 3 summarises the features exploited by the two learner models.

Model	Target class	Phenominer corpus[1]	JNLPBA04 corpus	LX	MM	OR	CT	POS
HMM	BF	+	-	+	-	-	+	-
CRF	BF	+	-	+	+	+	+	+
HMM	GGP	+	+	+	-	-	+	-
CRF	GGP	+	+	+	+	+	+	+

Table 3: Resource combinations compared in our experiments. [1]This is applied within the 10-fold cross validation framework.

3.3.3 Knowledge-based labeler

The knowledge-based labeler is divided into *Rule matching* and *Dictionary matching* modules. Rule matching is an implementation of Khordad's approach using MetaMap, a subset of the UMLS, the HPO as well as 5 staged heuristics to identify phenotypes. For example, if a phrase has the form: "modifier (from the list of selected modifiers[3]) +[Anatomy] or [Physiology]" it is a phenotype name.

[3]The list of 85 high frequency modifiers from the HPO is available from the first author

Dictionary matching uses a longest string matching approach to recognise entities from the following resources: BF entities from the HPO (9500 terms describing human phenotypes) and the MP (9162 terms describing mouse phenotypes); GGP entities from the National Center for Biotechnology Information's gene list (9 million gene names). These sources were chosen because of their high standing within the biomedical community.

3.3.4 Merge results

Merge results assigns the final entity label to each token in the corpus by applying the following rules to each source module output. Processing proceeds sentence by sentence.

1. Following Jimeno et al. (2008) we combine the putative entity labels by collecting any entity-specific result that has been proposed by at least one method. This is intended to maximise recall. Thus, the O tag (non-entity label) has the least priority.
2. Based on our ontological analysis of BF and GGP it is often possible for a GGP to form a fully embedded part of a BF mention. For example, $[[\text{HLA-DQ}]_{GGP}\ \text{expression}]_{BF}$. We therefore apply a longest span rule and give priority to BF over GGP giving $[\text{HLA-DQ expression}]_{BF}$.
3. If there is a boundary conflict, we merge neighbouring entity mentions that share parts of their token sequence. For example, if we have $[\text{AB}]_{GGP}$ and $[\text{BC}]_{BF}$ then we merge them into one phrase [ABC] and label it with the highest priority tag, i.e. BF. Although this appears rare in GGP and BF we included this rule for expandability when we want to introduce further entity classes.

4 Evaluation

4.1 Metrics

We follow standard metrics of evaluation for the task using F1 as our primary method of comparison[4]. In these experiments matching is calculated using partial matching, i.e. a correct match is recorded when the span of text that is manually annotated in the gold standard corpus and the span of text output as an entity by the NER tagger partially overlap. For example a system annotation of *[median cleft lip]/palate* would be judged correct for a gold standard annotation of *median [cleft lip/palate]*. Various authors in the biomedical NER domain such as Kabiljo et al. (2009) have offered a reason for why this or other methods such as sloppy left boundary matching might be preferred to strict matching for genes and proteins. In summary it is thought that with partial matching, for the entity types examined so far, the core part of the entity was in most cases correctly found. In contrast, strict matching places too much faith in arbitrary choices in annotation guidelines.

4.2 Experiments on the KMR corpus

Our initial test run is conducted on the KMR corpus with micro-averaged F1 scores shown in Table 4. Since the corpus only contains phenotype tags no GGP results are shown.

We noted that curiously the results we observed for Khordad's method are slightly down by approximately 3 points of F1 on those given in their article. This appears mostly to affect

[4] F1 is the harmonic mean of recall (R) and precision (P) and is calculated as $F1 = 2PR/P + R$.

precision rather than recall and is possibly due to implementation differences such as changes to MetaMap/UMLS. Given the small amount of training data it is not surprising to see the pure machine learning based methods perform relatively poorly (F1:34.07,68.29) against the pure knowledge based approach (F1:83.29). Khordad's staged rule based system with dictionary lookup performs the best (F1:89.58) with the new Hybrid approach a reasonably close second (F1:85.27). It is encouraging though to see that the rule-based combination of the learner and KB outputs add value to the KB-only result.

		Model				
Class	**Metric**	**Khordad**	**HMM**	**CRF**	**KB**[1]	**Hybrid**[2]
BF	P	90.74	37.54	65.09	87.64	86.37
	R	88.44	31.18	71.83	79.36	84.19
	F	89.58	34.07	68.29	83.29	85.27

Table 4: (F)-scores, (R)ecall and (P)recision for each entity type on the KMR corpus using models with partial matching. [1]The KB method uses the Merge module to resolve conflicts. [2]Hybrid refers to the jointly applied system.

4.3 Experiments on the Phenominer corpus

For the Phenominer data set we chose to add the GENIA NER tagger trained on the JNLPBA04 corpus as a baseline for GGP. Note that we also combined this corpus data with Phenominer for training the CRF and HMM GGP recogniser. Khordad's method remains as our baseline for phenotypes. Micro-averaged F1 scores are shown in Table 5. With regard to GGP entities we observed that whilst the GENIA tagger performed robustly (F1:80.89), the Hybrid model appears to significantly outperform this (F1:85.48). The surprising result is that Khordad's method performs relatively poorly (F1:61.38) on BF. Again we try to dig down into the results in the Discussion to get an understanding behind the complex contributing factors.

			Model				
Class	**Metric**	**Khordad**	**GENIA**[3]	**HMM**[1,2]	**CRF**[1]	**KB**[4]	**Hybrid**[5]
BF	P	65.89	-	34.67	66.32	61.24	78.21
	R	57.44	-	38.11	64.17	60.91	75.96
	F	61.38	-	36.31	65.23	61.07	77.07
GGP	P	-	78.35	64.03	76.84	92.74	86.67
	R	-	83.61	65.80	80.07	61.31	84.32
	F	-	80.89	64.90	78.42	73.82	85.48
Total	Micro avg-F	-	-	56.46	75.19	71.62	85.04
	Macro avg-F	-	-	50.61	71.83	67.45	81.28

Table 5: (F)-scores, (R)ecall and (P)recision for each entity type on the Phenominer corpus using models with partial matching. [1]HMM and CRF are trained separately on each entity class and resolved in the Merge module. [2]Training included the JNLPBA04 corpus data for GGPs. [3]The GENIA method is the GENIA NER tagger trained on the GGP entities in the JNLPBA04 corpus. [4]The KB method uses the Merge module to resolve conflicts. [5]Hybrid refers to the jointly applied system.

5 Discussion

The results on the Phenominer corpus for Hybrid (F1:77.07) on BF are very encouraging and as we hoped demonstrate the strength of combining a mildly context sensitive ML approach with knowledge base lookup. Current NE methods based on a state-of-the-art learning approach such as CRF seem well suited to non-complex NE types such as GGP but maybe less effective for complex entities such as BF. Given the small size of the corpora we must be cautious in this conclusion. With regard to the KB approach for BF, our first impression was that the phenotype resources (HPO and MP) may to some extent lack coverage on the Phenominer corpus but we discuss below why this conclusion maybe too simplistic.

We start our analysis with the necessary observation that the Phenominer and KMR corpora do not offer a strict like-for-like comparison and are therefore most useful to highlight areas of difficulty. Importantly as we noted in Section 2, there is the issue of causality which is implicitly encoded into Khordad's schema and absent from ours. This means that our bodily features may not have a genetic or environmental cause. There is also the issue of granularity: our schema is more complex as it encodes bodily features from the genetic level upwards whereas Khordad's operates on the cellular level upwards. A statistical analysis points to further differences. We found that the average phenotype mention length in the KMR corpus was 1.72 tokens with the longest term being 5 tokens: [*hypoplasia of the corpus callosum*]. In contrast the average bodily feature mention in Phenominer is 2.89 tokens with the longest being [*susceptibility to psoriasis (PS) and psoriatic arthritis (PSA)*]. The longest GGP in Phenominer is 16 tokens: [*chromosomes 1 (D1S235), 4 (D4S1647), 12 (D12S373), 16 (D16S403), and 17 (D17S1301))*]. Both of these examples from Phenominer indicate structural term issues related to coordination and elipsis which are not easily handled by the simple longest term match approach that we have adopted.

Table 6 shows examples of where the Hybrid method disagreed with the KMR corpus. Whilst we have not conducted an in-depth analysis the examples seem reasonable and indicative of differences between the two coding schemas regarding causality of a bodily feature, algorithmic differences in how we prioritize UMLS semantic types related to *Disorder* and gaps in the knowledge resources.

No.	**Standard annotation**	**System annotation**	**Issue**[1]	**Cause of error**
1	eversion of the lateral eyelid	-	FN	Cannot be found in HPO or by rule matching
2	cervical rachischisis	-	FN	Hybrid system does not include default assignment for UMLS semantic types
3	absent nervi olphactorii	-	FN	
4	-	pregnancy	FP	Bodily feature does not differentiate between normal and abnormal
5	-	female	FP	
6	-	height	FP	

Table 6: Sources of error by the Hybrid system on the KMR corpus. [1] FN: False Negative; FP: False Positive.

Table 7 looks now at examples in the Phenominer corpus where the Hybrid approach disagreed

with Khordad's model. In the table the Hybrid model output agrees with the annotated corpus and the Issue column refers to the Khordad annotation. We see in particular that differences in the schema semantics account for many of the errors. The Phenominer schema for bodily features does not include disease mentions and simple anatomical entities but these may both be considered as phenotypes by the HPO. Clearly a notion of the compositional semantic relationships between types within terms is important to fully resolve the score differences.

Since Khordad's method relies to a greater extent than Hybrid on the HPO, we tested a number of terms from the Phenominer corpus by searching for them in the HPO. Using the exact match facility in OBO-Edit[5] we found several gaps. The following terms could not be found: complex terms such as [*perivascular distribution and granular deposits of immunoglobins*] as well as some gene specific terms such as [*IGG1 disorder*]. Surprisingly several seemingly common terms such as [*kidney impairment*] and [*abnormal thyroid function*] could also not be identified from a simple exact match. In the case of [*kidney impairment*] a suitable match might be found in *Abnormality of renal physiology* (HPO ID 0000082) by replacing the organ name with its anatomical adjective. Of 12 BF mentions in the Phenominer corpus that were not in the HPO our analysis revealed that 9 of them could be found by Hybrid. The ones that were not found tended to be very long and involved either coordination or a preposition phrase.

No.	**Hybrid annotation**[2]	**Khordad annotation**	**Issue**[1]	**Cause of error**
1	pathogenic process	-	FN	These entries do not belong to the UMLS's 15 target types, and are not in the HPO, and cannot be recognised by the pattern rules.
2	gene expression	-	FN	
3	RA susceptibility	- -	FN	
4	-	Inflammatory bowel disease	FP	Although this is present in HPO it is considered as a disease in our guidelines
5	-	enteropathy bowel disease	FP	Although this is present in HPO it is considered as an anatomical entity in our guidelines
6	-	asthma susceptibility gene	FP	Although this is present in HPO it is considered as GGP in our guidelines

Table 7: Sources of error by Khordad's system on the Phenominer corpus. [1] FN: False Negative; FP: False Positive. [2] We show here Hybrid system outputs that are correctly annotated.

Finally we show examples of disagreement for the Hybrid method on the Phenominer corpus in Table 8. As is common the biomedical literature we noticed a high proportion of coordination issues as well as ambiguity caused by generic terms.

[5]OBO-Edit: the OBO ontology editor: `http://oboedit.org/`

No.	Standard annotation[2]	Hybrid annotation	Issue[1]	Cause of error
1	FEV 1	-	FN	Because of orthographic similarity to genes this is tagged as GGP
2	[asthma]$_{BF}$ and [atopy phenotypes]$_{BF}$	[asthma and atopy phenotypes]$_{BF}$	FP	Coordination creates a boundary error
3	emotion	-	FN	This generic term is context sensitive
4	Diabetes Mellitus	[Diabetes Mellitus]$_{BF}$	FP	Entity class error
5	[citrullination]$_{BF}$ of the [endogenous antigen]$_{GGP}$	[citrullination of the endogenous antigen]$_{GGP}$	FP	Boundary error due preposition phrase

Table 8: Sources of error by the Hybrid system on the Phenominer corpus. [1] FN: False Negative; FP: False Positive. [2] We show here Hybrid system outputs that are correctly annotated.

6 Conclusions and future work

We have presented new results and analysis that add evidence to how phenotype candidates can be identified using named entity technology. The methods we have employed are aimed at making tractable the annotation of a critical semantics in the scientific literature. To do this we have matched surface forms to their attested forms in domain resources, balanced against contextual evidence from annotations in the scientific literature. The benchmark tests have demonstrated that the Hybrid method performs strongly on both the KMR corpus as well as the new Phenominer corpus. The evidence points towards complementarities between the existing phenotype resources and contextual evidence from annotated corpora.

Our methods have been formulated to be simple, effective and extensible with a focus on providing input to more knowledge intensive techniques downstream that can identify causality. Simplicity though may have sacrified both precision and recall in some cases, e.g. in the issue of coordination, in including generic and underspecified references and in adopting a longest matching approach to annotation.

There is considerable scope for further investigation. F1 might be increased using a machine learning framework such as integer linear programming (Koomen et al., 2005) to resolve hypotheses against multiple constraints much as we have tried to do manually in the Merge module. Coverage might be extended by including disjoint entities and a deeper analysis of embedded entity semantics such as that employed by Alex et al. (2007). In line with Hoehndorf et al. (2010) future solutions may need to focus on decomposing phenotypes in terms of their internal relations such as qualities [6].

[6] e.g. The Phenotypic Attribute and Trait Ontology `http://obofoundry.org/cgi-bin/detail.cgi?quality`

Acknowledgments

We gratefully acknowledge partial funding from NII and from an EC Marie Curie International Incoming Fellowship award (Project: Phenominer).

References

Alex, B., Grover, C., and Haddow, B. (2007). BioNLP 2007 Workshop at ACL2007, Prague, Czech Republic. In *Recognising Nested Named Entities in Biomedical Text*, pages 65–72.

Bard, J. B. L. and Rhee, S. Y. (2004). Ontologies in biology: design, applications and future challenges. *Nature Reviews Genetics*, 5(3):213–222.

Beisswanger, E., Schulz, S., Stenzhorn, H., and Hanh, U. (2008). BioTop: an upper domain ontology for the life sciences. *International Journal of Applied Ontology*, 3:205–212.

Bikel, D., Miller, S., Schwartz, R., and Wesichedel, R. (1997). Nymble: a high-performance learning name-finder. In Grishman, R., editor, *Proceedings of the Fifth Conference on Applied Natural Language Processing (ANLP'97), Washington Marriot Hotel, Washington D.C., USA.*, pages 194–201.

Bodenreider, O., Mitchell, J. A., and McCray, A. T. (2002). Evaluation of the UMLS as a terminology and knowledge resource. In *Proc. Americal Medical Informatics Association (AMIA) Annual Symposium, San Antonio, TX*, pages 61–65. AMIA.

Collier, N., Nobata, C., and Tsujii, J. (2000). Extracting the names of genes and gene products with a hidden Markov model. In *Proceedings of the 18th International Conference on Computational Linguistics (COLING'2000), Saarbrucken, Germany*, pages 201–207.

Dowell, K., McAndrews-Hill, M., Hill, D., Drabkin, D., and Blake, J. (2009). Integrating text mining into the MGI biocuration workflow. *Database*, bap019.

Freimer, N. and Sabatti, C. (2003). The human phenome project. *Nature Genetics*, 34(1):15–21.

Fukuda, K., Tsunoda, T., Tamura, A., and Takagi, T. (1998). Toward information extraction: identifying protein names from biological papers. In *Proceedings of the Pacific Symposium on Biocomputing'98 (PSB'98)*, pages 707–718.

Gene Ontology Consortium (2000). Gene ontology: tool for the unification of biology. *Nature Genetics*, 25:19–29.

Hamosh, A., Scott, A. F., Amberger, J. S., and Bocchini, C. A. (2005). Online mendelian inheritance of man (OMIM), a knowledgebase of human genes and genetic disorders. *Nucleic Acids Research*, 33(suppl 1):D514–D517.

Hirschman, L., Burns, G., Krallinger, M., Arighi, C., Bretonnel Cohen, K., Valencia, A., Wu, C., Chatr-Aryamontri, A., Dowell, K., Huala, E., Lourenco, A., Nash, R., Veuthey, A., Wiegers, T., and Winter, A. (2012). Text mining for the biocuration workflow. *Database*, 2012(bas020). doi:10.1093/database/base020.

Hoehndorf, R., Harris, M. A., Herre, H., Rustici, G., and Gkoutos, G. V. (2012). Semantic integration of physiology phenotypes with an application to the cellular phenotype ontology. *Bioinformatics*, 28(13):1783–1789.

Hoehndorf, R., Oellrich, A., and Rebholz-Schuhmann, R. (2010). Interoperability between phenotype and anatomy ontologies. *Bioinformatics*, 24(24):3112–3118.

Hunter, L. and Bretonnel Cohen, K. (2006). Biomedical language processing: Perspective what's beyond pubmed? *Molecular Cell*, 21(5):589–594.

Jimeno, A., Jimenez-Ruiz, E., Lee, V., Gaudan, S., Berlanga, R., and Rebholz-Schuhmann, D. (2008). Assessment of disease named entity recognition on a corpus of annotated sentences. *BMC Bioinformatics*, 9(Suppl 3):S3.

Kabiljo, R., Clegg, A., and Shepherd, A. (2009). A realistic assessment of methods for extracting gene/protein interactions from free text. *BMC Bioinformatics*, 10(1):233.

Kazama, J., Makino, T., Ohta, Y., and Tsujii, J. (2002). Tuning support vector machines for biomedical named entity recognition. In *Workshop on Natural Language Processing in the Biomedical Domain at the Association for Computational Linguistics (ACL) 2002*, pages 1–8.

Khordad, M. (2012). Personal communication by email, August 31st 2012.

Khordad, M., Mercer, R. E., and Rogan, P. (2011). Improving phenotype name recognition. In *Advances in Artificial Intelligence*, volume 6657/2011, pages 246–257. Lecture Notes in Computer Science.

Kim, J., Ohta, T., Tsuruoka, Y., Tateisi, Y., and Collier, N. (2004). Introduction to the bio-entity recognition task at JNLPBA. In Collier, N., Ruch, P., and Nazarenko, A., editors, *Proceedings of the International Joint Workshop on Natural Language Processing in Biomedicine and its Applications (JNLPBA), Geneva, Switzerland*, pages 70–75. held in conjunction with COLING'2004.

Kim, J. D., Ohta, T., Tateishi, Y., and Tsujii, J. (2003). GENIA corpus - a semantically annotated corpus for bio-textmining. *Bioinformatics*, 19(Suppl.1):180–182.

Koomen, P., Punyakanok, V., Roth, D., and Yih, W. (2005). Generalized inference with multiple semantic role labeling system. In *Ninth Conference on Computational Natural Language Learning (CoNLL '05), Michigan, USA*, pages 181–184.

Krauthammer, M. and Nenadic, G. (2004). Term identification in the biomedical literature. *Journal of Biomedical Informatics*, 37(6):512 – 526.

Lafferty, J., McCallum, A., and Pereira, F. (2001). Conditional random fields: probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth International Conference on Machine Learning, Massachusetts, USA*, pages 282–289.

Lage, K., Karlberg, E. O., Storling, Z. M., Olason, P. I., Pederson, A. G., Rigina, O., Hinsby, A. M., Tumer, Z., Pociot, F., Tommerup, N., Moreau, Y., and Brunak, S. (2007). A human phenome-interactome network of protein complexes implicated in genetic disorders. *Nature Biotechnology*, 25:309–316.

Leaman, R. and Gonzalez, G. (2008). BANNER: an executable survey of advances in biomedical named entity recognition. In *Proceedings of the Pacific Symposium on Biocomputing, Hawai'i, USA*, pages 652–663.

Magnini, B., Pianta, E., Popescu, O., and Speranza, M. (2006). Ontology population from textual mentions: task definition and benchmark. In *Proc. ACL/COLING Workshop on Ontology Population and Learning (OLP2), Sidney, Australia*, pages 26–32.

McDonald, R. and Pereira, F. (2005). Identifying gene and protein mentions in text using conditional random fields. *BMC Bioinformatics*, 6(Suppl 1:S6).

Rabiner, L. and Juang, B. (1986). An introduction to hidden Markov models. *IEEE ASSP Magazine*, pages 4–16.

Rebholz-Schuhmann, D., Jimeno-Yepes, A. J., van Mulligen, E. M., Kang, N., Kors, J., Milward, D., Corbett, P., Bukyo, E., Beisswanger, E., and Hanh, U. (2010). CALBC silver standard corpus. *Journal of Bioinformatics and Computational Biology*, 8(1):163–179.

Rindflesch, T. C., Hunter, L., and Aronson, A. R. (1999). Mining molecular binding terminology from biomedical text. In *American Medical Informatics Association (AMIA)'99 annual symposium, Washington DC, USA*, pages 127–131.

Robinson, P. N. and Mundlos, S. (2010). The human phenotype ontology. *Clinical Genetics*, 77(6):525–534.

Scheuermann, R., Ceusters, W., and Smith, B. (2009). Toward an ontological treatment of disease and diagnosis. In *AMIA Summit on Translational Bioinformatics, San Francisco, CA*, pages 116–120.

Schwartz, A. and Hearst, M. (2003). A simple algorithm for identifying abbreviations in biomedical text. In *Pacific Symposium on BioComputing, Hawai'i, USA*, pages 451–462.

Settles, B. (2004). Biomedical named entity recognition using conditional random fields. In *Proceedings of the International Joint Workshop on Natural Language Processing in Biomedicine and its Applications (JNLPBA) at COLING'2004, Geneva, Switzerland*, pages 104–107.

Smith, C. L. and Eppig, J. T. (2009). The mammalian phenotype ontology: enabling robust annotation and comparative analysis. *Wiley Interdisciplinary Reviews: Systems Biology and Medicine*, 1(3):390–399.

Suakkaphon, N., Zhang, Z., and Chen, H. (2011). Disease named entity recognition using semisupervised learning and conditional random fields. *Journal of the American Society for Information Science and Technology*, 62(4):727–737.

Tateisi, Y., Ohta, T., Collier, N. H., Nobata, C., and Tsujii, J. (2000). Building an annotated corpus from biology research papers. In *Proc. COLING 2000 Workshop on Semantically Annotated Corpora and Intelligent Content, Saarbrucken, Germany*, pages 28–34.

Tsuruoka, Y., Tateisi, Y., Kim, J. D., Ohta, T., McNaught, J., Ananiadou, S., and Tsujii, J. (2005). Developing a robust part-of-speech tagger for biomedical texts. In Bozanis, P. and Houstis, E., editors, *Advances in Informatics: 10th Panhellenic Conference on Informatics, Volos, Greece, Proceedings. LNCS*, pages 382–392. Springer.

Wu, X., Jiang, R., Zhang, M. Q., and Li, S. (2008). Network-based global inference of human disease genes. *Systems Biology*, 4(189).

Zhou, G., Zhang, J., Su, J., Shen, D., and Tan, C. (2003). Recognizing names in biomedical texts: a machine learning approach. *Bioinformatics*, 20(7):1178–1190.

Using Argumentative Zones for Extractive Summarization of Scientific Articles

Danish Contractor[1,2] *Yufan Guo*[1] *Anna Korhonen*[1]
(1) Computer Laboratory, University of Cambridge, Cambridge, United Kingdom
(2) IBM Research - India, New Delhi, India
{dc536,yg244,alk23}@cam.ac.uk

ABSTRACT
Information structure, i.e the way speakers construct sentences to present new information in the context of old, can capture rich linguistic information about the discourse structure of scientific documents. Information structure has been found useful for important Natural Language Processing (NLP) tasks, such as information retrieval and extraction. Since scientific articles typically follow a certain discourse structure describing the prior work, problem being solved, methods used, and so forth, it could also be useful for summarization of these articles. In this work we focus on a scheme of information structure called Argumentative Zoning (AZ), and investigate whether its categories could support extractive text summarization in a scientific domain. We develop a summarization system that uses AZ categories (i) as features and (ii) in the final sentence selection process. We evaluate the system directly as well as using task-based evaluation. The results show that AZ can support both full document and customized summarization. We report a statistically significant improvement in summarization performance against a competitive baseline that uses journal section labels instead of AZ information.

TITLE AND ABSTRACT IN MANDARIN

一种根据“论证结构”自动摘录科技文献的方法

信息结构是指作者组织语句陈述信息的方式。信息结构例如科技文献的篇章结构包含丰富的语言信息，有助于解决自然语言处理领域的一些重要问题例如信息检索和信息提取等。科技文献通常使用特定的篇章结构来陈述以往的研究，阐述研究问题以及研究方法等等，这些篇章结构可以被用于文献的自动摘录。本文着眼于一类特定的信息结构 — “论证结构”，研究其是否有助于更好地摘录科技文献。在本文开发的摘录系统中，“论证结构”有两种用途：一是作为特征供机器学习，二是用于最终的语句筛选过程。本文对该系统进行了直接和间接的评测，测试结果显示“论证结构”有助于更好地对全文或指定信息进行摘录。基于“论证结构”的摘录系统显著性优于基于章节标题的摘录系统。

KEYWORDS: discourse, information structure, argumentative zones, summarization, document summarization, information access.

KEYWORDS IN MANDARIN: 篇章，信息结构，论证结构，文本摘要，信息获取.

Proceedings of COLING 2012: Technical Papers, pages 663–678,
COLING 2012, Mumbai, December 2012.

1 Introduction

Information structure is the study of how writers package information into a sentence and convey new information (e.g. new methods, results and conclusions) in the context of old information (e.g. previous or related work) within a document. A number of frameworks capturing different aspects of information structure (including e.g. discourse, rhetorical, argumentative and conceptual) have been proposed, many of which focus on scientific documents (Teufel and Moens, 2002; Shatkay et al., 2008; Teufel et al., 2009; Liakata et al., 2010). Scientific documents are highly structured in nature, and knowledge about their information structure can support important Natural Language Processing (NLP) systems aimed at improving information access to scientific literature.

To date, information structure has proven useful for different information retrieval and extraction tasks, as well as for manual literature review (Tbahriti et al., 2006; Ruch et al., 2007; Guo et al., 2011b). One NLP task which is highly important for the scientific domain and which might similarly benefit from information structure is document summarization. Scientific articles are well structured containing sections such as "Prior work", "Method", "Experiments" etc, and also contain a rich network of citations. While section and citation based features have been exploited extensively in prior summarization works (see section 2), categories of information structure have received little attention. A good summarization system should be able to identify the "key" concepts in an article and generate a summary that has good coverage over the ideas expressed in the article. In the case of scientific documents, information structure can capture rich linguistic characteristics defined by eg. the discourse status of sentences, and therefore could help a summarization system select the right mix of sentences to be used from the document.

(Teufel and Moens, 2002) introduced a scheme of information structure called Argumentative Zoning (AZ) which classifies sentences in scientific text into categories (such as Aim, Background, Own, Contrast and Basis) on the basis of their rhetorical status in scientific discourse. They performed experiments which show that AZ can be used to identify and summarize novel contributions as well as background information in a scientific article. However, they did not investigate integrating AZ in an automatic summarization system.

In this paper we focus on this topic and explore whether knowledge about information structure could be used to support an actual summarization system performing extractive summarization in the scientific domain. Like (Teufel and Moens, 2002), we focus on AZ because this scheme has aided many other NLP tasks and has shown wide applicability across different scientific domains (including e.g. computational linguistics, chemistry, biology). Experimenting on biomedical corpus data, we use the version of AZ adapted for biology by (Mizuta et al., 2006).

We develop a simple summarization system for evaluation purposes and use it as a framework when investigating two approaches to integrating both manually and automatically obtained AZ categories into summarization: (i) including them as features in a classification task for selecting sentences that should be part of a summary, along with other features that have traditionally been found useful in such a classification task, and (ii) using them as a selection mechanism for identifying the final set of sentences that should be made part of the summaries.

We evaluate these approaches via two task-based evaluations - extractive summarization of complete articles as well as generation of customized summaries based on user requirements. We also compare their performance against a competitive baseline that makes use of section labels (instead of AZ labels) in biomedical articles. To the best of our knowledge, this is

the first work that compares the use of section labels against AZ in summarization. The results are promising. They demonstrate that AZ can be an effective feature to include in summarization systems and can improve the quality of summaries generated for scientific papers. Both manually and automatically obtained AZ labels prove useful. In the future, the approach could be optimised for integration in state of the art summarization systems as well as used for task-based evaluation and comparison of different automatic AZ lebeling systems.

2 Related work

2.1 Summarization in Scientific Literature

Scientific literature continues to be a major domain for summarization research along with news articles. Different types of summaries can be generated for such documents. For example, one could be interested in automatically generating an abstract-like summary for an article or a group of articles, or one may require a customized summary describing specific types of information (e.g. experiments or results) in articles only. Further, sentences in such summaries may be a paraphrased representation of the information present in the original documents (abstractive summarization) or may be a subset of those in the original article (extractive summarization).

An important characteristic of scientific articles is the presence of citations. Citations have been used in different summarization systems. Recent work such as that by (Abu-Jbara and Radev, 2011), (Qazvinian et al., 2010) and (Qazvinian and Radev, 2010) make use of citation sentences in other scientific papers to summarize the contributions of a paper. Although it is the ideas of one paper that are being summarized, this approach involves searching for references to the paper in other papers, and extracting sentences from them to build summaries.

Other recent work such as that of (Qazvinian and Radev, 2010) uses Markov Random Fields to detect patterns that create context data (background information) for a paper, while (Mei and Zhai, 2008) use citations as a measure of "impact" in a field and use it for summarization.

Latent Semantic Analysis based methods have also been used for summarization of documents. (Steinberger et al., 2005) use LSA along with anaphora resolution to improve document summarization while (Ozsoy et al., 2010) propose multiple LSA based summarization algorithms in which the sentence selection criteria is modified using the "concept" matrix derived at the end of singular value decomposition (SVD) step.

In this paper we use argumentative zones (AZ) for extractive summarization of scientific papers. Extractive summarization generates summaries by selecting a subset of the sentences from the original document. We use a classifier trained using the author-created abstracts of articles to identify sentences from the full document for a system-generated summary and use clustering to further select sentences. AZ information is used both as a feature in classification as well as a guiding step during clustering.

2.2 Argumentative Zoning and NLP Tasks

Argumentative Zoning (AZ) classifies sentences from scientific text based on their rhetorical status in terms of problem solving (e.g."What are the contributions of the paper?"), intellectual attribution (sentences that describe prior work etc) and relatedness amongst articles. The original AZ scheme of (Teufel and Moens, 2002), applied to the domain of computational linguistics, included five rhetorical zone categories (Aim, Background, Own, Contrast and Basis)

and a fully supervised classifier was trained to classify each sentence in scientific articles in one of these categories. Subsequent work and applications of this scheme to other domains (e.g. chemistry, biology) have resulted in finer-grained AZ classifications.

Most approaches to automatic AZ detection rely on fully supervised machine learning. A high accuracy above 80% have been reported with the best of these approaches. (Guo et al., 2011a) has developed an approach based on active learning which performs (as its best) as well as fully supervised approaches but requires only a small amount of labeled data.

Most work on information structure has been evaluated directly on manually annotated data sets. Previous task-based evaluations, mostly conducted on AZ, include information retrieval and extraction tasks, along with literature review in biomedicine (Tbahriti et al., 2006) (Ruch et al., 2007), (Guo et al., 2011b).

(Teufel and Moens, 2002) reported experiments which suggest that AZ should also be helpful for automatic summarization. They used zones to identify and "summarize" new contributions in scientific papers. Sentences from the Aim, Contrast and Basis zones were used to highlight new contributions of a paper. When including information about "background work" in a summary, directly using sentences labeled with the Background zone reduced precision. They therefore trained a classifier based on annotated data that identifies sentences from the Background zone for a short "summarized" version of the document. Although this work suggests that AZ could be useful for summarization, it does not develop or employ an actual summarization system.

Related work by (Farzindar and Lapalme, 2004) used "thematic" structures (rather than AZ) (Introduction, Context, Judicial Analysis and Conclusion) in law judgments to generate summaries. They identified "cue" strings for each of these themes and used verb classes to filter out citation sentences. To summarize the text they used a heuristic function based on position of paragraphs in a document, position of paragraph in a thematic segment, tf-idf distribution and cue words specific to each theme. The summary lengths were controlled by using the distribution of themes in the abstract to select a proportionate number of sentences from each theme. However, scientific articles differ from law judgments because the documents are highly structured and contain "sections" which are defined by the authors.

In this paper we investigate whether AZ could be used to support a summarization system in the scientific domain. We focus on biomedicine and experiment with the AZ scheme developed for biology by (Mizuta et al., 2006). We employ manual AZ annotations in the main experiments (in order to investigate the direct impact of AZ on summarization) but also report experiments where automatic annotations from the weakly supervised AZ labeling system of (Guo et al., 2011a) are used. We integrate the AZ labels into a summarization system as features and also use them to aid the final sentence selection process in summarization. We perform both direct and task-based evaluation which shows that both methods can support automatic summarization.

3 Method

Most scientific papers contain an abstract which provides a short description of the work presented in the paper. The abstracts are created by the authors and can be regarded as summaries of papers. Using such abstracts as the gold-standard we describe a method for generating summaries. The summaries can vary in length (i.e. be more elaborate or concise than the original abstracts) which is useful in the scientific domain where users (e.g. scientists) have highly varied summarization needs.

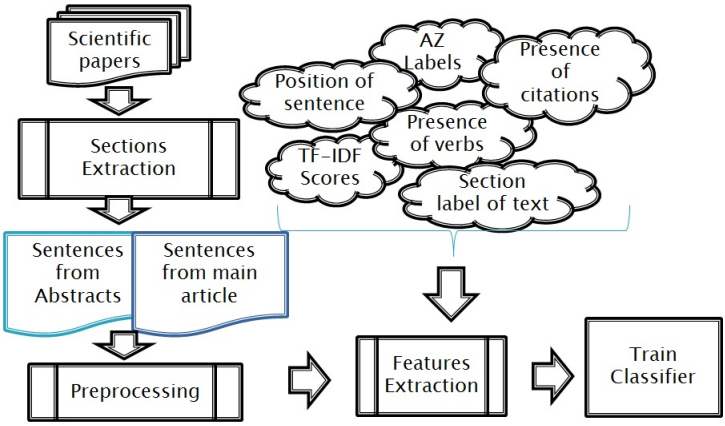


Figure 1: Schematic diagram showing the training phase of the summarization system.

Our method has two main stages: classification and sentence clustering. The classifier creates an initial candidate set of sentences for the summary, and the sentence clusterer identifies groups of similar sentences in this set which are then used to create the final summary. The clustering step, employed by many summarization systems, removes redundancy from the candidate pool and thus improves the quality of the summary.

Figure 1 gives an overview of the training phase of the summarization system. Sentences from the training articles are pre-processed as described in section 4.1. and annotated with the section labels (i.e. the section of the article to which the sentence belongs) and zone labels. They also undergo stop word removal and lemmatization. After pre-processing, the feature vector representations of the sentences are created and used to train a classifier using the Weka tool kit[1].

After training, the system can accept documents for summarization. An article is pre-processed and its feature vector representation is created as during training. A parameter specified by a user controls the compression ratio by adjusting the classifier threshold as well the number of clusters used. After classification, the positively labeled instances are filtered using a sentence clusterer, and a final summary is generated. Figure 2 shows an overview of the execution phase of the summarization system.

The next section describes the actual methods and features used for classification. Section 3.2 gives details about the clustering stage and section 3.3 describes how the parameter specifying the compression ratio is used to adjust the length of the summaries.

3.1 Sub-component for classification

Let A be the set of sentences in the abstracts of papers, and let D be the sentences in the main sections of the papers. Using the set of sentences in A and D, we trained a classifier that learns

[1]http://www.cs.waikato.ac.nz/ml/weka

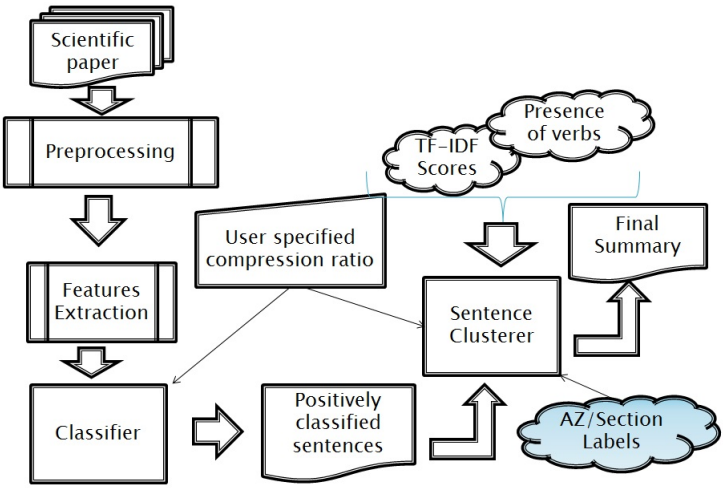


Figure 2: Schematic diagram showing the execution phase of the summarization system.

how to generate a set of sentences $C, C \subseteq D$, which is the candidate set of sentences that are to be made part of a summary.

We require both positive and negative labeled instances for training a classifier. The sentences in the abstract can be considered as positive labeled instances but those in the main text are unlabeled, i.e. they could be positive or negative. The problem of training a classifier using positive and unlabeled data has been studied before. A state-of-the-art method has been described in (Elkan and Noto, 2008), where the classifier is built using positive and unlabeled instances. The model predicts probabilities that differ by a constant factor from the actual conditional probabilities of being positive. Using the constant factor, one can estimate the probabilities of positive and negative instances. We employ this method to train a classifier for selecting the candidate set of sentences (referred to as the "non-traditonal classifier" based method in section 4.3.2).

Instead of using the positive and unlabeled data, one could artificially generate some labeled data and use that for training a classifier. Consider a similarity metric $\omega(S_1, S_2)$ that returns a score indicating the similarity between two sentences S_1 and S_2. Using this similarity metric ω [2] we can identify those sentences in the main text that are most similar to the sentences in the abstract and label them as positive instances, while the others can be labeled as negative instances. A traditional classifier model based on Support Vector Machines was built using these positive and negative instances.

[2]We used two measures – n-gram overlap based similarity between sentences and cosine similarity between sentence feature vectors – and found that the latter gave better results.

3.1.1 Features used for classification

We used various features for representing the training and test samples used by the classifier, including the new AZ feature as well as features that have proved successful in previous related works, e.g. (Abu-Jbara and Radev, 2011).

- Verbs feature: Sentences in the abstract tend to contain many verbs. For example, text fragments like "We showed that", "we found that", "We used", "we proved" are very common in abstracts, often using past tense. Using the StanfordNLP Part-of-speech (POS) tagger (Toutanova et al., 2003), each sentence was tagged and verbs along with their tenses were identified and included in the feature set for that sentence.
- tf-idf values: TF-IDF scores for each sentence were used as features for the sentences. The size of tf-idf features is the size of the vocabulary. Each word in the vocabulary along with its tf-idf value in a sentence were used as features.
- Citation and reference occurrences: Sentences containing citations are frequently found where published related work is discussed. They tend to occur in the background, prior or related work sections. We also keep track of sentences that contain references to figures and tables. These point to a section in the same document, while a citation points to a different document. Each sentence is assigned two boolean features, indicating the presence of a citation and a reference to figures or tables, respectively.
- Argumentative zones: Each sentence is labeled with one of eight AZ categories both manually and automatically using the system of (Guo et al., 2011a).
- Locative features: Sentences tend to have locative characteristics, e.g. most sentences describing prior work occur in the beginning of a paper, while those describing future work tend to occur at the end. Position of sentences has earlier been found to be useful in summarization tasks (Baxendale, 1958) (Conroy and O'leary, 2001).

We experimented with different combinations of these features. Citation and location based features were only used by the traditional classifier, as these features are unavailable when the original abstracts are used directly for training.

3.2 Sub-component for clustering

A well rounded summary should briefly describe the nature of the problem, the work conducted, and the nature of the results obtained, without repeating any information. Once the sentences have been classified, a clustering step is used to remove redundancy from them and to identify similar sentences. Using the section labels to group similar sentences, we applied the k-means clustering (Lloyd, 1982) to detect clusters within each group. By selecting the centroid from each of the k-clusters in each section group, the final set of sentences were identified for a summary.

An alternative to using the section labels for grouping sentences is to make use of AZ. Sentences with the same AZ label can be grouped together, and the clusters can be identified within each group. We experimented with this option as well. The feature vector representation of sentences used for clustering consists of tf-idf weights as well as variables indicating the presence or absence of verbs.

3.3 Controlling the length of summaries

The compression ratio of a summary is defined as the ratio of the number of sentences in the summary generated and the number of sentences in the original article. Using the compression ratio as an input parameter to the system, the length of the summaries can be controlled.

The compression ratio is used to adjust the classification threshold of the classifier. If the number of sentences is too low, the classification threshold is reduced by a fixed step size and sentences are re-classified. This process is repeated until the the compression ratio is a fixed constant t % more than the actual compression ratio required. This relaxed compression ratio is used so that the clustering stage has enough sentences to choose from.

In clustering, the compression ratio is used to determine the number of sub-clusters to be created within each AZ/section label group. Let the compression ratio be denoted by cr, the length of original document be l and the number of distinct AZ labels/section labels in the classified set be m. Then the number of clusters k is given by :

$$k = ceil(\frac{cr * l}{m}) \quad (1)$$

where $ceil(x)$ is a function that returns the smallest integral value that is greater than or equal to the real number x.

4 Experiments and Results

4.1 Data and Pre-processing

We used a corpus of 50 biomedical articles sourced from a number of journals on cancer which are available online at PubMed[3]. The corpus contains 580 sentences in the abstracts and 7,989 sentences in the main body. The sentences were annotated according to the AZ annotation scheme of (Mizuta et al., 2006). Eight AZ categories[4] appeared in the annotated data[5], including Background, Conclusion, Problem, Connection, Method, Difference, Result and Future work. Inter-annotator agreement between the two annotators (one domain expert and one computational linguist) was high $\kappa = 0.83$ according to Cohen's kappa (Cohen, 1960).

	Sentence Type	Training data	Test data	Validation set
Biomedical corpus	Abstract sentences	308	206	66
	Main article sentences	5046	2943	-

Table 1: Details of data set

For the experiments with automatically obtained AZ labels, we used the weakly-supervised method of (Guo et al., 2011a) to identify the AZ category of each sentence. Based on the active learning and self-training, the method was trained using just 10% of labeled data in a corpus of 1000 biomedical articles. With accuracy of 81% it performs similarly with supervised methods that employ all the labels (Guo et al., 2010).

[3]http://www.ncbi.nlm.nih.gov/pubmed/

[4]Please see the paper of (Mizuta et al., 2006) for the full details of the annotation scheme and examples of different zone categories.

[5]The data and the source code of the methods described in this paper are available on request.

We split the articles in our corpus into 3 sets for training, testing and validation. The validation set was created from a small set of sentences from abstracts and was used to learn a classifier from positive and unlabeled training samples (Elkan and Noto, 2008).

All sentences were tagged using the Stanford NLP POS tagger (Toutanova et al., 2003) with the Penn treebank tagset[6]. In addition, the articles were lemmatized using the Stanford NLP lemmatizer[7] and stop words were removed using stop-lists available on the Internet.

4.2 Experiments

We evaluated the method on two tasks: full document and customized summarization. In full document summarization, a user-specified compression ratio is used to automatically summarize the contents of the entire article. In customized summarization, the user specifies the length and the focus of the summary to be generated (e.g. a summary of the "methods" described in the paper only).

4.2.1 Evaluation Measures

The ROUGE-N measure (Lin, 2004) is used frequently for evaluation of summarization systems. ROUGE stands for Recall Oriented Understudy for Gisting Evaluation and the ROUGE-N score is calculated by counting the number of overlapping N-grams between a user generated/reference summary and a system summary. ROUGE does not consider the length of the summaries, and therefore, if an entire article is returned, it could get the best ROUGE score as the number of n-gram matches will be high. Therefore, compression ratios (cr), i.e. ratio of the number of sentences in the summary ($|S_{summary}|$) generated and the number of sentences in the original article ($|S_{article}|$) are also frequently used.

$$cr = \frac{|S_{summary}|}{|S_{article}|} \tag{2}$$

We used as the primary evaluation metric the F_1 [8] measure, calculated using the number of overlapping n-grams between the summary generated at different compression ratios and the abstracts created by the authors.

4.3 Full article summarization

In this task, sentences from the abstracts were used to learn how to generate full length summaries of articles. Different combinations of features were used to train classifiers.

4.3.1 Training

Sentences from the abstracts were used to train a non-traditional classifier of (Elkan and Noto, 2008) and to create an artificial set of positive and negative instances for training the traditional classifier. The artificial data set was created by selecting such sentences from the main text that were similar to sentences from the abstract. The similarity criteria was based on the cosine distance between the feature vectors of the sentences. The following section describes the results for both these methods with different combinations of features.

[6] http://www.ims.uni-stuttgart.de/projekte/CorpusWorkbench/CQP-HTMLDemo/PennTreebankTS.html
[7] http://www-nlp.stanford.edu/nlp/javadoc/javanlp/edu/stanford/nlp/process/Morphology.html
[8] http://en.wikipedia.org/wiki/F1_score

Classification features	Cluster groups created using	Precision (P)	Recall (R)	F1 Measure
Verb, tf–idf	None	0.1966	0.3926	0.2620
	Section label	0.1946	0.4058	0.2630
	AZ Label	**0.2091**	**0.4301**	**0.2814**
Verb,tf–idf, AZ	None	0.1938	0.3766	0.2559
	Section label	0.1861	0.3985	0.2537
	AZ Label	0.2054	0.4287	0.2777

Figure 3: Full Document summarization results using non-traditional classifier and compression ratio of 10 %

4.3.2 Results

Figure 3 shows the performance of the non-traditional classifier methods for summarization when manual AZ labels are used. We varied the compression ratio between 5% and 25% in steps of 5% and studied the performance. We report here the results using the compression ratio of 10% because it produces summary length that corresponds the closest to the length of the actual abstracts. As can be seen, during clustering, the use of AZ labels considerably improves the F_1 scores and also improves both precision and recall (ROUGE-1). The features used during classification are the verb features, the tf-idf features and the AZ label features. Other features (e.g. locative and citation based) were not used in this method as they are unavailable in the sentences from the abstracts and in the training data.

Figure 4 shows the performance of the traditional classifier based summarization, again using manual AZ labels. Here the results are better when clustering employs the AZ labels instead of section labels for grouping sentences. Features based on location of sentences and citations were used as the training data consists of sentences from the main text. The use of citation features along with locative features improves the performance of the summarizer, though the non-traditional classifier outperforms the traditional classifier.

The use of section labels for clustering is a tough baseline to beat, as these sections contain sentences that the authors themselves deemed fit to belong to those sections. For example, sentences belonging to a section called “Result” could be considered to identify sentences relating to “Results”, but the use of sentences belonging to the argumentative zone called “Result” are not confined just to the “Result” section, and the use of the AZ labels shows a significant improvement in performance. We also experimented using clustering without creating cluster groups based on sections or AZ and found that the use of AZ labels improves performance. The improvement in F_1 scores when using AZ during the clustering stage was found to be statistically significant ($p < 0.03$).

Finally, we performed an experiment using automatically detected AZ labels by the method of (Guo et al., 2011a). Using the best feature configurations for the task, this results in a small (2%) drop in F_1 scores when compared to the use of manual labels, but slight improvement when compared to the use of section labels (See Figure 5).

Classification features	Cluster groups created using	Precision (P)	Recall (R)	F1 Measure
Verb, tf–idf	None	0.1984	0.3779	0.2602
	Section label	0.1966	0.3827	0.2597
	AZ Label	0.1994	0.3993	0.2660
Verb,tf–idf, AZ	None	0.1952	0.3871	0.2595
	Section label	0.1936	0.3950	0.2599
	AZ Label	0.1953	0.4215	0.2669
Verb,tf–idf, AZ, Citation	None	0.2063	0.3288	0.2535
	Section label	0.2061	0.3290	0.2535
	AZ Label	0.2146	0.3372	0.2632
Verb, tf–idf, AZ, Citation, Locative	None	0.2066	0.3882	0.2697
	Section label	0.2048	0.3733	0.2644
	AZ Label	**0.2102**	**0.3781**	**0.2707**

Figure 4: Full Document summarization results using traditional classifier and compression ratio of 10 %

Classification features	Cluster groups created using	Precision (P)	Recall (R)	F1 Measure
Verb, tf–idf (Non Traditional Classifier)	None	0.1966	0.3926	0.2620
	Section label	0.1946	0.4058	0.2630
	Weakly Supervised AZ Label	0.2096	0.4010	0.2753
	AZ Label	**0.2091**	**0.4301**	**0.2814**

Figure 5: Performance when using AZ labels automatically generated using the weakly supervised method of (Guo et al., 2011a)

4.4 Customized summarization

In this task, the system generates summaries for some parts of the paper based on user requirements. We evaluated the system on two customized summary tasks that our experts found useful for biomedical literature review: the summarization of "Results" in a paper and the "Discussion" in a paper. The evaluation was done against gold-standard summaries for 50 articles, generated by a human expert (an expert in biomedical research). The expert was asked to generate the customized summaries by selecting sentences from the main body of the articles and to ensure that the summaries generated do not exceed 40 % of the full article length. The gold-standard summaries were also used for training as described in the next section.

4.4.1 Training

The gold-standard summary set was split for training and testing (60-40 % split). The sentences from the gold-standard were used as positive labels for training the classifier, while the negative instances were labeled in two ways.

In the first method, sentences from the main article, which were not part of the gold-standard summary, were labeled as negative instances. Thus, in this case, the whole set of sentences from the article is available for training. In the second method, the AZ labels of the sentences are used to get a reduced set of negative instances. Sentences from the zone best suited to the customized summary task are selected. For example, in the customized summarization task for "Results"' in a paper, the negative instances will only contain sentences which are not part of the gold-standard and which belong to the "Results" zone. This method is referred to as "Zone pre-selection"' in the results section.

4.4.2 Results

For the customized summarization task, the summary lengths are controlled based on the number of sentences desired instead of the compression ratio. Estimating a compression ratio based on the full length of the article is not suitable because the task focuses on summarizing a "part" of the article (only the "results" for example), and not the entire article. The "part" of the article that is summarized is based on the "type" of information the user is interested in, and is not by itself, explicitly demarcated in the original article.

In this task, therefore, if x is the number of sentences required in the summary, the the number of clusters are given by:

$$n = ceil(\frac{x}{m}) \tag{3}$$

When zone pre-selection is employed, the clustering stage was found to be less useful for the "Results" and "Discussion" summarization tasks, because the classification stage itself was able to identify a good set of sentences to be used for the summary. The problem of redundant information is reduced, because the summaries are generated of some "parts" of the document and not the entire document, reducing chances of redundancy.

It must also be noted that this behaviour may not be universally true for all types of customized summaries. For example, summarization of the "Background" work in a paper, which usually contains more information and tend to be larger sections, may contain redundant information and the classifier may not be as selective due to the increase in length of the text being summarized.

Customized Summary Type	Features	Precision	Recall	F1
Summarize Results	Verbs, tf–idf	0.1649	0.4766	0.2451
	Verbs,tf–idf,AZ	0.1613	0.4906	0.2428
	Verbs,tf–idf, Zone pre–selection	**0.3312**	**0.6253**	**0.4330**
Summarize Discussion	Verbs, tf–idf	0.1178	0.3318	0.1739
	Verbs,tf–idf,AZ	0.1285	0.3354	0.1858
	Verbs,tf–idf, Zone pre–selection	**0.1838**	**0.4959**	**0.2682**

Figure 6: Customized Summarization results for summary length of 15 sentences

Customized Summary Type	Zone Pre–selection	Precision	Recall	F1
Summarize Results	Weakly Supervised AZ	0.3257	0.6145	0.4257
	Manually labeled AZ	0.3312	0.6253	0.4330
Summarize Discussion	Weakly Supervised AZ	0.1867	0.4882	0.2702
	Manually labeled AZ	0.1838	0.4959	0.2682

Figure 7: Comparison of customized summarization results using weakly supervised and manually labeled AZ

Figure 6 presents results for the two customized summarization tasks. The use of zone pre-selection improves the results considerably against the baseline system which does not make use of zone pre-selection. There is a 76 % improvement in F_1 scores in the "Results" task and an improvement of 54 % in the "Discussion" task. It can also be seen that the use of AZ as a feature for classification does not cause significant change in performance. This was also noticed in the full document summarization task, where AZ were found to be most effective during the clustering stage and not during the classification stage.

Figure 7 compares the performance of the best performing customized summarization configuration (Zone pre-selection) using weakly supervised AZ labels. An analysis of the errors made by the weakly supervised automated AZ labeling method showed that the "Results" and "Conclusion" AZ labels account for 9% and 20% of the errors respectively. The "Conclusion" AZ category is one where the annotators to have most disagreement, partly because many sentences include elements of both discussion and some other zone (e.g. methods or results), yet annotators are asked to assign each sentence to one category only. Nevertheless, our experiments shows that the performance of the summarization system when using automatically generated AZ labels is comparable to that of a system using manually labeled AZ labels.

The results presented in this section are promising, showing that AZ can yield improvements in both full document and customized summarization tasks in biomedicine.

Conclusion and Future Work

Most work on the information structure of scientific literature has been evaluated directly against manually labeled data. Task-based evaluation has mainly concentrated on information retrieval and extraction tasks. We have investigated whether AZ could be used to benefit summarization of scientific articles. Although previous work had suggested that AZ can improve summarization, no experiment had been conducted using a full AZ scheme and a real summarization system.

We developed a simple summarization system that uses a classifier to identify a set of candidate sentences, and uses clustering along with AZ labels to reduce redundancy in the summaries generated. The system is capable of creating full document summaries of different length and information density as well as customized summaries based on user requirements. Both types of summaries can be helpful for users in the scientific domain.

We evaluated the summarization performance on both full document and customized summarization and reported statistically significant improvement in performance scores when using AZ labels. The system outperforms a strong baseline method that uses section labels instead of the AZ labels. The improvement of approximately 7 % in F_1 scores in the full document summarization and an improvement of 54-76 % in customized summarization clearly shows that AZ can benefit automatic summarization.

Our main focus was on manual AZ annotations because we wanted to investigate the direct impact and the upper bound of AZ on summarization. However, also our pilot experiments using automatic AZ annotations show improvement in summarization performance. Future work could use our method as a framework for task-based evaluation of AZ labeling systems.

In this initial investigation on the topic, we kept the summarization framework intentionally simple for evaluation purposes. Future work could optimise the use of AZ for state-of-the-art summarization systems and also explore further ways of integrating AZ in the task. For example, our experiments show that zones are useful for building better clusters. Instead of employing clustering to reduce redundancy, one could investigate the use of diversity ranking algorithms. Once the sentences have been grouped based on zone labels, as described in section 3.2, diversity ranking algorithms, e.g. (Radlinski et al., 2008), could be used to obtain a ranked list of topically or information "diverse" sentences, from which the summary could be built.

Alternatively, instead of using a classification and clustering based approach, sentences from the main article could be selected based on a diversity ranking algorithm and then the final summary could be built using the distribution of zones in the abstracts or gold standard summaries as a "summary template".

Although we focused on AZ due to its good applicability to different scientific domains, success in previous task-based evaluations, and the availability of a weakly-supervised AZ detection method which enables easy porting between NLP tasks, it would be interesting to investigate and compare the usefulness of other schemes of information structure for summarization.

Acknowledgement

The first author's studies at the University of Cambridge were partially funded by the Cambridge Overseas Trust. We would also like to thank Dr. Ilona Silins, University of Cambridge for creating the gold-standard summaries for the customized summarization task. The work in this paper was also funded by the Royal Society (UK), EPSRC (UK) grant EP/G051070/1 and EU grant 7FP-ITC-248064.

References

Abu-Jbara, A. and Radev, D. (2011). Coherent citation-based summarization of scientific papers. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1*, HLT '11, pages 500–509, Stroudsburg, PA, USA. Association for Computational Linguistics.

Baxendale, P. B. (1958). Machine-made index for technical literature: an experiment. *IBM J. Res. Dev.*, 2(4):354–361.

Cohen, J. (1960). A coefficient of agreement for nominal scales. *Educational and Psychological Measurement*, 20(1):37–46.

Conroy, J. M. and O'leary, D. P. (2001). Text summarization via hidden markov models. In *Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '01, pages 406–407, New York, NY, USA. ACM.

Elkan, C. and Noto, K. (2008). Learning classifiers from only positive and unlabeled data. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '08, pages 213–220, New York, NY, USA. ACM.

Farzindar, A. and Lapalme, G. (2004). Letsum, a text summarization system in law field. In *THE FACE OF TEXT conference (Computer Assisted Text Analysis in the Humanities)*, pages 27–36, McMaster University, Hamilton, Ontario, Canada.

Guo, Y., Korhonen, A., Liakata, M., Karolinska, I. S., Sun, L., and Stenius, U. (2010). Identifying the information structure of scientific abstracts: an investigation of three different schemes. In *Proceedings of the 2010 Workshop on Biomedical Natural Language Processing*, BioNLP '10, pages 99–107, Stroudsburg, PA, USA. Association for Computational Linguistics.

Guo, Y., Korhonen, A., and Poibeau, T. (2011a). A weakly-supervised approach to argumentative zoning of scientific documents. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '11, pages 273–283, Stroudsburg, PA, USA. Association for Computational Linguistics.

Guo, Y., Korhonen, A., Silins, I., and Stenius, U. (2011b). Weakly supervised learning of information structure of scientific abstracts - is it accurate enough to benefit real-world tasks in biomedicine? *Bioinformatics*, 27(22):3179–3185.

Liakata, M., Teufel, S., Siddharthan, A., and Batchelor, C. R. (2010). Corpora for the conceptualisation and zoning of scientific papers. In Calzolari, N., Choukri, K., Maegaard, B., Mariani, J., Odijk, J., Piperidis, S., Rosner, M., and Tapias, D., editors, *LREC*. European Language Resources Association.

Lin, C. Y. (2004). Looking for a few good metrics: Automatic summarization evaluation - how many samples are enough? In *Proceedings of the NTCIR Workshop 4*.

Lloyd, S. P. (1982). Least squares quantization in pcm. *IEEE Transactions on Information Theory*, pages 129–136.

Mei, Q. and Zhai, C. (2008). Generating impact-based summaries for scientific literature. In *Proceedings of ACL-08: HLT*, pages 816–824, Columbus, Ohio. Association for Computational Linguistics.

Mizuta, Y., Korhonen, A., Mullen, T., and Collier, N. (2006). Zone analysis in biology articles as a basis for information extraction. *I. J. Medical Informatics*, pages 468–487.

Ozsoy, M. G., Cicekli, I., and Alpaslan, F. N. (2010). Text summarization of turkish texts using latent semantic analysis. In *Proceedings of the 23rd International Conference on Computational Linguistics*, COLING '10, pages 869–876, Stroudsburg, PA, USA. Association for Computational Linguistics.

Qazvinian, V. and Radev, D. R. (2010). Identifying non-explicit citing sentences for citation-based summarization. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, ACL '10, pages 555–564, Stroudsburg, PA, USA. Association for Computational Linguistics.

Qazvinian, V., Radev, D. R., and Özgür, A. (2010). Citation summarization through keyphrase extraction. In *Proceedings of the 23rd International Conference on Computational Linguistics*, COLING '10, pages 895–903, Stroudsburg, PA, USA. Association for Computational Linguistics.

Radlinski, F., Kleinberg, R., and Joachims, T. (2008). Learning diverse rankings with multi-armed bandits. In *Proceedings of the 25th international conference on Machine learning*, ICML '08, pages 784–791, New York, NY, USA. ACM.

Ruch, P., Boyer, C., Chichester, C., Tbahriti, I., Geissbuhler, A., Fabry, P., Gobeill, J., Pillet, V., Rebholz-Schuhmann, D., Lovis, C., and Veuthey, A.-L. (2007). Using argumentation to extract key sentences from biomedical abstracts. *I. J. Medical Informatics*, 76(2-3):195–200.

Shatkay, H., Pan, F., Rzhetsky, A., and Wilbur, W. J. (2008). Multi-dimensional classification of biomedical text. *Bioinformatics*, 24(18):2086–2093.

Steinberger, J., Kabadjov, M. A., and Poesio, M. (2005). Improving lsa-based summarization with anaphora resolution. In *In Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*, pages 1–8.

Tbahriti, I., Chichester, C., Lisacek, F., and Ruch, P. (2006). Using argumentation to retrieve articles with similar citations: An inquiry into improving related articles search in the medline digital library. *I. J. Medical Informatics*, 75(6):488–495.

Teufel, S. and Moens, M. (2002). Summarizing scientific articles - experiments with relevance and rhetorical status. *Computational Linguistics*, 28:2002.

Teufel, S., Siddharthan, A., and Batchelor, C. (2009). Towards discipline-independent argumentative zoning: evidence from chemistry and computational linguistics. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 3 - Volume 3*, EMNLP '09, pages 1493–1502, Stroudsburg, PA, USA. Association for Computational Linguistics.

Toutanova, K., Klein, D., Manning, C. D., and Singer, Y. (2003). Feature-rich part-of-speech tagging with a cyclic dependency network. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology - Volume 1*, NAACL '03, pages 173–180, Stroudsburg, PA, USA. Association for Computational Linguistics.

Annotation Tools and Knowledge Representation for a Text-To-Scene System

Bob Coyne[1] *Alex Klapheke*[1]
Masoud Rouhizadeh[2] *Richard Sproat*[3] *Daniel Bauer*[1]
(1) Columbia University
(2) Oregon Health Sciences University
(3) Google
coyne@cs.columbia.edu, agk2118@columbia.edu, masoud@cslu.ogi.edu,
rws@xoba.com, bauer@cs.columbia.edu

ABSTRACT
Text-to-scene conversion requires knowledge about how actions and locations are expressed in language and realized in the world. To provide this knowlege, we are creating a lexical resource (*VigNet*) that extends FrameNet by creating a set of intermediate frames (*vignettes*) that bridge between the *high-level* semantics of FrameNet frames and a new set of *low-level* primitive graphical frames. Vignettes can be thought of as a link between *function* and *form* – between what a scene means and what it looks like. In this paper, we describe the set of primitive graphical frames and the functional properties of 3D objects (*affordances*) we use in this decomposition. We examine the methods and tools we have developed to populate VigNet with a large number of action and location vignettes.

KEYWORDS: text-to-scene conversion, world knowledge, frame semantics, visual semantics, linguistic annotation, crowdsourcing.

Proceedings of COLING 2012: Technical Papers, pages 679–694,
COLING 2012, Mumbai, December 2012.

Figure 1: Mocked-up scenes using the WASH-FRUIT-IN-SINK vignette ("John washes the apple") and WASH-FLOOR-W-SPONGE vignette ("John washes the floor").

1 Introduction

3D graphics authoring is a difficult process, requiring users to master a series of complex menus, dialog boxes, and often tedious direct manipulation techniques. Natural language offers an interface that is intuitive and immediately accessible to anyone, without requiring any special skill or training. The WordsEye system (Coyne and Sproat, 2001) lets users create 3D scenes by describing them in language. It has been used by several thousand users to create over 10,000 scenes by merely describing them. We have tested WordsEye as an educational tool with rising 5th grade children in a summer enrichment program where it was found to significantly improve literacy skills over the students who had taken the more traditional version of the course (Coyne et al., 2011b). As one of the students said "When you read a book, you don't get any pictures. WordsEye helps you create your own pictures, so you can picture in your mind what happens in the story." The students were also introduced to WordsEye's face and emotion manipulation capabilities – the children loved including themselves and other people in scenes and modifying the facial expressions. We are currently experimenting with automatically depicting Twitter tweets as a way to bring text-to-scene visualization to a wider audience and to test the limits of the system in an open domain with ill-formed text. In this paper we describe a new set annotation tools and the graphical primitives we have developed in order to build a knowledge base that maps linguistic constructs into semantic frames representing spatial relations and other graphical relations.

WordsEye currently focuses on directly expressed spatial relations and other graphically realizable properties. As a result, users must describe scenes in somewhat stilted language. Our goal is to build a comprehensive text-to-scene system that can handle a wide range of input text. When considering sentences such as *John is washing an apple* and *John is washing the floor*, it becomes apparent that different graphical knowledge is needed to generate scenes representing the meaning of these two sentences (see Figure 1): the human actor is assuming different poses, he is interacting differently with the thing being washed, and the water, present in both scenes, is supplied differently. If we consider the types of knowledge needed for scene generation, we find that we cannot simply associate a single set of knowledge with the English verb *wash*. Instead we need to take into account the arguments of the verb as well.

Knowledge about verbs and their arguments is encoded in FrameNet (Ruppenhofer et al., 2010). FrameNet is a lexical resource focused on representing semantic relations and their possible instantiations in lexical items. In FrameNet, lexical items are grouped into frames that represent their shared semantic structure. A FrameNet frame consists of a set of frame-based roles, called *frame elements* (FEs). For example, the COMMERCE_SELL frame includes frame elements for Seller, Goods, and Buyer. These and other FEs represent the key roles that characterize the

meaning of the lexical units in that frame. Frames can contain any number of individual lexical units. The COMMERCE_SELL frame, for example, has lexical units for words like *retail*, *sell*, and *vend*. The exact expression of FEs for a given sentence constitutes what FrameNet refers to as a *valence pattern*. Valence patterns map grammatical roles to frame element for a given verb. Every verb typically has many valence patterns, representing the various ways that verb can be used in sentences. So, for the verb *give*, the sentence *John gave the book to Mary* has the valence pattern of: ((Donor Subject) (Theme Object) (Recipient Dep/to)). And *John gave Mary the book* has the valence pattern of ((Donor Subject) (Recipient Object) (Theme Dep/NP)). FrameNet also supports *frame-to-frame relations* – these allow frames to be inherited, to perspectivize each other, or to map to a sequence of temporally ordered subframes.

(Coyne et al., 2011a) describes an extension to FrameNet called *vignettes*. Vignettes are frames that represent the graphical realization of actions and compound objects such as locations. Vignettes can be thought of a bridge between the *high-level* semantics encoded by FrameNet and the *low-level* semantics (spatial relations and other graphical properties of objects) required to construct a 3D scene. Vignettes inherit high-level semantics from FrameNet via normal frame-to-frame inheritance and decompose into low-level graphical frames using a new SUBFRAME-PARALLEL frame-to-frame relation. Vignettes can be defined not only for actions but also for locations or any other compound objects. For example, a living room might contain a sofa, coffee table, and fireplace in a particular arrangement

There is a long history in artificial intelligence and cognitive linguistics of decomposing meaning into semantic primitives. These efforts fall into two broad classes – those focusing on primitive features of objects used to distinguish one object from another (for example in prototype theory (Rosch, 1973)) and those focused on state changes, temporal relations, and causality (Miller and Johnson-Laird, 1976). Conceptual Dependency (Schank and Abelson, 1977) is an early representational system that specifies a small number of state-change primitives into which all meaning is reduced. In lexical conceptual structure (Jackendoff, 1985), lexical relations are decomposed into a similar set of primitives. VerbNet (Kipper et al., 2000) grounds verb semantics into a small number of causal primitives representing temporal constraints tied to causality and state changes. In contrast to these causally and temporally oriented approaches, vignettes map semantics into sets of graphical constraints active at a single moment in time. This allows for and emphasizes *contextual entailment* rather than causal and temporal reasoning.

In order to apply graphical primitives to objects, it is necessary to know how objects are used and can be manipulated. The concept of *affordances* (Gibson, 1977; Norman, 1988) was introduced in the study of ergonomics and the psychological interpretation of the environment and has been examined as well from a philosophical perspective (Haugeland, 1995). Affordances are traditionally considered to be the qualities of objects in the environment that allow people or animals to interact with them. For example, the door of a refrigerator provides an affordance that allows access to the interior of the refrigerator, and the handle on a door provides an affordance that allows the door to be grasped in order to open and close it. We take a slightly broader view and include as affordances any functional or physical property of an object that allows it to participate not only in actions with people but also in relations with other objects.

One of our main tasks is to define vignettes to represent a wide variety of actions and locations. Previous work explored different methods for building location vignettes. Sproat (2001) attempted to extract associations between actions and locations from text corpora. While producing interesting associations, the extracted data was fairly noisy and required hand

editing. Furthermore, much of the information that we are looking for is common-sense knowledge that is taken for granted by human beings and is not explicitly stated in corpora. In other work Rouhizadeh et al. (2010) and Rouhizadeh et al. (2011b), Amazon Mechanical Turk (AMT) was used to collect information about locations of different objects, their parts, and surrounding objects. Various corpus association and WordNet similarity measures were applied to filter the undesirable AMT inputs.Reasonably clean data was achieved by this approach. In Rouhizadeh et al. (2011c) AMT was used for building a low-level description corpus for locations by collecting free-form text descriptions of room locations based on their pictures. The WordsEye NLP module was used to extract location elements from processed descriptions. Objects and other elements of locations were extracted in the form of RELATION–GROUND–FIGURE. Directly collecting objects of locations with AMT was investigated in Rouhizadeh et al. (2011a). AMT was then used for annotating the orientation of those objects (for more details see subsection 4.1).

This paper is structured as follows. In Section 2, we describe the graphical primitives used in vignettes as well as the set of affordances on 3D objects that enable vignettes to specify their arguments in a generic way. In Section 3, we describe how we choose and process lexical patterns to serve as input valence patterns to be used in defining action vignettes. In Section 4, we describe the annotation tools and graphical user interfaces we have developed to define location vignettes and action vignettes (with their associated valence patterns). We also describe the user interface we developed for assigning affordances and normalized part names to 3D objects. We conclude and describe future work in Section 5.

2 Vignettes and Affordances

In this section we examine, in more detail, the set of primitive graphical frames used by vignettes and the 3D object affordances used in applying those graphical primitives.

2.1 Primitive Graphical Frames

We observe that visual scenes can be decomposed into a relatively small and recurring set of primitive graphical relations. These relations represent different spatial and graphical properties such as positions, orientations, sizes, surface properties, character poses, and facial expressions. So, for example, *the man washing the floor* can be decomposed into a set of relations consisting of the man in a kneeling pose on the floor, with a bucket to his side, and holding a sponge that he applies to the floor. To capture the different manners of performing actions within the same high-level semantic frame, we define a specialized type of sub-frame called a *vignette* (Coyne et al., 2011a). Vignettes can be thought of as a bridge between *form* (the way scenes look) and *function* (what is happening or conveyed in a scene). They encode and map low-level graphical relations to the high-level semantics encoded by FrameNet. To make this decomposition, the following sets of primitive graphical frames (grouped by type) are used:

Venue and time of day: This specifies the typical time of day and the *venue*. The time of day is graphically realized by controlling the position and brightness of light sources within the scene (such as the sun position). The venue represents the local human-scale area where the action takes place and what can be seen at one time. For example, a living room or a kitchen would be a venue, but a typical house as a whole would contain many venues. The venues, themselves, are arrangements of 3D objects, and hence can be represented by location vignettes.

Holding/Touching target or patient: These specify that an agent is holding or touching an object. They vary in how the target object is situated. Arguments are provided to specify the particular body part and pose involved in holding or touching the object.

GRASP/TOUCH-ON-SELF: *button one's shirt*
GRASP/TOUCH FIXTURE: *hold a doorknob*
GRASP/TOUCH NON-FIXTURE: *hold a coffee mug*
POSITION-THEME-MANUALLY: *put a picture on a wall*

Apply Handheld Instruments: These specify a handheld instrument being applied to an object. Handheld instruments typically have affordances (see section 2.2), such as a HANDLE that allow them to be held and a INSTRUMENT-HOTSPOT, such as the blade of a knife or tip of a pencil, that is applied to a target. Undirected instruments (playing a violin) often involve specialized poses that are associated with the use of those instruments.

APPLY-INSTRUMENT-TO-HELD-PATIENT: *write on a handheld notepad*
APPLY-INSTRUMENT-USING-WORKSURFACE: *cut carrots on a table*
APPLY-INSTRUMENT-TO-TARGET: *paint the wall*
POSITION-THEME-WITH-INSTRUMENT: *roast the marshmallow*
USE-UNDIRECTED-INSTRUMENT: *play the violin. talk on the phone*

Using stationary machines/fixtures: *Fixtures* are large stationary objects. *Machines* are fixtures that can be operated to achieve some effect. Machines, such as ovens, can affect a PATIENT. The patient can rest on the machine's PATIENT-AREA affordance. Other machines, such as vending machines, can have INPUT/OUTPUT-AREA affordances. They differ from instruments in that they are not held or otherwise supported by the user. See Section 2.2 for a listing of affordances.

APPLY-MACHINE-TO-PATIENT: *boil the potatoes on the stove*
USE-FIXTURE/MACHINE: *open the door*

Looking/gesturing at target: In these frames, the agent is looking or gesturing at an object or in some direction. The particular pose or manner of gesturing can be specified.

LOOK-AT-TARGET: *glance across the room*
GESTURE-AT-TARGET: *wave at the stranger*
LOOK-AT-WITH-INSTRUMENT: *take a picture of friends*

Aiming and projectiles: In these frames, the agent is aiming or hurling an object at a remote target. A projectile can be either included or not.

HURL-PROJECTILE-AT-TARGET: *throw the stone, kick the ball*
AIM-INSTRUMENT-AT-TARGET : *shoot the rifle*
AIM-FIXTURE/MACHINE-AT-TARGET : *shoot the cannon*

Agent-in-Motion: In these frames, the agent is moving, possibly in or with a vehicle. This frame includes a source or goal location and an area or path for the motion.

SELF-MOTION: *swim across the lake*
USE-VEHICLE: *ride the horse*
PUSH-OR-PULL-FIXTURE/VEHICLE: *push the wheelbarrow*

Humans and Poses: These include facial expressions, body poses, and other body states. In the two-person interaction case, a set of specialized two-person poses can be specified.

STANCE: *stand. jump, sit, ...*
FACIAL-EXPRESSION: *happy, excited, ...*
THOUGHT-SPEECH-BUBBLE: *think about home*
WEAR: *dressed in old pair of jeans*
TWO-PERSION-INTERACTION: *John hugged Mary*

Low-Level Spatial and Graphical Primitives: This is a grab-bag of lower level primitives such spatial relations and surface properties (colors, textures, reflectivity). These are often used to specify relations and properties on objects in the vignette (often to represent the resulting state of an object from an action) rather than to directly encode the main action of the input text.

PART-OF: one entity is part of another

POSITION: *A flower in a vase.* Note that spatial relations rely on the spatial regions and affordances on the objects. (Coyne et al., 2010) Therefore the exact realization of spatial relations, like other graphical primitives, will depend on the exact set of arguments and the affordances they provide.

POSITION-BETWEEN: a figure is between a ground1 and a ground2

ORIENTATION: an entity is facing a target object or direction

SURFACE-ATTRIBUTE: the shininess, color, texture, or transparency of an entity

LIGHT-PROPERTIES: a light source color or brightness

ENTITY-SIZE-SHAPE: the size or shape of an entity

ENTITY-STATE: an entity is folded, crumpled, shattered, etc.

We note the following: 1) These graphically primitive frames themselves can sometimes be conceptually decomposed into even finer-grained graphical operations. For example, to put a character in a given pose, it is necessary to individually orient the limbs in a certain way. So it could be argued that the set of graphical relations described above are not truly primitive. Our focus, however, is to capture the cognitively salient graphical features of a scene. We therefore handle these very low-level details in our 3D graphics subsystem rather than attempting to represent them directly with vignettes. 2) If needed, a graphical frame can specify not just an entity as an argument, but also what affordance on that entity is used. For example in USE-FIXTURE/MACHINE, the TOUCH-CONTROL affordance (such as a button or switch to turn on a piece of electronics) would typically be used. If a non-default value is needed, or there is no default, it can be specified as an argument to the graphical primitive itself. 3) The graphical primitives can also include arguments for necessary supports or containers. For example, the various GRASP primitives specify not only a GRASPED-THEME argument, but also allow a CONTAINER-FOR-THEME. This allows a single primitive to handle actions like *John held the coffee* which actually involves holding a coffee mug rather than the coffee itself. We do this for convenience. In all cases, the relations between containers and supports could be specified with separate graphical primitives.

2.2 Affordances and Spatial Tags

In order to apply graphical relations to actual 3D objects we need knowledge of the structure of those objects. For example, opening a door involves grasping the doorknob, and putting a flower in a vase involves putting the stem of the flower into the container area of the vase. This knowledge of objects is captured by the notion of *affordances*. These affordances constrain how objects and human characters interact with one another in the world. Note that fairly complex poses like riding a bicycle can be accomplished by using the FOOTHOLD (pedal), GRIP-CONTROL (handlebars), and SEAT-STRADDLING (seat) affordances. We identified the following set of affordances by examining over 2000 3D objects in our library and identifying what parts of those objects would function as affordances. The interface we used for assigning these affordances to the 3D objects is described in Section 4.3.

Human Location: WALKING-AREA, PATH, WALKTHROUGH-OPENING, DOOR-GATE, VIEWING-WINDOW, VIEWING-OPENING-AFFORDANCE

Hotspot: INSTRUMENT-HOTSPOT, MACHINE-PATIENT-AREA, SOUND-SOURCE, LIGHT-SOURCE
Container and Surface: WORK-SURFACE, MACHINE-PATIENT-AREA, STORAGE-AREA, CONTAINER-BASIN, CAP-COVER, RECEPTICLE
Self-Support: SEAT-WITH-BACK, SEAT, SEAT-STRADDLING, HANDHOLD, FOOTHOLD, HANDHOLD-FOOTHOLD, LYING-SUPPORT, ARM-REST, LEG-REST, HEAD-REST
Touch-Grip: INSTRUMENT-GRIP, CARRYING-GRIP, OPEN-CLOSE-GRIP, PUSH-PULL-GRIP, TOUCH-CONTROL, GRIP-CONTROL, PEDAL, EYEPIECE, EARPIECE, NOSEPIECE, MOUTHPIECE, INSERTION-PIECE
Functional area: OUTPUT-AREA, INPUT-AREA, DISPLAY, WRITING-AREA
Spatial regions: BASE, STEM, CANOPY, TOP-SURFACE, BOTTOM-SURFACE, WALL

3 Preparation for Action Vignette Annotation

In this section we discuss how we prepared a set of approximately 1000 core verbs and their arguments to serve as input for the action vignette annotation process. These verbs were manually chosen, using subjective judgements, to include verbs that are commonly used as well as those that are concrete in nature and hence could be readily depicted. We also specified relative priorities to further guide our annotation efforts. Some of these verbs are shown in Figure 6. The list of actual verb phrases to annotate was obtained from the British National Corpus (BNC), which we parsed with the MICA parser (Bangalore et al., 2009). An AWK script was written to extract verbs from the output along with particles, direct and indirect objects, and prepositional phrases which convey salient information about the action. These verb-argument tuples provided the skeletal sentences and valence patterns representing the typical ways these verbs would be used. It is these verb-argument tuples that would then be annotated.

3.1 Parsing

MICA (Bangalore et al., 2009) is a dependency parser that uses tree insertion grammars and supertagging. A supertag is an elementary tree of a tree grammar, associated with a lexical item. The tree insertion grammar MICA is trained on was automatically extracted from a TIG-converted version of the Penn Treebank. The parser first assigns n-best supertags to an input token sequence, and then extracts from the lattice of super-tags the most likely complete tree insertion grammar derivation. Each supertag carries information about the syntactic context the lexical item may occur in and other lexical information. The Mica post-processor can use this information to augmented the resulting dependency parse with deep linguistic features, such as ubcategorization information and voice. Verbal arguments associated with each supertag can be identified as *deep syntactic* roles (subject, object, indirect object in normalized active-voice word order). The referents of empty arguments in control and raising constructions can be identified and re-attached to their verbs. This makes extraction of verb/core-argument seeds much easier and more reliable.

3.2 Extraction and Sorting

The script works in the following way: any verb that is not marked as passive voice is recorded by the script in its lemma form, which then looks for verb particles, objects, and prepositional phrases. Verb particles, irrespective of their position in the original sentence, are placed with the verb, and the verb–particle combination is treated as a distinct lexical item. The script then searches for the direct and indirect objects of the verb. Object pronouns which refer to people are normalized to *someone*, and those referring to objects to *something*; the exception is *them*, which is ambiguous in this respect. Reflexive pronouns are similarly normalized to

oneself. Articles are normalized to *a* or *an*, and possessive adjectives to *one's*. No other noun modifiers are preserved. The script then looks for prepositions which are children of the verb, and finds their objects, making the same modifications to these objects as to direct and indirect objects. The preposition and its objects are returned as a single *prepositional phrase* unit. The direct object, indirect object, and prepositional phrase are returned after the verb in the order in which they appear in the sentence. Finally, if the script detects an infinitive verb which is subject-controlled by the verb being processed, the phrase *to do something* is added to the very end of the phrase.

There were, at times, problems with the parse which impeded the function of the extractor, prompting the addition of code in the extractor work around these errors. For example, the contraction *I'm* was parsed as two lexical items by MICA: *I* and *'m*. However, *'m* was not recognized as a form of the word *am*—itself a form of the verb *to be*—but rather was analyzed as a distinct verb. Thus, the extractor was modified to detect and fix this particular case.

After a list of verb phrases was produced from the BNC, it was pared down to a more manageable size. First, the verb phrases were sorted by frequency, so that the most common phrases appeared at the top of the list. From this, we filtered out verb phrases based on whether they were part of our list of core verbs. Some examples of final extracted verb-argument tuples:
(131 (:VERB "eat") (:DIRECT-OBJECT "something")) – 131 verb phrases with *eat* and a pronoun.
(24 (:VERB "eat") (:DIRECT-OBJECT "a meal")) – 24 verb phrases with *eat* and *meal*.

4 Annotation Methods and Tools

In this section we describe some of the different methodologies and user interfaces we have developed for defining vignettes and assigning affordances and normalized part-names to 3D objects.

4.1 Using AMT to build location vignettes

We have been investigating the use of Amazon Mechanical Turk (AMT) for building locations vignettes. The inputs to our AMT tasks are 'typical' photos of different rooms, that show large objects typical of that particular room. We carefully selected the picture from the results of image searches using Google and Bing. Turkers of each task had to be in the US and had previous approval rating of at least 99%. Restricting the location of the Turkers increases the chance that they are native speakers of English, or at least have good command of the language.

Phase 1: Collecting the functionally and visually important objects of rooms

The functionally important objects for a room are those that are required in order for the room to be recognized or to function properly. The visually important objects are those that help define the basic structural makeup of that particular room instance, such as large objects and those that are fixed in location. Examples of those objects in a kitchen can be "stove", "oven", "sink", "cabinets", and so on. After collecting the objects from several AMT tasks, we post-process them with the following steps (Rouhizadeh et al., 2011a):

1. Manual checking of spelling and converting plural nouns to singular.
2. Removing conjunctions such as "and","or", and "/".
3. Substituting the objects belonging to the same WordNet synset with the most frequent word of the synset. ("tub", "bath", and "bathtub" ⇒ "bathtub")

4. Substituting words with major substrings in common (“night stand", “night-stand" ⇒ “nightstand").
5. Selecting the head nouns of compounds (“computer monitor" ⇒ “monitor").

Phase 2: Collecting the visual properties of the rooms

Turkers should determine the room layout (diagonal or horizontal), room size (small, medium, or large), ceiling height, wall texture (painted color, wallpaper pattern, fabric, wood paneling, tile, concrete, or stone), and floor texture (tile, wood, carpeted, stone, or concrete).

Phase 3: Collecting the spatial relations between the objects

For each object **O** that is collected in phase 1 Turkers should answer the following questions: (see Figure 2)

1. Is **O** located against a wall? If so, determine the wall.
2. Is **O** near another object? If so, determine the object, determine the direction (front, back, or side), and the distance (1 ft, 2 ft, 3 ft, or 4 ft or more).
3. Is **O** supported by (i.e. on, part of, or attached-to) another object? If so, determine the object.
4. Is **O** facing another object? (e.g. chair facing a table) If so, determine the object.

We have completed phases 1 and 2 for 85 rooms and are now performing phase 3 for those rooms. To evaluate the results of phase 1, we compare the objects we collect to a gold-standard set of objects that are found in five rooms compiled by an expert. 91% of our collected objects were correct (precision) and we could gather 88% of the objects that we expected (recall).

4.2 Defining Location Vignettes using a Text-to-Scene System

We are also using WordsEye itself to define location vignettes for rooms (see Figures 3, 4). Annotators use WordsEye to textually describe rooms, and in the process see what those rooms look like. Those textual descriptions correspond to the graphical relations needed to define vignettes. We have currently defined about 50 fairly detailed rooms of different types using this method. The main advantage over the menu driven approach described in 4.1 is that the annotator can much more quickly and easily describe simple spatial relations (e.g. *the bed is against the wall and 3 feet to the right of the dresser*) than finding and filling in parameters for the objects, the relations, and their arguments on a complex set of menus. In addition, this method also has advantages over “working blind” with either menus or the purely textual descriptions described in Rouhizadeh et al. (2011c). The visual feedback of seeing the location of the room as it is described lets the annotator gauge how their specifications will actually be interpreted and depicted. In addition, the annotator can work incrementally and base additional input on an actual rendered scene rather than relying on a fleeting mental image or an interpretation of existing text or menu specifications. Furthermore, this makes it easier for annotators to use already-defined locations as a textual and visual starting point for additional variations.

Using the text-to-scene system to depict locations as they are described also serves to ensure that the inputs are well formed. All inputs are parsed and converted to a semantic representation consisting of objects and graphical primitives. As a result the input text is automatically converted into the graphical relations used by the vignette being defined. No post-processing is required. This applies not only to the relations but also to the specific object types that are used in the location. For example, if the user specifies that a chair is in a kitchen, they can pick the specific type of chair and avoid inappropriate chairs like lounge chairs and electric chairs.

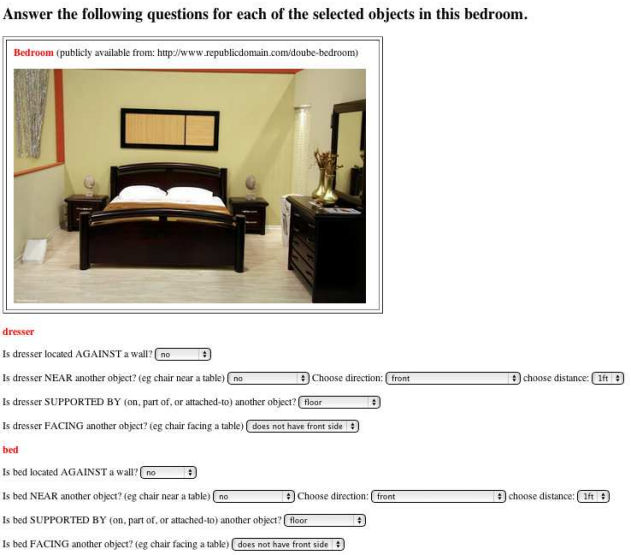


Figure 2: Collecting spatial relations between objects in Phase 3 of the AMT task

Figure 3: Kitchen location vignette defined with WordsEye

Figure 4: Other location vignettes defined with WordsEye

4.3 Assigning Part Names and Affordances

WordsEye utilizes a library of 2200 common 3D objects. The parts of these objects are often named (by the 3D artist who created them) with some abbreviated form of a normal English word. For example, the left and right tusks on one of the 3D models for an elephant are named *ltusk* and *rtusk*. In addition to normalizing part names, we need to specify which parts function as affordances. We developed a web-based user interface to assign both a normalized part name and any corresponding affordances for every part (see Figure 5).

It is important to note that we are defining parts and affordances on actual 3D objects rather than on conceptual types in the ontology. This eliminates a typical problem in ontologies where higher-level concepts define properties that may not apply to all lower level concepts. For example, in WordNet (Fellbaum, 1998), the synset *shoe* has a meronym (part designation) of *shoelace*. Not all types of shoes, however, have shoelaces. In particular, the synset *loafer* is a hyponym of *shoe*, but loafers don't have shoelaces. Since our taxonomy contains actual 3D objects, we can instead assign the exact set of parts, affordances, and other properties that apply to those specific objects. We can then infer by induction that most shoes have laces, but not all.

We have successfully annotated our entire 3D library in this manner, in the process renaming about 18,000 parts and assigning 2,400 affordances. Since our interface lets the annotator assign part names by typing in a word, there were cases of unknown or misspelled words. There were also cases of ambiguity, with multiple word senses for the same input part name. These were fixed in a post-process by automatically finding all problematic cases (those with either no known word sense or more than one) and manually assigning the correct sense.

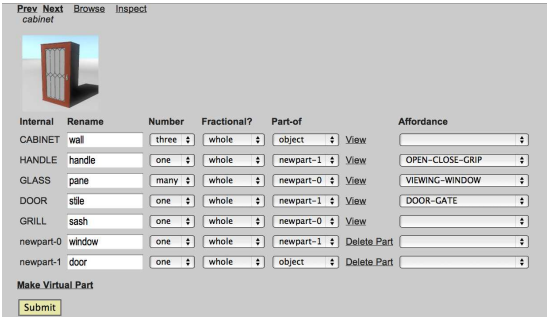


Figure 5: Parts and affordances for a door 3D object

4.4 Defining Action Vignettes

Action vignettes are specialized FrameNet frames that represent different ways of decomposing an action into graphical primitive frames. Like other frames, they can be evoked by a lexical item in its syntactic context. In order to simplify the annotation process, we are currently mapping vignettes directly to valence patterns, bypassing any corresponding FrameNet frames. We are decoupling the vignette definition process from FrameNet for a couple reasons. First, many common verbs (such as *bounce*) have no defined lexical items within FrameNet. About

17% of our core verbs have no FrameNet equivalent. Secondly, the mapping from frames to vignettes would complicate and slow down the annotation process, forcing the annotator to understand much of FrameNet in addition to our graphical primitives. These problems will be addressed in future work (see Section 5).

Annotating action vignettes involves mapping each verb-argument tuple to sets of primitive graphical frames representing how that verb (with arguments) would be depicted. See Figure 7. These decomposed tuples implicitly define new vignettes, where the tuples correspond to valence patterns that specify when that vignette is invoked. Since many tuples can be depicted in the same basic manner (albeit with different input arguments), we allow inheritance between vignettes. Inherited vignettes decompose to the same set of graphical primitives as their parent, but can override the values of the arguments. For example, *wash an apple* and *wash a pear* would invoke the same vignette, which we can think of as WASH-FRUIT-IN-SINK. The user interface supports this functionality by allowing the annotator to select and assign multiple input tuples to the same vignette.

We have so far annotated about 90 core verbs using this interface. In the process, we have specified 450 top-level vignettes and 2500 inherited vignettes (some of which override the inherited values). We have found that the original set of graphical primitives (Section 2.1) has remained fairly stable during this process, with an occasional new parameter or value type being added as new cases were encountered. We have made changes to the user interface (for example, adding different sorting keys and viewing options) based on experience using it. See Figure 8.

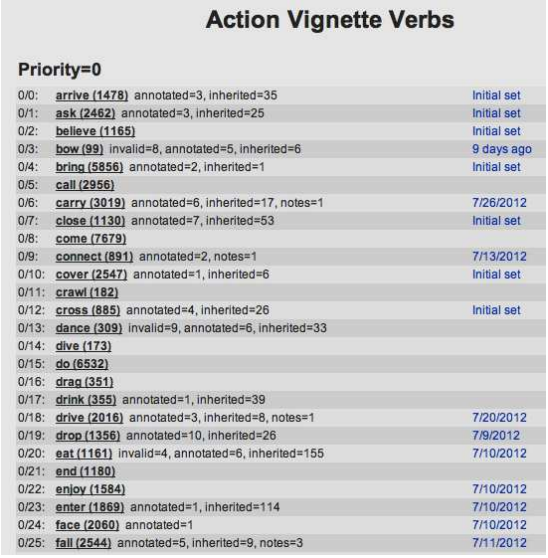


Figure 6: Action Vignette Browser for selecting verbs and showing annotation status

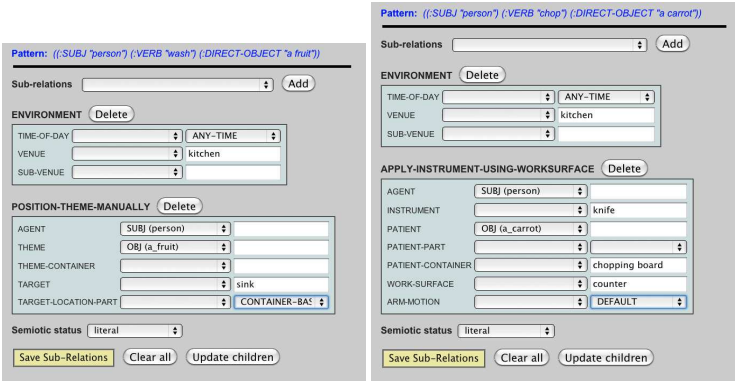


Figure 7: Action Vignette Editor for *wash fruit* and *chop carrot*

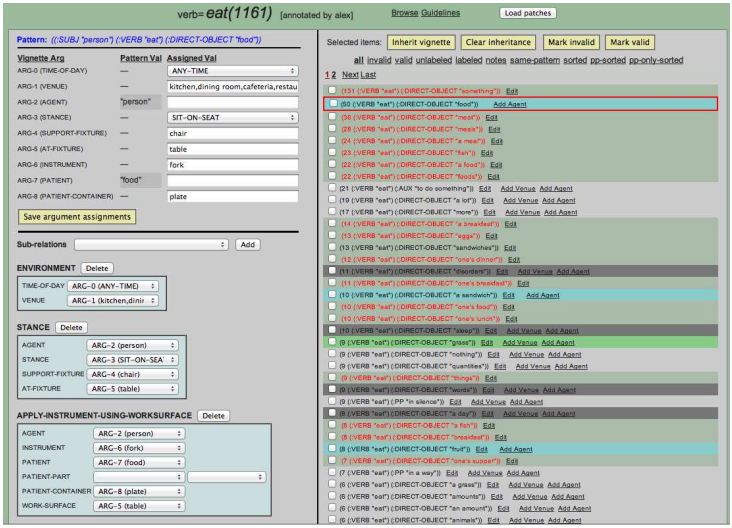


Figure 8: Action Vignette Editor. Left side shows vignette for the current input pattern. Top-left shows vignette arguments and bottom-left shows graphical decomposition applied to those arguments. Right side shows input tuples. Color coding is used on tuples to denote inheritance relations and annotation status. Different filtering and sorting options allow the annotator to focus on particular valence patterns.

5 Conclusion and Future Work

In this paper we have presented approaches to acquiring real-world knowledge to be used in a text-to-scene system. The core of our approach is embodied in the notion of vignettes. Vignettes are frames that are decomposable into grounded graphical relations. To implement vignettes we haved defined a specific set of graphical primitives that allow us to map between high-level to low-level semantics. These graphical primitives enable a wide variety of scenes to specified with a sufficient level of detail. In order to apply these graphical primitives to 3D objects we have defined a set of affordances representing the structure of those objects and how they are used and manipulated. Finally, building on this framework of vignettes, graphical primitives, and affordances, we have developed several methods for populating our resource with both locations and action vignettes. For action vignettes, this involved preparing a corpus of verb-argument tuples to be used as input data and developing tools to annotate that data with vignettes. To define location vignettes we used both AMT crowdsourcing methods and WordsEye, our text-to-scene system, as an annotation tool itself. Our resulting resource (called VigNet) is a lexically-oriented knowledge-base with lexical inputs mapping to vignettes, which in turn map to graphical primitives and actual 3D objects. VigNet will be made publicly available at `http://www.cs.columbia.edu/speech/text2scene`.

Future work includes finishing the vignette annotation process. In addition, action vignettes, as defined in the user interface, will require some amount of post-processing in order to be used. In particular, we need to handle unknown words and word sense ambiguity issues for those words that were typed in by the annotator. Those will be normalized and disambiguated in a separate semi-automatic pass as we did for part names (Section 4.3). We will also perform a separate post-processing task to link vignettes to their corresponding FrameNet frames.

One main challenge in using vignettes is that there won't be a vignette to match every possible input. Instead, it will be necessary to generalize the input arguments to find the closest vignette. For example, a vignette defined for *wash an apple* (using a sink) can be applied to washing any small round fruit. This is partially addressed by annotators listing multiple possible values to fill arguments when defining the vignette. In general, however, finding the closest vignette will involve estimating the semantic distance between the arguments of the candidate vignettes and those of the input sentence. To do this, we will leverage information in our ontology that specifies the sizes, shapes, substances, and other semantic properties of all 3D objects and their parents. We also note that vignettes, themselves, can explicitly specify arbitrarily complex sets of constraints and restrictions on their arguments to help make these matches.

A second major challege in using vignettes is to compose actions vignettes with location vignettes in the course of text-to-scene generation. We intend to accomplish this as follows. If an action is mentioned, then the default venue for that action will evoke a set of possible location vignettes. Any affordances required by the action will be unified with those provided by the location. For example, the vignette for *chop carrots*, might supply a default VENUE of a kitchen and APPLY-INSTRUMENT-USING-WORKSURFACE. The kitchen vignette includes a counter and kitchen table, both of which provide a WORK-SURFACE affordance.

Acknowledgments

This work was supported in part by NSF IIS- 0904361. Any opinions, findings and conclusions or recommendations expressed in this material are the authors' and do not necessarily reflect those of the sponsors.

References

Bangalore, S., Boullier, P., Nasr, A., Rambow, O., and Sagot, B. (2009). Mica: a probabilistic dependency parser based on tree insertion grammars application note. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics, Companion Volume: Short Papers*, pages 185–188. Association for Computational Linguistics.

Coyne, B., Bauer, D., and Rambow, O. (2011a). VigNet: Grounding Language in Graphics using Frame Semantics. In *Workshop on Relational Models of Semantics (RELMS) at ACL*.

Coyne, B., Schudel, C., Bitz, M., and Hirschberg, J. (2011b). Evaluating a Text-to-Scene Generation System as an Aid to Literacy. In *Workshop on Speech and Language Technology in Education (SlaTE) at Interspeech 2011*.

Coyne, B. and Sproat, R. (2001). Wordseye: An automatic text-to-scene conversion system. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, pages 487– 496, Los Angeles, CA, USA.

Coyne, B., Sproat, R., and Hirschberg, J. (2010). Spatial relations in text-to-scene conversion. In *Computational Models of Spatial Language Interpretation, Workshop at Spatial Cognition 2010*, pages 9–16, Mt. Hood, OR, USA.

Fellbaum, C. (1998). *WordNet: an electronic lexical database*. MIT Press.

Gibson, J. (1977). The Theory of Affordances. In Shaw, R. and Bransford, J., editor, *Perceiving, acting, and knowing: Toward an ecological psychology*, pages 67–82. Erlbaum, Hillsdale, NJ.

Haugeland, J. (1995). Mind embodied and embedded. In Houng, Y. and Ho, J., editors, *Mind and Cognition*. Academia Sinica, Taipei.

Jackendoff, R. (1985). *Semantics and cognition*, volume 8. The MIT Press.

Kipper, K., Dang, H. T., and Palmer, M. (2000). Class-Based Construction of a Verb Lexicon. In *Proceedings of AAAI 2000*, Austin, TX.

Miller, G. and Johnson-Laird, P. (1976). *Language and perception.* Belknap Press.

Norman, D. A. (1988). *The Psychology of Everyday Things*. Basic Books.

Rosch, E. (1973). Natural categories. *Cognitive psychology*, 4(3):328–350.

Rouhizadeh, M., Bauer, D., Coyne, B., Rambow, O., and Sproat, R. (2011a). Collecting spatial information for locations in a text-to-scene conversion system. In *Proceedings of the Workshop on Computational Models of Spatial Language Interpretation and Generation (CoSLI 2011)*, Boston, MA.

Rouhizadeh, M., Bowler, M., Sproat, R., and Coyne, B. (2010). Data collection and normalization for building the scenario-based lexical knowledge resource of a text-to-scene conversion system. In *Proceedings of SMAP 2010: 5th International Workshop on Semantic Media Adaptation and Personalization*, pages 25 –30, Limassol, Cyprus.

Rouhizadeh, M., Bowler, M., Sproat, R., and Coyne, B. (2011b). Collecting semantic data by Mechanical Turk for the lexical knowledge resource of a text-to-picture generating system. In Bos, J. and Pulman, S., editors, *Proceedings of the Ninth International Conference on Computational Semantics (IWCS 2011)*, The University of Oxford.

Rouhizadeh, M., Coyne, B., and Sproat, R. (2011c). Collecting semantic information for locations in the scenario-based lexical knowledge resource of a text-to-scene conversion system. In König, A., Dengel, A., Hinkelmann, K., Kise, K., Howlett, R., and Jain, L., editors, *Knowledge-Based and Intelligent Information and Engineering Systems*, volume 6884 of *Lecture Notes in Computer Science*, pages 378–387. Springer Berlin / Heidelberg.

Ruppenhofer, J., Ellsworth, M., Petruck, M., Johnson, C. R., and Scheffczyk, J. (2010). *Framenet II: Extended Theory and Practice*. ICSI Berkeley.

Schank, R. C. and Abelson, R. (1977). *Scripts, Plans, Goals, and Understanding*. Earlbaum, Hillsdale, NJ.

Sproat, R. (2001). Inferring the environment in a text-to-scene conversion system. In *Proceedings of The First International Conference on Knowledge Capture*, pages 147–154, Victoria, BC, Canada.

Towards efficient HPSG generation for German, a non-configurational language

Berthold CRYSMANN[1] *Woodley PACKARD*[2]
(1) LLF (UMR 7110), CNRS & U Paris–Diderot, 5 rue Thomas Mann, F-75205 Paris Cedex 13
(2) University of Washington, Seattle, WA, USA
crysmann@linguist.jussieu.fr,sweaglesw@sweaglesw.org

ABSTRACT
In this paper, we propose a rule-based method to improve efficiency in bottom-up chart generation with GG, an open-source reversible large-scale HPSG for German. Following an in-depth analysis of efficiency problems in the baseline system, we show that costly combinatorial explosion in brute force bottom-up search can be largely avoided using information already contained implicitly in the input semantics: either (i) information is globally present, but needs to be made locally available to a particular elementary predication, or (ii) semantic configurations in the input have a clear translation to syntactic constraints, provided some knowledge of the grammar. We propose several performance features targeting inflection and extraction, as well as more language-specific features, relating to verb movement and discontinuous complex predicates. In a series of experiments on three different test suites we show that 7 out of 8 features are consistently effective in reducing generation times, both in isolation and in combination. Combining all efficiency measures, we observe a speedup factor of 4.5 for our less complex test suites, increasing to almost 28 for the more complex one: the fact that performance benefits drastically increase with input length suggests that our method scales up well in the sense that it effectively heads off the problem with exponential growth. The present approach of using a generator-internal transfer grammar has the added advantage that it locates performance-related issues close to the grammar, thereby keeping the external semantic interface as general as possible.

TITLE AND ABSTRACT IN GERMAN

Effiziente HPSG-Generierung für das Deutsche

Wir stellen eine regelbasierte Methode vor, zur automatischen Anreicherung der semantischen Eingabe einer reversiblen HPSG des Deutschen, die es erlaubt, teure uninformierte Suche bei der Bottom-Up-Chart-Generierung weitgehend zu vermeiden, indem (i) globale Information, die implizit in der Eingabe vorhanden ist, explizit und lokal verfügbar gemacht wird, und (ii) syntaktische Constraints aus semantischen Konfigurationen abgeleitet werden. Wir schlagen Performanzfeatures für verschiedene Phänomene vor, wie Flexion, Extraktion, Verbbewegung und diskontuierliche komplexe Prädikate. Unsere Experimente zeigen erhebliche Effizienzsteigerungen (Faktor 4.5–Faktor 27.8), deren Zunahme mit steigender Eingabekomplexität korreliert, was die gute Skalierbarkeit unserer Methode belegt. Der generator-interne Transferansatz zeichnet sich weiterhin dadurch aus, daß Performanzaspekte grammatik-nah behandelt werden, wodurch die externe Semantikschnittstelle so allgemein wie möglich bleibt.

KEYWORDS: Surface generation, HPSG, German.

Proceedings of COLING 2012: Technical Papers, pages 695–710,
COLING 2012, Mumbai, December 2012.

1 Introduction

1.1 HPSG bottom-up generation and non-configurationality

Recent advances in the efficiency of bottom-up chart generation with reversible HPSG grammars (Carroll and Oepen, 2005), namely local ambiguity factoring under subsumption and index accessibility filtering, appear to have solved the most pressing efficiency problems associated with HPSG generation for English, turning reversible linguistically motivated grammars like the ERG (Copestake and Flickinger, 2000) into interesting resources for offline and online surface generation. While the efficiency measures implemented in the LKB and ACE generators (see section 1.2) are also effective for German, these measures appear to be insufficient to resolve generation performance issues for GG, a large-scale HPSG for German originally developed at DFKI (Müller and Kasper, 2000; Crysmann, 2003, 2005, 2007). In fact, even on moderately complex inputs, the generator quickly runs into a combinatorial explosion, having so far prevented the grammar from being usable for any serious real-time NLG tasks.

Upon closer inspection of the source of the inefficiency, it became quickly apparent that the observed performance problems are the result of a conspiracy of several factors, most of which can be subsumed under the notion of non-configurationality:

- Relatively free constituent order

  In contrast to English, constituent order in German clauses is relatively free, permitting permutation of complements, including the subject, as well as interspersal of modifiers in pretty much any position. As a result, chart size grows rather quickly, with ambiguity packing being ineffective until rather large, i.e. mostly clausal, structures are built.

- Verb placement

  German finite verbs display a placement alternation between clause-initial (V1/V2) and clause-final realisation, determined by clausal construction type. Under a bottom-up regime, both left-branching and right-branching structures must be explored. Furthermore, PPs and sentential complements easily extrapose across final verbs, thereby increasing the search space even more. In the case of particle verbs, initial placement of the verb leaves the particle in final position, giving rise to discontinuous lexical items, related by (simulated) head movement (Kiss and Wesche, 1991; Müller and Kasper, 2000; Crysmann, 2003, among others).

- Argument composition

  Auxiliaries, modals, raising verbs, and, optionally, control verbs form a verb cluster with their non-finite complements. Arguments of upstairs (=governing) and downstairs (=governed) verbs can be interleaved (e.g. *... weil ein Buch*$_2$ *er*$_1$ *ihm*$_2$ *zu kaufen*$_2$ *versprach*$_1$ 'because he$_1$ promised$_1$ to buy him$_2$ a book$_2$.'), making it necessary to compose arguments of the downstairs verb (*kaufen* 'buy') onto the valence lists of the upstairs verb (*versprechen* 'promise'), resulting in the creation of complex predicates.

  Argument composition interacts with both free constituent order and verb placement. In particular the latter means that some members of the composed valence list must be hypothesised before the initial verb has been encountered, leading to partially underspecified valence lists (e.g. *Letzte Woche versprach*$_1$ *ein Buch*$_2$ *er*$_1$ *ihm*$_2$ *zu kaufen*$_2$ 'Last week, he$_1$ promised$_1$ to buy him$_2$ a book$_2$.'). Since the underspecified valencies are not constrained as to the identity of the argument (no semantic Skolem constants), any chart

item (e.g., *letzte Woche* 'last week') that matches the underspecified syntactic description can sneak in (i.e., locally satisfy the hypothesised valency), thereby creating massive local ambiguity.

- Partial VP fronting

  Verb fronting in German may leave some (or all) arguments behind for realisation in the *Mittelfeld* (e.g. *Kaufen*$_2$ *[soll*$_1$ *er*$_{1,2}$ *ihm*$_2$ *das Buch*$_2$ *morgen]. / Das Buch*$_2$ *kaufen*$_2$ *[soll*$_1$ *er*$_{1,2}$ *ihm*$_2$ *morgen]. / Ihm*$_2$ *das Buch*$_2$ *kaufen*$_2$ *[soll*$_1$ *er*$_{1,2}$ *morgen].* 'He$_{1,2}$ should$_1$ buy$_2$ him$_2$ the book$_2$ tomorrow.'). Since the core sentence has to be generated before it combines with the fronted element, construction of the core sentence needs to proceed without any access to valence information. As a result we experience a massive combinatorial explosion that can only be controlled very late, i.e. once the entire sentential structure has been built.

- Rich inflection

  While not a problem in itself, the fact that German NPs are inflected for case multiplies the existing performance issues, most specifically in the case of underspecified valence information, since irrelevant case inflection can only be detected quite late.

In the present paper we suggest a method that automatically enriches the input semantics in such a way, as to derive local syntacto-semantic constraints from the global semantic configuration: as a consequence we shall be able to eliminate globally unsuccessful generator hypotheses early on in bottom-up chart generation.

```
[ LTOP: h0
  INDEX: e19
  RELS: < [ prpstn_m_rel LBL: h0 MARG: h16  ARG0: e19 TPC: x17  PSV: u2 ]
   [ "_pron_n_ppro_rel" LBL: h3
     ARG0: x17 [ --TOP: + --COH: + --PUNCT: prop-punct --CAS: n-list PNG.PN: 2s ] ]
   [ "pronoun_q_rel" LBL: h7 ARG0: x17 RSTR: h9  body: h10  ]
   [ "_sollen_v_modal-haben_rel" LBL: h18
     ARG0: e19 [ --TOP: - --COH: - --SIND: 2s --PUNCT: prop-punct
                 --TPC: tpc-non-event-non-mod --SUB: -
                 TENSE: present MOOD: indicative PERFECTIVE: - ]
     ARG1: h14 ]
   [ "_schnarchen_v_n-haben_rel" LBL: h12
     ARG0: e15 [ --TOP: - --COH: - --PUNCT: prop-punct --TPC: tpc-non-event-non-mod
                 --SUB: bool TENSE: untensed PERFECTIVE: - ]
     ARG1: x17 ] >
  HCONS: < h18 qeq h16 h9 qeq h3 h14 qeq h12 > ]
```

Figure 1: Enriched input MRS for *Du sollst schnarchen* 'You should snore'

1.2 The ACE generator

The open source ACE platform (http://sweaglesw.org/linguistics/ace/) implements a natural language generator based primarily on chart generation (Kay, 1996). The input to the generation system is a grammar and the semantics of the utterance to be generated (expressed in Minimal Recursion Semantics, or MRS (Copestake et al., 2005)). The grammar is primarily a declarative formalism, in that it defines a bidirectional relationship between MRSes and strings. The generator's output is the list of all strings which are related to the input MRS by the grammar. To combat the exponential worst-case complexity of the chart generation algorithm,

ACE deploys two key efficiency measures described by (Carroll and Oepen, 2005), namely ambiguity packing under subsumption and index accessibility filtering. In these respects it is quite similar to the LKB parser-generator system (Copestake, 2002). While ACE, just like the LKB, supports not only parsing and generation modes, but also MRS-based transfer, its main advantage resides in its processing efficiency: compared to the LKB, generation speed on the LOGON Rondane treebank (1169 items, avg. sentence length: 14.13) is 14.7 times better than that of the LKB, bringing average generation times down from 6.34s (LKB) to 0.43 (ACE).

2 MRS term rewriting for generation efficiency

The central mechanism by which we intend to address the generation efficiency problem is to automatically enrich the input semantics to the generator in such a way that global information implicitly present in the MRS representation will be made explicit and locally available on relevant elementary predications. By doing so, we will make them ultimately accessible to the grammar during generation. By means of an automated and quasi-deterministic rewrite step on the input MRS, we hope to strike a good balance between a maximally grammar-independent external semantic interface, suitable for application developers, and an enriched input to the generator that will hopefully reduce brute-force search by means of automatically derived syntacto-semantic constraints.

2.1 The LOGON MRS term rewrite system

Within the context of the LOGON MT project (Oepen et al., 2004, 2007), the LKB processing and development platform was extended with a term rewrite system for semantic representation using Minimal Recursion Semantics (Copestake et al., 2005).

In the LOGON system, a *transfer grammar* is a sequential, resource-sensitive set of rewrite rules which, when applied one after another in order, transform an MRS produced by a source-language grammar into an MRS suitable for NLG with the target-language grammar. We adopt the same formalism for a different purpose. A rewrite rule is a tuple of patterns for matching pieces of MRSes, consisting of an *input* pattern, a *context* pattern, a *filter* pattern and an *output* pattern. A rule $< I, C, O, F >$ causes a part of the current MRS matching the input pattern I to be replaced with the output O, provided that the context C also matches the input MRS and the filter F does *not* match. The patterns can be interdependent, so that e.g. the output can copy information matched by the input and the context. Each of the four patterns I, C, O, F contains any number of descriptions of elementary predications[1].

In an MRS (cf. figure 1), an elementary predication consists of a predicate name and any number of named roles, whose values are logical variables. A description of an elementary predication can use a regular expression to constrain which predications can match it. Several sorts of type constraints can be imposed on the values of individual roles, including unifiability with, subsumption by, or equality to a particular type (*x,e,u,i,h*). As for *properties*, i.e. features, of such variables, matching is restricted to mere unifiability. Finally, it is possible to specify that two or more variables matched in different elementary predication descriptions within the same rule have the same identity, by assigning a coreference tag. Similarly, for variable properties.

Formally, the rewrite system has the computational power of a Turing machine. As such, it is not possible to give bounds on the time complexity of applying an arbitrary rule set. However, in practice the operation is tractable for the types of rule sets considered here.

[1] Descriptions involving handle constraints are also possible, though less common.

2.2 Implementation of term rewrite system in ACE

The ACE platform, similarly to the LKB, allows grammarians to interpose a step of term rewriting between the declarative portion of the grammar and the publicly displayed MRS. The purpose of this is to allow grammarians additional freedom in designing the MRS schema described by the grammar proper, while maintaining a semantic interface that is more stable between grammar revisions and also affording an opportunity to remove remnants of non-semantic information. Since the term rewrite system is not bidirectional, separate rule sets are used after parsing and before generation.

The external MRS input to the generator is passed through the *pre-generation* rewrite system, resulting in a so-called internal MRS input (cf. figure 1). It is this MRS that is used to identify the initial set of grammar entities that need to be added to the chart. The immediate result of chart generation is a set of strings together with the sequence of grammar rules and lexemes that licensed them, and the MRS corresponding to that analysis. The result MRSes are passed through the *post parsing* rewrite rules, resulting in external MRSes. Only those strings whose corresponding external MRSes are subsumed[2] by the external MRS input are output.

A single rule can match an MRS multiple ways. Due to resource sensitivity, the order in which the matches have the rule applied can in principal affect the outcome. When a rule matches the input in K different places, there are $K!$ possible match orderings to try, which could each yield a different result, making the complexity of the operation worse than exponential. In practice, this issue can be so severe that the time spent in the rewriting process dominates the overall generation time. However, through careful rule-writing it is possible to ensure commutativity. ACE features a mode in which only one (arbitrary) match ordering is performed, rather than executing all $K!$ orderings (only to determine that they all have identical results). We exploit this feature to reduce the time spent in rewriting to a fraction of the overall generation time.[3]

2.3 Rule-based enrichment of input semantics

In this subsection, we shall present in some detail the individual efficiency measures our transfer grammar automatically derives from the generic external semantic representation. The efficiency measures we implemented can be largely classified into three groups: inflection-related measures, which mainly reduce the number of inflectional variants in the chart, German-specific measures related to verb placement and argument composition, and finally, extraction-related measures.

Most of the enrichment was done by means of having the transfer grammar augment semantic variables with additional performance features. Values of these features are typically atomic types (cf. figure 1 for a sample MRS: performance features are prefixed with two dashes and rendered in blue). In one case, i.e. *oind*, the transfer grammar adds an additional role argument (individual variable) to relevant elementary predications. On the grammar side, rules were additionally constrained according to these efficiency features. Unless specialised to some value by the MRS rewrite grammar, the enriched grammar rules will apply just as before, enabling us to measure performance gains by simply activating or deactivating blocks of transfer rules.

[2] By subsumption of MRSes, it is meant that every predicate in the input MRS must be realised in the output MRS, and the identity of the logical variables is the same (modulo renaming). It is considered permissible for the output MRS to be more specific than the input MRS, permitting, inter alia paraphrase generation by input underspecification.

[3] Average transfer processing times on the three test suites discussed in this paper are as follows: MRS: 13.2ms, TSNLP: 12.1ms, Babel: 29.4ms. Transfer times are already included in the overall processing time reported in table 1 below.

In this subsection, we shall discuss each measure in turn, together with brief remarks on the implementation and an estimate of the expected benefits.

2.3.1 Inflection

Case (*cas*) One of the most straightforward efficiency measures to come up with when confronted with a highly inflectional language such as German is to eliminate inflected forms from the chart that cannot possibly be part of a globally well-formed realisation. While some inflectional forms are readily filtered by the semantic input or the lexicon, namely predicate-inherent information such as TAM (tense/aspect/mood) and number/gender for nominal expressions, this is not the case for morphosyntactic case, which is determined by properties of the governing predicate.

In a configurational language, the expected inefficiency of inflecting every NP for all possible cases, even irrelevant cases may be suboptimal, but not really a matter for concern, since the NP will locally combine with its governing predicate, rendering NPs in irrelevant cases inert during further search. In a non-configurational language such as German, which features argument composition, heads combine with complements that are not their own arguments but rather those of a predicate they compose with, i.e., they need to cater for unknown raised arguments by means of underspecified valence lists. As a result, the identity of the inherited arguments is not known, so any XP present in the chart, however inflected, can sneak into these underspecified lists, to be ruled out, in the majority of cases, only when a significantly larger structure has been built. Unfortunately, languages with an articulate case system tend to be of the non-configurational, rather than the configurational type.

In order to predict the case for NPs, we developed a set of 35 rules that derive case requirements from lexical and structural properties of the input semantics (cf. the `--CAS` feature in figure 1). While some cases are indeed trivial, e.g., predicting the case of obliques, or arguments of prepositions, others are not: first, since individuals can be arguments to more than one predicate, as witnessed in relative clause constructions, such individual variables must be exempted from case prediction. Second, raising and, in particular, voice alternation can change case assignment properties. Thus, the transfer grammar must carefully anticipate these properties in a case by case fashion.

Apart from an overall slight reduction in chart size, we expect this feature to be particularly useful in all constructions involving locally underspecified valence lists, including the quite common case of separable particle verbs and raising and control constructions, as well as more specific, yet quite expensive ones like partial VP fronting.

Punctuation (*punct*) The implementation of punctuation in GG (Kilian, 2007) follows that of the English ERG in using inflectional rules. Even when limiting ourselves to basic sentence punctuation (commas, period, question mark, exclamation mark), almost every chart item can be inflected in 5 different ways, given that it cannot be known a priori which chart item will end up, e.g., at the right periphery of the entire sentence, where sentence mode (declarative/interrogative/imperative) is expressed. What is globally known, however, is sentence mode: all it takes is to distribute exactly this information onto every elementary predication (cf. `--PUNCT` in figure 1). Abstracting away from quotations, every sentence will be in only one of the three modes, so the number of punctuation variants for each chart item can be brought down to 3 (instead of 5).

This measure, while simple-minded and straightforwardly implemented (5 rules), is nevertheless

expected to be highly efficient, given that it indiscriminately targets almost every lexical edge (heads and dependents alike) and therefore has a significant impact on the overall size of the search space, given the bottom-up regime of the generator. The only edges that do not benefit from this (or any other measure discussed in this paper) are those corresponding to semantically empty lexical items, like, e.g., auxiliaries, relative pronouns etc.

2.3.2 Verb movement and direction of branching (sub)

A peculiarity of German syntax that has quite strong repercussions on processing efficiency is verb placement: while non-finite verbs are placed in phrase-final position, finite verbs display an alternation between final and initial position: in relative clause, embedded interrogatives, as well as subordinate clauses introduced by a complementiser or subordinating conjunction, the finite verb is realised in final position, otherwise initially, including matrix clauses.

The global construction type ("matrix order" vs. "subordinate order") can be calculated from properties of the input semantics, in particular, by taking into consideration sentence mode (declarative vs. interrogative), the kind of embedding (relativisation, complementation, type of conjunction), as well as the presence and nature of the topicalised element (embedded V2 vs. embedded wh vs. that-clause). The pre-generation transfer grammar uses this information to determine for each verb whether it is in a "subordinate" or "non-subordinate" context (cf. `--SUB` in figure 1). In addition to predicting direction of branching for simple verbs, the main benefit of this feature is that we can decide when to hypothesise head movement of the verb.

2.3.3 Discontinuous complex predicates

Coherent vs. non-coherent constructions (*coh*) Probably one of the strongest factors responsible for generation inefficiency is due to discontinuous complex predicates leading to local underspecification of valence lists that permit sneaking in from any XP edge in the chart, triggering massive combinatorial explosion. Fortunately, whether some predicate permits argument composition or not is a lexical matter, with composition being restricted to auxiliaries, modals, raising and control verbs. Thus, the transfer grammar marks the arguments of non-finite complements of modals etc., as to their potential of undergoing argument composition (*coh +*). Likewise, arguments of finite verbs that are expressed periphrastically (perfective, future, passives) are marked for composition. With arguments of all other predicates being marked with a negative value, underspecified valence lists can be protected to some degree against illicit intrusion of arguments (cf. `--COH` in figure 1). This feature is expected to be particularly helpful in those cases where we are confronted with entirely underspecified valence lists, as with separable particle verbs and partial VP fronting.

Predicting upstairs objects (raising/control) As discussed in the introduction, discontinuous verb clusters may necessitate hypothesising valencies of the initial verbs to be realised in the *Mittelfeld*, in particular objects of initial raising and control verbs that intersperse with the arguments of the final verb or verb cluster. Since the subcategorisation requirement of the upstairs verb are not known during bottom-up construction of the *Mittelfeld*, additional arguments are hypothesised even in cases where the initial verb takes no complement at all. Furthermore, such hypothesised argument slots provide potential for illicit intrusion.

On the basis of the predicate argument structure, however, it is quite straightforward to decide not only whether an argument should be hypothesised or not, but also to determine its identity. To this end, the transfer grammar redundantly encodes the upstairs verb's object as an additional

argument role (*oind*) which is used by the grammar to restrict any additional argument slot.

Predicting properties of raised subjects (*sind*) The last feature relating to complex predicates targets modals and subject raising verbs, which agree with a subject that is not their own argument. In order to limit the number of inflected variants of potentially expensive items in the chart (they all trigger argument composition), the transfer derives the person-number information of the syntactic subject from the argument structure of their non-finite argument (cf. `--SIND` in figure 1). Since the scope of this feature is limited, we did not have any a priori expectation as to whether the potential gains in certain construction will be sufficient to offset the overhead incurred by the extended rule set.

2.3.4 Non-local dependencies

Long distance dependencies, like topicalisation, wh-fronting and relativisation are a notorious source of inefficiency in syntactic processing. In German, extraction is very common: even in ordinary declaratives, some constituent is extracted from the matrix or an embedded clause and placed into the sentence-initial *Vorfeld*, a kind of topic position. In the external MRS, the distinguished individual variable of the topicalised element is represented as an information-structural property of the proposition or question relation (TPC feature). Topicalised elements can be arguments, modifiers (scopal or intersective), as well as heads, in the case of (partial) VP fronting. Moreover, as a side-effect of wide-spread scrambling, there is no canonical position even for arguments, let alone adjuncts, so gap prediction is vital.

Local vs. non-local realisation (*top*) Predicting local vs. non-local realisation of arguments is expected to be both straightforward and effective: given that the individual variable of the topicalised element is already registered in the external MRS, it is almost sufficient to redundantly encode this fact as a property of the variable, thereby making it visible on the governing predicate as well, i.e., the context from which extraction proceeds, and mark all remaining arguments as local (cf. `--TOP` in figure 1). This basic scenario gets, of course, slightly more complicated given the fact that individual variables can be arguments of more than one predicate, which may or may not be a reason for concern: in the case of across-the-board extraction from coordinate structures, it can be harmless, whereas in the case of relativisation, we are confronted with the possibility of an individual which is realised locally with respect to the upstairs predicate, yet non-locally within the relative clause. In the transfer grammar, this is resolved by means of a three-valued system of types (+,−,*na*), where Boolean values correspond to topicalised (+) and non-topicalised (−) realisation, whereas *na* represents the neutral case (relativisation). Both local head-complement rules (*na_or_−*) and complement extraction rules (*na_or_+*) are made sensitive to this distinction.

As a consequence of this feature, we expect some considerable reduction in chart size: with the exception of relativised arguments, all arguments will be marked as either local or non-local, thereby eliminating a great deal of non-determinism.

Predict extraction type and gap site (*tpc*) While prediction of argument extraction as sketched above can reduce some of the complexity incurred by long-distance dependencies, it is rather moot when it comes to items that are not represented on the argument structure of the local head at all, e.g. modifiers.

Taking into consideration the semantic relation that the topicalised elementary predication enters in with some other elementary predication, it is possible to detect, from the external

MRS, both modifier status (scopal vs. intersective) and location of the gap site, i.e. the modified item. In a similar way, it is possible to identify cases of (partial) VP fronting.

Complementing the *top* feature, which marks extraction as a property of arguments, the transfer grammar introduces a feature *tpc* to identify the locus of the gap as a property of the head. Values of this feature serve to distinguish further between different types of extraction, e.g. intersective vs. scopal modification, verb fronting and plain argument extraction (cf. `--TPC` in figure 1). Extraction rules are made to be sensitive to properties of the head, accordingly. The ordered transfer rules first try to detect instances of modifier fronting and partial VP fronting and mark the event variable of the elementary predication corresponding to the head for the appropriate extraction type. All remaining verbal and predicative elementary predications are marked as a potential site for argument extraction, thereby ruling out modifier or verb fronting from these sites.

A priori, it is difficult to assess the efficiency of this feature. However, given the rather unconstrained nature of modification, and therefore modifier extraction, it is safe to expect some decent benefit.

3 Evaluation

In order to assess the impact of the proposed measures on generation efficiency, we carried out several experiments on three different regression test suites for German: the Babel test suite (Müller, 2004), the TSNLP test suite (Lehmann et al., 1996), and the German version of the CSLI MRS test suite. The test suites were parsed, and successfully analysed test items were subsequently disambiguated using the Redwoods treebank annotation tool (Oepen et al., 2002). This left us with a total of 2,259 semantic input representations for the generator (Babel: 609, TSNLP: 1547, MRS: 103).

None of the test suites used in the experiments here was specifically designed for the purposes of NLG. Rather, all three are general purpose, phenomenon-oriented regression test suites. However, there are some differences in the design of the individual test suites that we expect to affect the impact of our performance improvements: while the MRS and TSNLP test suites consist of rather short utterances (MRS: 4.44 words/item, TSNLP: 4.76 words/item), Babel is slightly more complex (6.76 words/item). Another important difference relates to the kind of phenomena included in the test suite: to give an example, TSNLP includes a fair amount of non-sentential items for testing NP-internal agreement, a phenomenon which should be entirely unaffected by most of the efficiency measures suggested here, which are all targeted at clausal syntax.

All test runs were performed on a Linux (kernel 2.6.32) compute server with 12 Intel Xeon X5650 2.67GHz CPUs and 16GB RAM, running 4 processes in parallel (on an otherwise idle machine). The ACE generator was run in standard configuration, i.e. with a memory limit of 1.2 GB for forest creation plus another 300 MB for unpacking. The number of realisations per item was limited to 1000. Tests have been profiled using [incr tsdb] (Oepen, 2002).

In addition to comparing the performance of the full pre-generation transfer grammar to that of the baseline, we conducted a number of additional test runs to evaluate the effectiveness of each performance feature on its own (*+feature*), as well as each feature's contribution to the combined performance (*−feature*).

3.1 Results

The main results are summarised in tables 1 and 2, giving crucial performance indicators (overall processing time and passive edges) for all three test suites. Comparing baseline performance (*Base*) to the combined effect of all features (*All*), we observe a speedup of around a factor of 4.5 for MRS and TSNLP test suites. On the more complex Babel test suite efficiency gains even go up to a factor of almost 28.[4] As indicated in table 1, average speedup on all three test suites is at 18.5.

	MRS		TSNLP		Babel		MRS+TSNLP+Babel	
	Time (s)	Red.	Time (s)	Red.	Time (s)	Red.	Time (s)	Red.
Base	0.257	1.00	0.273	1.00	7.213	1.00	2.143	1.00
+cas	0.222	1.16	0.216	1.26	4.547	1.59	1.384	1.55
+punct	0.150	1.71	0.159	1.71	4.598	1.57	1.355	1.58
+sub	0.210	1.22	0.215	1.27	6.042	1.19	1.786	1.20
+coh	0.226	1.14	0.275	0.99	5.681	1.27	1.730	1.24
+oind	0.242	1.06	0.262	1.04	5.506	1.31	1.675	1.28
+sind	0.262	0.98	0.274	1.00	7.035	1.03	2.096	1.02
+top	0.130	1.97	0.136	2.00	5.309	1.36	1.530	1.40
+tpc	0.131	1.96	0.178	1.53	3.602	2.00	1.099	1.95
All	0.054	4.71	0.063	4.31	0.260	27.79	0.116	18.52
-cas	0.056	4.59	0.065	4.20	0.371	19.46	0.147	14.58
-punct	0.063	4.11	0.081	3.38	0.415	17.39	0.170	12.61
-sub	0.058	4.44	0.067	4.09	0.304	23.71	0.130	16.44
-coh	0.054	4.78	0.061	**4.49**	0.353	20.45	0.139	15.41
-oind	0.056	4.59	0.062	4.37	0.526	13.72	0.187	11.46
-sind	0.050	**5.12**	0.062	4.40	0.258	**27.95**	0.114	18.74
-top	0.076	3.37	0.081	3.37	0.381	18.92	0.162	13.25
-tpc	0.064	4.01	0.079	3.46	0.897	8.04	0.299	7.17
-index	0.049	5.24	0.069	3.97	0.433	16.67	0.166	12.91
-pack	0.079	3.23	0.087	3.14	0.563	12.82	0.215	9.98

Table 1: Processing time and speedup factor

Similarly positive efficiency factors can be observed regarding space consumption (edges), although space savings typically fall short of time savings, given the fact that we are using a generator with ambiguity packing.[5]

Table 2 also details generation coverage achieved by each test run: on MRS and TSNLP test suites, coverage is 100% throughout. On Babel, we achieve full coverage, once a sufficient number of efficiency measures is enabled (second half of table 2). Test runs for baseline performance, as well as those with only a single performance feature activated at a time (top half of table 2), occasionally run into memory exhaustion, accounting for reduced coverage. However, since coverage on all test runs is either greater or equal to that of the baseline, a potential floor effect benefits the baseline more than any other runs, thus leaving the significance of our results unaffected.

[4] Apparently, combination of features pays off much better in terms of time savings than mere multiplication of individual factors would suggest, an effect that has been previously noted in the context of chart generation (Carroll and Oepen, 2005).

[5] Passive edges reported in table 2 are packed edges: thus, any edge filtered by our performance features can lead to time savings at several points during generation, namely edge creation, packing, and unpacking.

	MRS			TSNLP			Babel		
	Cov	Edges	Red.	Cov	Edges	Red.	Cov	Edges	Red.
Base	100.0	703	1.00	100.0	693	1.00	96.7	4864	1.00
+cas	100.0	663	1.06	100.0	630	1.10	97.9	3817	1.27
+punct	100.0	440	1.60	100.0	458	1.51	97.5	3364	1.45
+sub	100.0	555	1.27	100.0	567	1.22	96.9	4137	1.18
+coh	100.0	645	1.09	100.0	692	1.00	98.0	4460	1.09
+oind	100.0	675	1.04	100.0	676	1.02	98.2	3959	1.23
+sind	100.0	706	1.00	100.0	692	1.00	96.9	4859	1.00
+top	100.0	392	1.79	100.0	441	1.57	98.0	3539	1.37
+tpc	100.0	410	1.71	100.0	452	1.53	98.0	2931	1.66
All	100.0	116	6.07	100.0	172	4.03	100.0	554	8.78
-cas	100.0	133	5.28	100.0	194	3.57	100.0	693	7.02
-punct	100.0	179	3.92	100.0	242	2.87	100.0	860	5.65
-sub	100.0	141	4.99	100.0	195	3.55	100.0	673	7.23
-coh	100.0	124	5.67	100.0	175	3.96	100.0	657	7.41
-oind	100.0	121	5.82	100.0	174	3.99	100.0	841	5.78
-sind	100.0	117	6.00	100.0	172	4.03	100.0	558	8.72
-top	100.0	205	3.43	100.0	251	2.76	100.0	825	5.90
-tpc	100.0	150	4.68	100.0	228	3.04	100.0	1173	4.15
-index	100.0	126	5.58	100.0	198	3.50	100.0	682	7.13
-pack	100.0	215	3.27	100.0	292	2.37	99.7	1292	3.77

Table 2: Generation coverage and space consumption (passive edges)

Investigating the impact of the individual features in more detail, we find that almost all of them are effective on at least two of the three test suites. The only exception is *+sind*, the feature which calculates subject agreement information for raising and modal verbs: not only do we not find any clear benefits in isolation; its inclusion also proves detrimental in combination with other features. Given that this measure is highly specific, its failure to give rise to positive effects is hardly surprising. All other features are effective not only in isolation (top half of the table), but we can observe from the runs in the lower half of each table (leave-one-out) that each feature still has an impact when used in combination.

Starting with the inflection-oriented features, i.e., *cas* and *punct*, we find that both have consistent impact on all test suites. However, the effect of controlling punctuation is clearly stronger than that of predicting case: in fact, punctuation is the second to third most effective feature of all features tested. We believe that this is due to the following factors: first, predicting morphosyntactic case can only ever have an effect on nominal expressions (nouns, determiners, attributive adjectives), whereas punctuation will affect every lexical item in the chart that corresponds to some elementary predication in the input, targeting nominal and non-nominal expressions alike. Furthermore, while case prediction only reduces the number of potential complements, predicting punctuation also has an effect on heads and modifiers. Second, sentential punctuation is a global feature, i.e., all elementary predicates will be specialised in the same way. Case assignment, however, is a local property, and must therefore cater for the situation where an individual variable is shared by two or more governing predicates, as, e.g., in the case of relativisation. As a net effect, individual variables must be exempt from case prediction in these cases. This explanation is further supported by the fact that the reduction factor on passive edges is very close to the theoretically maximal value for punctuation of 1.67 (5/3).

The features related to verb placement and argument composition (*sub,oind,coh*) all lead to performance improvements, albeit to differing degrees, depending on the feature and the test suites: while prediction of verb placement, i.e., direction of branching and presence/absence of verb movement (*sub*), leads to consistently good effects on all three test suites, as does the prediction of the absence/identity of the initial verb's object (*oind*), the *coh* feature, which exclusively caters for argument composition, shows more variable behaviour: though beneficial otherwise, space savings on TSNLP are negligible, with processing times even going up slightly. This may not be too surprising: while prediction of verb placement and direction of branching affect every clause, the *coh* feature will only show an effect in constructions involving particular predicates or tenses, which is a situation that can vary depending on the concrete input.

Finally, the two extraction features *top* and *tpc* show again consistent and highly effective performance improvements across all test suites, both in isolation and in combination. This confirms quite neatly our initial expectation that these two efficiency measures are largely independent, the former (*top*) targeting complement extraction, by virtue of their being represented on the head's argument structure, the latter (*tpc*) targeting the remaining cases, most notably gap prediction for adjunct extraction. Finally, the fact that long distance dependencies are not only costly, but also frequent and not specific to any particular construction, explains why good gap prediction gives rise to consistently high performance benefits across all test suites.

Before we close the presentation of the main results, we would like to briefly compare the efficiency of the measures proposed here to those suggested by Carroll and Oepen (2005), namely index accessibility filtering and ambiguity packing. Disabling each of these previously established performance features in turn (cf. the last two rows in tables 1 and 2), we can show that the detrimental effect shown on our test data is comparable to that incurred by disabling one of the features investigated here.

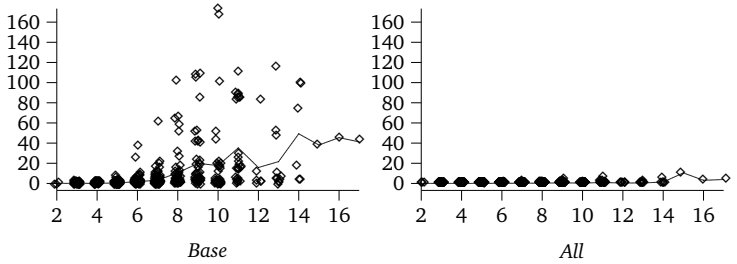


Figure 2: Processing time (in s) per string length (babel)

3.2 Discussion

We have observed during the presentation of the main results that the majority of MRS-derived efficiency features show comparable speedup effects across the three different test suites, when used in isolation. Notable exceptions were the somewhat more construction and, therefore, input-specific features *coh* and *oind* which are highly dependent on the presence of complex predicates. However, we observed quite strong differences (a factor around 5.5) as to the cumulative effects between babel on the one hand and the less complex TSNLP and MRS test suites on the other. In order to better understand the significance of the experiments reported on here we shall investigate the differences and discuss what practical implications will ensue.

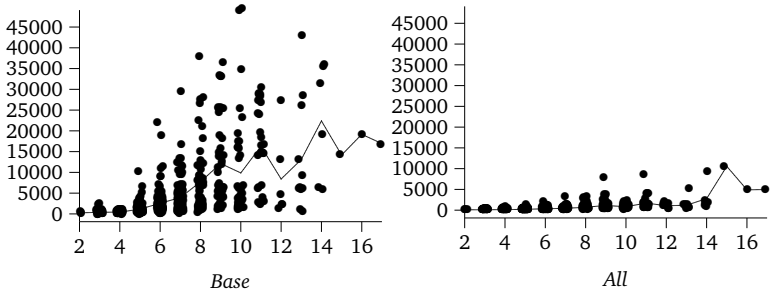


Figure 3: Passive edges per string length (babel)

The most obvious difference between Babel and the other two test suites is of course input length: by artificially reducing average input length on Babel to slightly above (4.85) that of TSNLP (by filtering out longer inputs), the cumulative speedup factor reduces to a factor around 9.5, compared to 27.5, which is still not fully comparable, yet much closer to the factor of 4.5 observed for MRS and TSNLP suites.[6] The impact of input length on relative generation speedup is also corroborated by the scatter plots of time and space (passive edge) consumption shown in figures 2 and 3: without any of the efficiency measures proposed in this paper, processing time begins to explode already at an average sentence length of 8 (see figure 2), averaging at around 8.4s. With the efficiency measures, processing time never even comes close to that level, leading to massive performance gains on longer inputs. The comparison of passive edges in figure 3 confirms even more clearly how the current efficiency measures particularly counter the combinatorial explosion observable with the baseline. To summarise, while all test suites witness good reduction of average processing times, it is clear that the real benefit of the generation efficiency measures suggested here becomes apparent with longer (and therefore more complex) inputs. For practical purposes, in particular for online processing, taming of the worst case complexity for longer inputs is more important than speedup factors on the relatively short utterances characteristic of MRS and TSNLP test suites.

Before we close, we should like to address the issue of how our method could be ported to grammars for languages other than German: while some of the concrete features we used are somewhat specific to German (or Dutch), others should be easily portable. The punctuation feature, as well as the the two extraction-related features *top* and *tpc* should be useful to improve generation efficiency in a wide range of languages: in a small experiment carried out with the ERG (Copestake and Flickinger, 2000) on the Rondane treebank (see section 1.2), we observed a 10.5% reduction in generation time for punctuation alone. We expect that other features, such as the *cas* feature will be useful for other less configurational languages, such as Slavic languages, given the fact that elaborate case systems and relatively free word order often go hand in hand.

3.3 Related work

An alternative strand of approaches towards efficient processing of unification grammars builds on the idea of compiling these grammars into formalisms with better worst-case complexity

[6] Reducing average string length to slightly below that of the MRS test suite (4.37), results in a cumulative speedup factor of only 5.4.

than native unification-based processing, such as CFG or TAG: e.g., Kiefer and Krieger (2000) proposed a CFG superset approximation of HPSG for parsing English and Japanese. However, this method has so far never been successfully applied to German, let alone for generation. Furthermore, despite potentially better raw performance, CFGs are plagued by at least as severe locality issues as bottom-up HPSG generation. TAGs, by contrast, with their extended domain of locality, constitute a much more interesting target formalism for compiling an HPSG generation grammar. In the literature, two such approaches have been reported: Kasper et al. (1995) describe a method of compiling Klaus Netter's HPSG of German to TAG, but the compilation did not cover the full grammar but only a fragment, and, unfortunately, no performance measures are reported for either parsing or generation. Becker and Lopez (2000) specifically capitalise on the fact that TAG's extended domain of locality gives rise to an a priori expectation towards greater generation efficiency, and, building on Kasper et al. (1995), they describe a compilation of the Verbmobil English and Japanese HPSG grammars into LTAG. Again, however, no performance tests are reported that could substantiate the claim of increased generation performance with the compiled grammar. Furthermore, to the best of our knowledge, no such compilation has ever been carried out for German.

In the context of native unification-based processing, Gardent and Kow (2007) suggest a method to enrich the semantic input to an FTAG of French with tree features that permit almost deterministic selection of generation paraphrases. Moreover, Gardent and Kow (2005) argue that such selection also leads to performance improvements, as they show on the basis of sample sentences. With respect to the German LFG (Rohrer and Forst, 2006; Cahill et al., 2007), Zarrieß and Kuhn (2010) propose a transfer approach to provide f-structure input for the XLE surface realiser from shallow semantic representations. The main motivation for this was that f-structures contain a high level of syntactic and even morphosyntactic detail that make them less suitable for paraphrasing and, more generally, for deployment in natural language generation systems. Zarrieß and Kuhn (2010) also discuss the impact of grammatical function prediction from semantic roles for generator efficiency: depending on the complexity of the transfer rules, they observe considerable differences in average generation time, ranging from 246.14s for the "naive rules" to 36.2s for their "informed rules", which operate on configurations rather than individual roles. Nakanishi et al. (2005) propose a beam search approach to tackle generation efficiency for an English HPSG. We believe their approach to be complementary to ours, since MRS enrichment can prune the search space with certainty and without locality restrictions, such that a future system using both methods will be able to provide good results at smaller beam sizes, taking advantage of a division of labour between transfer-based treatment of non-local and probabilistic pruning of local dependencies.

Conclusion

We have proposed a method to improve generation efficiency with GG, a reversible HPSG of German. Using a term rewrite system integrated into the generator, we automatically enrich the purely semantic input representation with additional syntacto-semantic constraints, derived from the semantic configuration. Evaluating our method on three different regression test suites for German, we have shown that this approach is highly successful in taming combinatorial explosion in bottom-up chart generation, leading to significant speedup factors: while on less complex inputs, we achieved a speedup by a factor of around 4.5, performance gains increase considerably on more complex inputs, yielding a speedup factor of almost 28, which shows that our method scales up well to increasing input lengths.

References

Becker, T. and Lopez, P. (2000). Adapting HPSG-to-TAG compilation to wide-coverage grammars. In *Proceedings of the 5th International Workshop on Tree-Adjoining Grammars and Related Formalisms (TAG+5)*, pages 47–54, Paris.

Cahill, A., Forst, M., and Rohrer, C. (2007). Stochastic realisation ranking for a free word order language. In *Proceedings of the Eleventh European Workshop on Natural Language Generation*, pages 17–24, Saarbrücken, Germany. DFKI GmbH. Document D-07-01.

Carroll, J. and Oepen, S. (2005). High efficiency realization for a wide-coverage unification grammar. *Natural Language Processing–IJCNLP 2005*, pages 165–176.

Copestake, A. (2002). *Implementing Typed Feature Structure Grammars*. CSLI Publications, Stanford.

Copestake, A. and Flickinger, D. (2000). An open-source grammar development environment and broad-coverage English grammar using HPSG. In *Proceedings of the Second conference on Language Resources and Evaluation (LREC-2000)*, Athens.

Copestake, A., Flickinger, D., Pollard, C., and Sag, I. (2005). Minimal recursion semantics: an introduction. *Research on Language and Computation*, 3(4):281–332.

Crysmann, B. (2003). On the efficient implementation of German verb placement in HPSG. In *Proceedings of RANLP 2003*, pages 112–116, Borovets, Bulgaria.

Crysmann, B. (2005). Relative clause extraposition in German: An efficient and portable implementation. *Research on Language and Computation*, 3(1):61–82.

Crysmann, B. (2007). Local ambiguity packing and discontinuity in German. In Baldwin, T., Dras, M., Hockenmaier, J., King, T. H., and van Noord, G., editors, *Proceedings of the ACL 2007 Workshop on Deep Linguistic Processing*, pages 144–151, Prague, Czech Republic. Association for Computational Linguistics.

Gardent, C. and Kow, E. (2005). Generating and selecting grammatical paraphrases. In *Proceedings of the 10th European Workshop on Natural Language Generation*.

Gardent, C. and Kow, E. (2007). A Symbolic Approach to Near-Deterministic Surface Realisation using Tree Adjoining Grammar. In *45th Annual Meeting of the Association for Computational Linguistics - ACL 2007*, pages 328–335, Prague, Czech Republic. Association for Computational Linguistics.

Kasper, R., Kiefer, B., Netter, K., and Vijay-Shanker, K. (1995). Compilation of HPSG to TAG. In *Proceedings of the 33rd annual meeting on Association for Computational Linguistics*, ACL '95, pages 92–99, Stroudsburg, PA, USA. Association for Computational Linguistics.

Kay, M. (1996). Chart generation. In *Proceedings of the 34th annual meeting on Association for Computational Linguistics*, pages 200–204. Association for Computational Linguistics.

Kiefer, B. and Krieger, H.-U. (2000). A context-free approximation of Head-Driven Phrase Structure Grammar. In *Proceedings of the 6th International Workshop on Parsing Technologies (IWPT 2000)*, pages 135–146.

Kilian, N. (2007). Zum Punkt gekommen. Master's thesis, Universität des Saarlandes, Saarbrücken.

Kiss, T. and Wesche, B. (1991). Verb order and head movement. In Herzog, O. and Rollinger, C.-R., editors, *Text Understanding in LILOG*, number 546 in Lecture Notes in Artificial Intelligence, pages 216–240. Springer-Verlag, Berlin.

Lehmann, S., Oepen, S., Regnier-Prost, S., Netter, K., Lux, V., Klein, J., Falkedal, K., Fouvry, F., Estival, D., Dauphin, E., Compagnion, H., Baur, J., Balkan, L., and Arnold, D. (1996). Tsnlp - test suites for natural language processing. In *The 16th International Conference on Computational Linguistics (COLING)*, volume 2, pages 711–716, Copenhagen, Denmark. Center for Sprogteknologi.

Müller, S. (2004). Continuous or discontinuous constituents? A comparison between syntactic analyses for constituent order and their processing systems. *Research on Language and Computation*, 2(2):209–257.

Müller, S. and Kasper, W. (2000). HPSG analysis of German. In Wahlster, W., editor, *Verbmobil: Foundations of Speech-to-Speech Translation*, pages 238–253. Springer, Berlin.

Nakanishi, H., Miyao, Y., and Tsujii, J. (2005). Probabilistic models for disambiguation of an HPSG-based chart generator. In *Proceedings of the Ninth International Workshop on Parsing Technology*, pages 93–102. Association for Computational Linguistics.

Oepen, S. (2002). *Competence and Performance Profiling for Constraint-based Grammars: A New Methodology, Toolkit, and Applications*. PhD thesis, Saarland University.

Oepen, S., Callahan, E., Flickinger, D., Manning, C., and Toutanova, K. (2002). LinGO Redwoods: A rich and dynamic treebank for HPSG. In *Beyond PARSEVAL. Workshop at the Third International Conference on Language Resources and Evaluation, LREC 2002*, Las Palmas, Spain.

Oepen, S., Dyvik, H., Lønning, J. T., Velldal, E., Beermann, D., Carroll, J., Flickinger, D., Hellan, L., Johannessen, J. B., Meurer, P., Nordgård, T., and Rosén, V. (2004). Som å kapp-ete med trollet? Towards MRS-based Norwegian–English Machine Translation. In *Proceedings of the 10th International Conference on Theoretical and Methodological Issues in Machine Translation*, Baltimore, MD.

Oepen, S., Velldal, E., Lønning, J. T., Meurer, P., Rosén, V., and Flickinger, D. (2007). Towards hybrid quality-oriented machine translation. On linguistics and probabilities in MT. In *Proceedings of the 11th International Conference on Theoretical and Methodological Issues in Machine Translation*, Skövde, Sweden.

Rohrer, C. and Forst, M. (2006). Improving coverage and parsing quality of a large-scale LFG for German. In *Proceedings of LREC-2006*, Genova.

Zarrieß, S. and Kuhn, J. (2010). Reversing f-structure rewriting for generation from meaning representations. In Butt, M. and King, T. H., editors, *Proceedings of the LFG10 Conference, Ottawa*, pages 479–499, Stanford. CSLI publications.

A Corpus-Based Study of Edit Categories in Featured and Non-Featured Wikipedia Articles

Johannes Daxenberger[1] *Iryna Gurevych*[1,2]
(1) Ubiquitous Knowledge Processing Lab (UKP-TUDA)
Department of Computer Science, Technische Universität Darmstadt
(2) Ubiquitous Knowledge Processing Lab (UKP-DIPF)
German Institute for Educational Research and Educational Information
www.ukp.tu-darmstadt.de

ABSTRACT

In this paper, we present a study of the collaborative writing process in Wikipedia. Our work is based on a corpus of 1,995 edits obtained from 891 article revisions in the English Wikipedia. We propose a 21-category classification scheme for edits based on Faigley and Witte's (1981) model. Example edit categories include spelling error corrections and vandalism. In a manual multi-label annotation study with 3 annotators, we obtain an inter-annotator agreement of $\alpha = 0.67$. We further analyze the distribution of edit categories for distinct stages in the revision history of 10 featured and 10 non-featured articles. Our results show that the information content in featured articles tends to become more stable after their promotion. On the opposite, this is not true for non-featured articles. We make the resulting corpus and the annotation guidelines freely available.[1]

TITLE AND ABSTRACT IN GERMAN

Eine Korpusbasierte Studie von Änderungstypen in Exzellenten und Nicht-Exzellenten Wikipedia-Artikeln

In dieser Arbeit stellen wir eine Studie über den kollaborativen Schreibprozess in Wikipedia vor. Unsere Studie basiert auf einem Korpus aus 1.995 Änderungen in 891 Artikelrevisionen der englischen Wikipedia. Wir schlagen ein Klassifikationsschema mit 21 Änderungenstypen vor, basierend auf dem Modell von Faigley and Witte (1981). Unter den Änderungenstypen befinden sich beispielsweise Rechtschreibkorrekturen und Vandalismus. In einer manuellen multi-label Annotationsstudie mit 3 Annotatoren erzielen wir eine Interrater-Reliabilität von $\alpha = 0.67$. Wir analysieren außerdem die Verteilung von Änderungstypen zu unterschiedlichen Stadien in der Revisionsgeschichte von 10 exzellenten und 10 nicht-exzellenten Artikeln. Unsere Ergebnisse zeigen, dass der Informationsgehalt in exzellenten Artikeln nach ihrer Auszeichnung tendenziell stabiler wird. Im Gegensatz dazu ist das bei nicht-exzellenten Artikeln nicht der Fall. Das dabei entstandene Korpus und die Annotationsrichtlinien stellen wir zur freien Verfügung.

KEYWORDS: Wikipedia, Revision History, Collaborative Writing, Quality Assessment.

KEYWORDS IN GERMAN: Wikipedia, Revisionsgeschichte, Kollaboratives Schreiben, Qualitätsbewertung.

[1] http://www.ukp.tu-darmstadt.de/data/wiki-edits/

Proceedings of COLING 2012: Technical Papers, pages 711–726,
COLING 2012, Mumbai, December 2012.

1 Introduction

Team work on a single product is a common process in daily life. Online collaboration software supports project management, version control systems enable the collaborative development of source code, and recent developments in cloud computing have generated new ways of collaborating on single files. A lot of research has been devoted to the development of user-friendly tools and editors for collaborative writing (Noel and Robert, 2004). Free tools include web-based software such as Zoho Writer, Google Drive or Etherpad, as well as Wikis such as Twiki, Foswiki and MediaWiki. Corpora for analyzing the writing process mostly come from the educational domain (Lee and Webster, 2012). An exception is the Digital Variants Archive[2], which contains contemporary texts by Spanish and Italian authors including various revisions of those texts. However, these corpora only consist of textual revisions by one author. Although there are many tools enabling users to collaboratively write texts, little work has been done to analyze the underlying collaboration process of the data that is created with these tools. One possible reason is that few corpora for analyzing collaborative writing are available.

Since their invention in the mid 90s, Wikis have become one of the most important tools for creating and sharing contents. They enable a detailed tracking of changes, as they usually implement a revision control system which saves every change to a page. At the time of writing, the number of revisions in the English online encyclopedia Wikipedia kept growing by 3.2 million revisions each month.[3] Various studies have processed parts of that data for different tasks such as extracting sentence simplification (Woodsend and Lapata, 2011) or spelling error correction (Zesch, 2012).

Whenever an editor of a page in Wikipedia saves changes, a new revision is created. As one revision may contain a set of distinct local changes, we distinguish between revisions and edits. We define an *edit* as a coherent local change, usually perceived by a human reader as one single editing action. For a pair of adjacent revisions, we denote the previous revision with r_{v-1} and the newer revision with r_v. For each (r_{v-1}, r_v)-pair, we calculate a set of n edits $e^k_{v-1,v}$ (where $k = \{0, 1, ...n-1\}$) that have been made to transform r_{v-1} into r_v (see Section 3.2). We label edits with *edit categories*.

Our contribution in this study is three-fold. First, we develop a classification system for edit categories based on established models from research on the writing process (Faigley and Witte, 1981). This addresses the proposal of Ferschke et al. (2012a) to investigate on the classification of textual revisions. The goal is to facilitate data extraction for NLP applications building upon revision history data. Second, we compile and annotate a corpus tailored towards a qualitative analysis of Wikipedia revisions and based on a set of edits and release it for free access to the research community. To the best of our knowledge, such a corpus is not available yet. Third, based on the annotations in our corpus, we analyze differences in the collaborative writing process of featured and non-featured articles. Featured articles are promoted as such after an internal reviewing process which confirms the required quality standards in Wikipedia (cf. Section 3.2). Although it was not possible to identify a relationship between a certain type of collaboration and article quality in terms of featured and non-featured articles, we show that the collaborative behavior among authors significantly changes once an article is awarded featured status.

The rest of this paper is structured as follows: Section 2 presents the related work. In Section

[2] `http://www.digitalvariants.org/` (accessed 2012-10-29)
[3] Source: `http://stats.wikimedia.org/EN/TablesDatabaseEdits.htm` (accessed 2012-10-29)

3, we describe our edit classification scheme and the corpus. Furthermore, we explain and evaluate the manual annotation of our corpus. Section 4 discusses the findings of our study with respect to related approaches. Finally, we summarize the main conclusions.

2 Related Work

2.1 Collaborative Writing

Sommers (1980) investigated the connections between writing and quality, particularly with respect to differences in the types of edits performed by experienced and unexperienced writers. Her analysis shows that unexperienced writers tend to revise at the sentence or word level, i.e. to make changes on the surface of the text. On the contrary, experienced writers are rather concerned with the meaning and structure of the entire text, that is, they make changes to the text base. In the later research, there has been a shift from revising one's own work (single-author writing) to collaboratively working on a single document (collaborative writing), cf. Ede and Lunsford (1990). Generally, the importance of collaborative writing has grown over the last decades and receives increased interest due to recent developments in the Web 2.0. Collaboration in Wikipedia has been subject to a series of studies (Liu and Ram, 2011). Wikipedia's revision history reflects a type of distributed collaboration, as the interaction between authors is strictly indirect. The communication between authors takes place via the metadata related to each revision in Wikipedia such as the author comment, the revision timestamp and the author name or IP address.

2.2 Edit Classification Schemes

Faigley and Witte (1981) present the first taxonomy capturing the intentions behind a textual change. Their scheme is designed to analyze the effects of edits on meaning. They define meaning as either inserting new information to the text or deleting old information. Edits which affect meaning are called *Text-Base Changes*; edits which do not affect meaning are called *Surface Changes*. They further divide Surface Changes into *Formal Changes* (mostly copy-edits like spelling corrections etc.) and *Meaning-Preserving Changes* (paraphrases). Text-Base Changes are split into *Microstructure* and *Macrostructure Changes*, where the former describe minor changes and the latter refer to changes that affect the summary or gist of the entire text. Meaning-Preserving Changes, Microstructure Changes and Macrostructure Changes are further divided into Additions, Deletions, Substitutions, Permutations, Distributions and Consolidations. Various studies have classified edits in Wikipedia; we compare them in Table 1.

Pfeil et al. (2006) propose a taxonomy of 13 categories, aiming to compare cultural differences in the writing process of one article in four language versions of Wikipedia (German, Dutch, French and Japanese). Their taxonomy is based on an analysis of the data, not on existing revision theories. Two annotators manually examined and labeled the 500 revision pairs in their corpus. They allowed for multi-labeling and resolved disagreement by discussion. No inter-annotator agreement is reported.

Jones (2008) analyzes differences in the collaborative writing process of featured and non-featured articles in Wikipedia. His taxonomy is based on Faigley and Witte's (1981) distinction between Macrostructure and Microstructure changes. His corpus consists of 10 Wikipedia articles which were nominated to be featured in January 2007, from which 5 were actually promoted and the other 5 were denied the featured article status. For the annotation process, he relies on revision comments that have been generated either by the authors or automatically,

	Pfeil et al. (2006)	Jones (2008)	Liu and Ram (2011)
Wikipedia Policy	Vandalism	Vandalism	Revert
	Reversion	Revert	
		Disambiguation	
Text-Base	Add Information	Significant addition	Sentence creation
	Delete Information	Significant deletion	Sentence deletion
	Clarify Information	Structural change	Sentence modification[a]
	Add Link	Add image	Link creation
	Delete Link	Fix or delete image	Link deletion
	Fix Link	Add link	Link modification
		Fix or delete link	Reference creation
			Reference deletion
			Reference modification
Surface	Style/Typography	Style or readability	
	Spelling		
	Grammar		
	Format		
	Mark-up Language		

[a] As Liu and Ram (2011) state, this category includes grammar and spelling changes. Hence, it is not entirely a Text-Base category.

Table 1: Three studies classifying revisions in Wikipedia and the categories they use.

not on the actual revision texts. As only one person annotated the corpus, no inter-annotator agreement is reported.

Liu and Ram (2011) study the relationship between collaboration and article quality in Wikipedia, aiming to identify types of authors (e.g. Starter, Copy Editors, All-round contributors). Their taxonomy builds on Pfeil et al. (2006). However, they transformed the taxonomy into higher-level categories, merging clarification and grammar-spelling into sentence modification. By doing this, they are able to automatically identify edit categories; however, because the annotation of edits is not done manually, no inter-annotator agreement can be reported. While the automatic identification of edit categories allows for analyzing a larger corpus, it blurs Faigley and Witte's (1981) distinction between Surface and Text-Base changes. The authors do not report on the quality of the automatic edit category identification. Their corpus consists of 1,600 English Wikipedia articles from March 2010, divided into each 400 articles which have been nominated as either featured, good, B- or C-class according the WikiProject article quality grading scheme[4]. Using these four kinds of Wikipedia-internal evaluated quality grades, Liu and Ram (2011) study the relationship between collaboration and article quality. For their analysis, the authors used only the revisions before the respective nomination of the articles. The novel contribution of their work is that they calculate edits with a higher granularity (sentence level). Each (r_{v-1}, r_v)-pair may be multi-labeled with a list of edit categories to reflect the number of edits.

Other approaches propose special purpose classification systems for Wikipedia edits. Among the latter, Chin et al. (2010) focus on vandalism classification. Their top-level categories are Revert, Delete, Insert and Change; their system cannot easily be compared to the aforementioned systems, which distinguish between Text-Base and Surface changes. They introduce a basic

[4] `http://en.wikipedia.org/wiki/Wikipedia:Version_1.0_Editorial_Team/Assessment` (accessed 2012-10-29)

distinction between content and format changes. Content includes text, links and images, format refers to HTML/CSS and templates. Fong and Biuk-Aghai (2010) present a system to automatically calculate and categorize edits. Similar to our system, their system computes a list of basic edit actions on the unparsed source text (i.e. including markup). Edits are calculated with granularity at the sentence and token level. The authors suggest a set of rules and categories to label the basic edit actions calculated before. Examples of their categories are (De)Wikify, Content Modification or Spelling Correction. The implementation and evaluation of their system is rather preliminary. Bronner and Monz (2012) distinguish between Factual and Fluency edits. They segment adjacent revisions into edits and classify them in a supervised machine learning system. Their system successfully classifies edits into Factual and Fluency edits with a maximum accuracy of 0.88.

Except for Bronner and Monz (2012), all of the above presented annotation studies label pairs of adjacent revisions, not edits. Hence, even if multi-labeling is applied, it is not possible to reassign each local edit $e^k_{v-1,v}$ with a category from the set or list of categories assigned to the (r_{v-1}, r_v)-pair. No manual annotation study which explicitly analyzes the agreement between raters has been carried out so far. Hence, the reliability of previous annotations is unclear. We address these issues as we annotate a set of edits rather than revisions. Furthermore, we evaluated our annotation study with a detailed inter-annotator agreement and error analysis.

3 Proposed Edit Classification

3.1 Classification Scheme

Our approach of classifying edits in Wikipedia builds upon previous work on document revision classification (Faigley and Witte, 1981) and studies about edits in Wikipedia (Pfeil et al., 2006; Jones, 2008; Liu and Ram, 2011). We follow Faigley and Witte (1981) and define the top level layers *Surface* and *Text-Base*, which differentiate between meaning-preserving and meaning-changing edits. However, contrary to Faigley and Witte (1981), we do consider all deletions and insertions of text as Text-Base changes. The only categories for textual edits in the Surface layer are PARAPHRASE and RELOCATION, cf. Table 2. To keep the taxonomy manageable, we do not follow Faigley and Witte (1981) in their fine-grained distinction of textual edits in Additions, Deletions, Substitutions, Permutations, Distributions and Consolidations. Our taxonomy is hierarchical with the three top layers *Wikipedia Policy*, *Surface* and *Text-Base*. Table 2 presents a short explanation and example for each category.

VANDALISM and REVERT are edit categories related to **Wikipedia Policies**. We define VANDALISM as an edit deliberately compromising Wikipedia's integrity (Adler et al., 2011). A REVERT undoes past edits by restoring previous revisions or parts of them (Flöck et al., 2012). As for the **Surface** layer, we include changes to the markup, as well as relocations, spelling and grammar corrections and paraphrases. We define all elements related to the Wiki markup language (see the examples in Table 2) as MARKUP. This includes HTML code, which can also be used in Wikipedia to render the layout of a page. The RELOCATION category is assigned to edits which move entire lines (copy-paste). We use the SPELLING/GRAMMAR category to label corrections of spelling or grammatical errors. Edits which rephrase or paraphrase words or sentences without altering their meaning, are labeled with the PARAPHRASE category. In the **Text-Base** layer, we define the INFORMATION category which labels meaning-changing edits to the text itself. We use the FILE category to label edits related to media types like images, videos or audio files. The REFERENCES category is assigned to edits affecting internal and external links as well as

Category	Description	r_{v-1}	r_v
Wikipedia Policy *Invalid edits as defined by internal Wikipedia Policies and respective defense mechanisms*			
Vandalism	Edits deliberately compromising Wikipedia's integrity	Einstein's **key** insight was	Einstein's **cheese master** insight was
Revert	Edits restoring a previous state of a page	**Hahahahhahahahahahahaha:)**	
Surface *Edits not affecting the meaning of the text*			
Paraphrase	Textual edits paraphrasing words or sentences	denominations **like** the	denominations **such as** the
Spelling/Grammar	Edits correcting spelling or grammatical errors	in **the** Ireland	in Ireland
Relocation	Edits moving entire lines[a]	**... -->** {{ChineseText}} **[[Category ...**	**...Dynasty}}** {{ChineseText}} ***[[Chinese...**
Markup-Insert	Edits affecting markup segments	an infant Parasaurolophus	an infant "Parasaurolophus"
Markup-Delete		'"AcDec"'	"AcDec"
Markup-Modify		=== The geometry of gravitation ===	== The geometry of gravitation ==
Text-Base *Edits affecting the meaning of the text*			
Information-Insert	Textual edits affecting information content	effects were tested by	effects were **both** tested by
Information-Delete		it is **not** a sacrament	it is a sacrament
Information-Modify		open steppes of Kashmir and **Siberia**.	open steppes of Kashmir and **Manchuria**.
File-Insert	Edits affecting files (media content)		**[[File:Dholeskull.jpg|...]]**
File-Delete		**[[Image:Victoria_Cross_bar.JPG|...]]**	
File-Modify		{{Infobox...|image=UMCLogo.**svg**... }}	{{Infobox...|image=UMCLogo.**xml**... }}
Reference-Insert	Edits affecting links, interwiki/language links or bibliographical references and citations		**[[sl:Erlang]]**
Reference-Delete		**[[**spondee**]]**	spondee
Reference-Modify		[[**molar**]]	**[[molar (tooth)|**molar]]
Template-Insert	Edits affecting templates (for including text from other pages, automatic text generation etc.)		**{{cite book|... }}**
Template-Delete		**{{clear}}**	
Template-Modify		{{**Unit** m|2|1}}	{{**Convert**|2|m|**ft**|1}}
Other	*Segmentation Errors*	[[pirate**s]]**	[[pirate**]]s**

[a] To increase readability, line breaks are omitted in the examples.

Table 2: Classification of Wikipedia edits with truncated examples from our corpus. Where necessary, we added the context, while the actual edit is bold-faced in r_{v-1} and/or r_v.

bibliographical citations. Different from Liu and Ram (2011), we do not distinguish between links and citations, as these edits refer to the same action in the sense of referencing something. Finally, the TEMPLATE category labels all edits related to templates. In Wikipedia, templates are indicated by double curly brackets and are used for including text from other pages, creating standardized messages or other automated text generation tasks.

All Text-Base edits and those in the MARKUP category are further divided into Insertions (I), Deletions (D) and Modifications (M). Insertions apply when new content or markup is added to the article, i.e. if the content or markup of $e^k_{v-1,v}$ has not been present in r_{v-1} but is present in r_v. Correspondingly, deletions remove the content or markup of $e^k_{v-1,v}$, so that the text that has been present in r_{v-1} is not present in r_v. Modifications apply to content and markup belonging to the same segment which has been changed from r_{v-1} to r_v. Here, we define a *segment* as the source element which is affected by the category of the respective edit, e.g. for modifications of the MARKUP, a markup element must be changed, for FILE edits, the embedded file must be changed etc. Correspondingly, a TEMPLATE-M edit must change the type of the template (i.e. its name) and not just a parameter of the template, as indicated in the respective example in Table 2.

We classify changes to the source text of a Wiki page, as opposed to the visual changes on the pages surface, i.e. the translated HTML which is displayed in the browser. We believe this yields a more accurate analysis of the writing process itself. Our taxonomy is geared toward edits in Wikis; however, it is fully language independent.

3.2 Corpus Construction

To draw conclusions about the relationship between the writing process and article quality, we determine distinguished articles based on the *featured* label[5] as defined by the Wikipedia community (Stvilia et al., 2008). Wikipedia has an internal review system to label articles that meet certain predefined quality criteria, e.g. they should be comprehensive, contain images where appropriate etc. The highest status an article can achieve is the featured status. Kittur and Kraut (2008) validated a set of articles with ratings from external users and found that the agreement between the external ratings and the internal ratings according to the WikiProject article quality grading scheme is substantial. For each featured articles (FA) in the English Wikipedia, we selected a non-featured article (NFA) with equal character length. From these article pairs, we randomly selected 10 pairs with equal or almost equal edit frequency (i.e. number of revisions per day) from different size ranges (see Table 3). Although we can assume that the FAs in our corpus have high quality, the NFAs show a broad quality spectrum according to the ratings by the WikiProjects' quality assessment teams, ranging from Start- to Good-class articles. However, none of the NFAs have been rated with the highest quality scores, namely featured or A-class. The selected articles cover a range of topics on historical, scientific and political issues. The youngest article is almost 6 years, the oldest is more than 9. We call the result *Wikipedia Quality Assessment Corpus* (WPQAC).

***Pre* and *Post* Revision Groups** From these article pairs, we selected 891 revisions containing 1,995 edits for the annotation study. From the FAs, we took the revision at the time of promotion to featured status (referred to as r_{prom}) specified on the respective Talk page as the reference and divided the article history into a *pre* and a *post* stage. *Pre* denotes all revisions made

[5] `http://en.wikipedia.org/wiki/Wikipedia:Featured_articles` (accessed 2012-10-29)

FA	NFA	Size	Freq.
1941 Atlantic hurricane season	Dactylic hexameter	18	0.1
William de Corbeil	European Liberal Democrat and Reform Party	26	0.1
Victoria Cross (Canada)	Erlang (programming language)	27	0.2
Deinosuchus	Intel 8086	32	0.2
Winfield Scott Hancock	Dhole	44	0.2
Laplace-Runge-Lenz vector	United Nations Relief and Works Agency	63	0.2
Introduction to general relativity	Subwoofer	70	0.4
United States Academic Decathlon	John Cage	78	0.5
Song Dynasty	Haile Selassie I	106	1.1
Euclidean algorithm	United Methodist Church	109	0.5

Table 3: The size of the latest revision (in 1,000 characters including Wiki markup) and edit frequency (average number of revisions per day) in WPQAC are equal for each FA-NFA pair.

Group	N_e	N_r	N_e/N_r
pre-FA	515	234	2.2
post-FA	485	144	3.4
pre-NFA	496	256	1.9
post-NFA	499	257	1.9
all	1995	891	2.2

Table 4: Revision groups in the annotated part of WPQAC with absolute numbers of edits and revisions.

previously to r_{prom} and *post* all revisions made after r_{prom}. Then, for each of the ten article pairs, we selected approximately 200 edits, namely each 50 edits from (r_{v-1}, r_v)-pairs

- in the second quarter of the pre stage of the FA article history (**pre-FA**),
- in the second half of the post stage of the FA article history (**post-FA**),
- in a pre-FA parallel stage in the NFA article history (**pre-NFA**),
- in a post-FA parallel stage in the NFA article history (**post-NFA**).

This way, we ensure that pre and post stage are comparable for all article pairs in our corpus with respect to the date of promotion of the FA. The annotated corpus is therefore split into four groups, with about 500 edits each, see Table 4. Slight differences in the sizes of the groups result from the fact that we had to choose adjacent revisions for each article and stage. These revisions contain diverging numbers of edits which did not always sum up to precisely 50.

The corpus has been selected to reflect the entire range of possible edits in Wikipedia, including bot edits, vandalism and reverts. Hence, no further filtering is done.

Edit segmentation The raw data for our corpus is extracted from the English Wikipedia Revision History, from the dump as of April 2011. We process the revision content (text with markup) using the Wikipedia Revision Toolkit (Ferschke et al., 2011). We do not parse the revision text, as we want to include both edits affecting the content and edits affecting the layout into one taxonomy. For each (r_{v-1}, r_v)-pair, we calculate all of the n changes $e^k_{v-1,v}$ that have been made to the current revision via an adapted version of the diff comparison algorithm by Heckel (1978). The algorithm splits each revision into its lines and numbers them. Then, it compares each line in r_{v-1} with each line in r_v to find differences in terms of inserted, deleted, modified and relocated lines. Although we only work with data from the English Wikipedia in this study, the segmentation process is fully language independent.

Inside modified lines, we additionally detect and mark changes (i.e. deletions, insertions and modifications) in situ using Neil Fraser's google-diff-match-patch library[6]. The last step is only done where the ratio of the number of overall changes in that line to the number of tokens in that line does not exceed a certain threshold. The latter serves to avoid splitting heavily edited lines into a very high number of counterintuitive edits. If, for example, stopwords like "the" or "a" are the only unchanged segments inside a modified line, we want the entire line to be marked as modified. We do further post-processing to recognize and merge associated edits, e.g. when adding a link (to merge `[[` and `]]`). This may yield errors as Wiki markup is a context-sensitive language and hence difficult to parse. In the manual annotation study, we annotate segmentation errors due to associated edits which have not been detected and merged by our algorithm with the OTHER category (cf. Table 2).

Our annotation study is carried out on edits as calculated by the segmentation algorithm explained above. The basic types of edits which the algorithm detects are insertions, deletions, modifications and relocations. Correspondingly, each (r_{v-1}, r_v)-pair can create more than one object to classify, depending on the number of edits it contains. Our annotated corpus consists of $N_r = 891$ revisions containing $N_e = 1,995$ edits. The median of edits per revision is 1, the standard deviation is 14.5 with a minimum of 1 and a maximum of 55 edits per revision. That is, most of the changes in our corpus modify articles in only one particular place.

3.3 Annotation Study

We employed three non-native speakers with working knowledge of the Wikipedia policies and markup to label the corpus based on written annotation guidelines. We define the annotation task as a multi-label classification, i.e. each $e^k_{v-1,v}$ calculated from a (r_{v-1}, r_v)-pair is assigned a set of categories $Y \subset L$, where L is the set of categories as defined in Table 2 (hence $|L| = 21$ and $|Y| \geq 1$). If, for example, an entire sentence is rewritten, this might not only affect the words but also the markup (e.g. when a bold-faced word is deleted) or references (e.g. when a link is added). Such an edit would be multi-labeled with INFORMATION-M and MARKUP-D or REFERENCE-I respectively. Further guidelines include the following:

- Edits labeled as VANDALISM, REVERT, RELOCATION or OTHER cannot be multi-labeled.
- If $e^k_{v-1,v}$ is labeled as VANDALISM, all $e^0_{v-1,v}, e^1_{v-1,v}, ...e^n_{v-1,v}$ must be labeled as VANDALISM, since all of those edits have the same author (with bad intentions).
- Edits removing or inserting white spaces or line breaks are labeled as MARKUP.

For the annotation of edits, we used the Apache UIMA[7] Cas Editor. That way, we were able to directly annotate on the source files which are produced by the UIMA pipeline we use to extract the raw text for each revision and to segment each (r_{v-1}, r_v)-pair into a list of edits. The annotators had access to all metadata information (author name, comment etc.) and the entire text of r_{v-1} and r_v.

We derive the gold standard annotations by means of a majority vote for each category. That means, for each $e^k_{v-1,v}$ which has been labeled with $l \in L$ by at least 2 annotators, we assign the category l in the gold standard. If all 3 annotators disagreed, i.e. if an edit was labeled with none of the categories at least 2 times, it is assigned the OTHER category in the gold standard. For example, one edit changed "...algorithm *will* not terminate..." to "...algorithm *does* not

[6] `http://code.google.com/p/google-diff-match-patch/` (accessed 2012-10-29)
[7] Unstructured Information Management System, `http://uima.apache.org/` (accessed 2012-10-29)

terminate...". One annotator labeled this edit as PARAPHRASE, the other one as INFORMATION-M and the third one as SPELLING/GRAMMAR. We observed this kind of total disagreement in 5.7% of all edits. The gold standard annotations have not been manually corrected subsequently.

Inter-annotator Agreement To estimate the reliability of the annotations, we compute the inter-annotator agreement per category using the multi-rater Kappa κ measure (Fleiss, 1971), see Table 5. For each edit, the proportion of agreeing votes (i.e. judgment pairs) out of the total number of pairs is calculated. With regard to the overall agreement, we need an appropriate agreement measure for multiple raters and multi-labeled edits. We employ Krippendorff's Alpha (Krippendorff, 1980) with a set-valued distance function, MASI (Passonneau, 2006). For each edit, we have a set of categories and consider the possibly partial agreement in the assigned category sets. The overall agreement in terms of Krippendorff's Alpha is $\alpha = 0.67$. This is at the lower boundary of what is usually considered to allow for drawing tentative conclusions (Krippendorff, 1980). To the best of our knowledge, no annotation study based on edit categories in Wikipedia has been carried out, hence, this value is hard to judge as we cannot compare it to other studies. We discuss the κ values across categories below (cf. Error Analysis).

Edit- vs. Revision-based Category Distribution To measure the absolute number of revisions labeled with a certain category C_r, we built the set of edit categories over all $e^k_{v-1,v}$ in each (r_{v-1}, r_v)-pair. When comparing the absolute number of edits labeled with a certain category C_e to C_r in Table 5, we observe that the MARKUP-D, SPELLING/GRAMMAR and PARAPHRASE categories have on average the highest number of edits per revision (more than two). All of them belong to the Surface layer, whereas many of the Text-Base edits (e.g. FILE, REFERENCE) show a lower ratio of edits per revision. This might be due to the fact that authors carrying out copy-edit changes have a focus on the entire article and change the text in various places which results in a higher number of edits. To the contrary, Text-Base edits may have a focus on a limited part of the article and hence edit in only one place. Furthermore, we could conclude that authors changing the article's text base save their edits more often, as this creates a higher number of revisions.

Single- vs. Multi-label Annotation Almost 15% of the edits are multi-labeled, and more than 30% of all revisions are multi-labeled. This shows that a lot of information would be lost if we opted against a multi-label annotation. The label cardinality, i.e. the average number of assigned categories per edit, cf. Tsoumakas et al. (2010), is $LC = \frac{1}{|D|}\sum_{i=1}^{|D|}|Y_i| = 1.2$ and the label density, i.e. the average fraction of assigned categories per edit, is $LD = \frac{1}{|D|}\sum_{i=1}^{|D|}\frac{|Y_i|}{|L|} = 0.06$, where D denotes our data set.

Error Analysis We turned the multi-labeled data into single-labeled data by transforming each unique category set which has been assigned to one of the edits into a new category $t \in T$. In our corpus, $|T| = 90$. Tsoumakas et al. (2010) refer to this transformation method as *Label Powerset*, as $T \subseteq P(L)$. We created and analyzed confusion matrices over the unique category sets for each annotator with respect to the gold standard. About 25% of all disagreement in terms of confused categories is due to edits which are labeled with the OTHER category in the gold standard. This is partly related to the fact that we labeled edits where all 3 annotators disagreed with the OTHER category in the gold standard. Furthermore, this category is not

Label	κ	P_O	Edits		Revisions		C_e			
			C_e	%	C_r	%	pre-FA	post-FA	pre-NFA	post-NFA
Information-I	0.64	0.91	280	11.67	200	13.11	71	59	81	69
Reference-I	0.79	0.95	262	10.92	209	13.70	59	37	87	79
Revert	0.83	0.96	254	10.59	128	8.39	66	55	50	83
Information-M	0.58	0.90	237	9.88	145	9.50	62	40	72	63
Markup-I	0.61	0.92	223	9.30	133	8.72	50	54	80	39
Vandalism	0.69	0.95	163	6.79	98	6.42	50	28	43	42
Spelling/Grammar	0.73	0.96	161	6.71	80	5.24	32	75	30	24
Information-D	0.55	0.93	139	5.79	80	5.24	54	32	22	31
Other[a]	0.18	0.97	139	5.79	86	5.64	42	36	26	35
Markup-D	0.58	0.95	131	5.46	59	3.87	22	60	23	26
Reference-D	0.68	0.97	88	3.67	66	4.33	35	6	24	23
Reference-M	0.54	0.96	88	3.67	78	5.11	24	8	30	26
Template-I	0.78	0.99	72	3.00	62	4.06	27	20	5	20
Paraphrase	0.31	0.96	54	2.25	24	1.57	6	12	7	29
Relocation	0.71	0.99	29	1.21	17	1.11	6	2	17	4
Template-D	0.66	0.99	26	1.08	20	1.31	13	5	1	7
Markup-M	0.25	0.97	17	0.71	13	0.85	8	2	6	1
Template-M	0.73	0.99	17	0.71	9	0.59	9	3	0	5
File-I	0.78	0.997	13	0.54	13	0.85	5	3	4	1
File-D	0.72	0.998	5	0.21	5	0.33	2	1	2	0
File-M	0.25	0.999	1	0.04	1	0.07	0	0	0	1
Text-Base	0.66	0.83	1228	51.19	888	58.19	361	214	328	325
Surface	0.61	0.83	615	25.64	326	21.36	124	205	163	123
Wikipedia Policy	0.79	0.93	417	17.38	226	14.81	116	83	93	125
All	—	—	2399	100	1526	100	643	538	610	608

[a]Excluded from top level categories. For that reason, percentages in the bottom rows do not sum up to 100%.

Table 5: Inter-annotator agreement, where κ is Fleiss' Kappa per category/layer and P_O the observed agreement per category/layer. C_e resp. C_r and % are the absolute numbers and percentages of edits resp. revisions labeled with a certain category in the gold standard.

well-defined. Further categories with low agreement are PARAPHRASE, FILE-M and MARKUP-M (cf. Table 5). FILE-M occurred only once in the gold standard. More than 40% of cases of disagreement involving MARKUP-M are labeled as OTHER in the gold standard, either due to segmentation errors (cf. Section 3.2), or because of general disagreement between all annotators. The PARAPHRASE category was not used consistently among the annotators and frequently confused with INFORMATION-M and SPELLING/GRAMMAR. Hence, the distinction between PARAPHRASE (non-meaning change) and INFORMATION-M (meaning change) has not been clear in many cases. For example, one edit replaced "several" with "many". Two annotators annotated this edit as PARAPHRASE, one as INFORMATION-M. A common problem in each of the categories was the distinction between insertions, modifications and deletions, particularly in the INFORMATION category. The annotators did not consequently adhere to the annotation guidelines (cf. Section 3.1) in some cases. If, for example, an edit *deletes* the word "not" in a phrase like "it is not a sacrament" (cf. Table 5), this edit also *changes* the meaning, which complicates the annotation of such edits.

One annotator labeled many instances of MARKUP-D as INFORMATION-D (9% of all cases of

disagreement with respect to the gold standard annotations). Furthermore, one annotator frequently (8%) forgot to multi-label MARKUP-I when larger portions of text were inserted (e.g. INFORMATION-I, REFERENCE-I instead of INFORMATION-I, REFERENCE-I, MARKUP-I).

For future work, we recommend to ignore edits labeled with the OTHER category. Categories with low agreement such as PARAPHRASE and MARKUP-M should be used with a grain of salt.

4 Discussion

4.1 Edit Category Distribution

The category distribution in our corpus partly corresponds with that in Pfeil et al. (2006) for the French, German, Japanese and Dutch Wikipedia, cf. Table 1. Additions of INFORMATION and REFERENCES are the most frequent categories.[8] VANDALISM in our corpus accounts for about 7% of all edits, which confirms the findings of Potthast (2010). Insertions clearly outnumber modifications and deletions, consistent with the studies of Jones (2008) and Pfeil et al. (2006). These findings confirm that our annotated corpus is a representative sample with regard to the collaborative writing process in Wikipedia.

Jones (2008) quotes only around 3% of edits in his *Add link* category, as compared to 28% in Pfeil et al. (2006) and 11% in the REFERENCE-I category in our corpus. Despite the fact that the categories might not fully overlap in their definitions, the low number in Jones's (2008) study could be an indicator that his approach to label edit categories based on the authors' comments does not fully capture the extent of certain edits.

The high deviation of absolute numbers of VANDALISM edits and REVERTS in our corpus is surprising. Manual inspection of the data shows that there are some Reverts of REVERTS (so called edit wars). Also, when comparing C_r to C_e for REVERT and VANDALISM in Table 5, apparently the number of edits per revision is much higher for REVERTS than for VANDALISM. This might be a particularity in our corpus, but we could also assume that vandals usually change a small portion of text, e.g. by inserting a swear word. On the other hand, authors applying a REVERT might not only revert vandalism but also undo legitimate edits which do not conform with their point of view.

4.2 Collaborative Writing and Quality

We designed WPQAC as a corpus to study differences in the quality of FAs and NFAs. To gain insights into the writing process, we analyzed the category distributions for different revision groups (cf. Table 4). Table 6 shows the Pearson correlations over category distributions between relevant groups. These calculations are based on the category frequencies of multi-labeled edits (Table 5, column C_e) for the revision groups.

Over all categories, we can see significant ($p < 0.01$, using Student's t-test) correlations between all of the groups, i.e. the frequencies of types of edits do not show significant differences among the revision groups. Generally, FAs and NFAs show a relatively high correlation. However, the correlation for pre-FA and post-FA revisions is clearly lower, as compared to pre-NFA and post-NFA. To reduce possible noise, we excluded the smaller categories from the groups and calculated the same correlations only for categories used to label at least 20 edits, i.e. with $C_e \geq 20$. As indicated in Table 6, the correlations between the pre-FA and post-FA as well as

[8] Ignoring Pfeil's (2006) *Format* category, which has partial overlap with our MARKUP category.

Group	r (all)	r (Top-16)	r (Jones, 2008)	Correlation criteria
All	0.87*	0.80*	0.91*	FA/NFA
All	0.90*	0.84*	—	pre/post
FA	0.72*	0.57	0.68	pre/post
NFA	0.87*	0.81*	—	pre/post
pre	0.86*	0.80*	—	FA/NFA
post	0.68*	0.52	—	FA/NFA

Table 6: Pearson correlation r between frequency distributions of edit categories by revision group for all and for the 16 largest categories. For comparison, we added the corresponding numbers for Jones's (2008) study. Values marked with * are statistically significant for $p < 0.01$.

post-FA and post-NFA are not statistically significant when calculated for the top 16 categories, i.e. we can assume that the two distributions come from different samples.

For the SPELLING/GRAMMAR and REFERENCE categories, deviances between the absolute number of edits in FAs and NFAs are particularly high (see Table 5). This is mainly because post-FA revisions show a higher number of SPELLING/GRAMMAR corrections and a lower number of REFERENCE edits as compared to pre-FA and NFAs. Improvements of style and grammar or spelling corrections are essential edits to produce thorough and high-quality content, hence, the higher number of this type of edits in post-FA revisions might be the result of the increased attention by experienced Wikipedia authors (Liu and Ram, 2011). The lower number of REFERENCE edits in post-FA revisions is not very surprising, as FAs need to be "well-researched", i.e. "verifiable against [...] reliable sources" according to Wikipedia's FA criteria[9] and we assume that this is the case for post-FA revisions. The high number of MARKUP-D edits in the post-FA revision group is due to one particular (r_{v-1}, r_v)-pair which deleted 42 markup tags in various places across the entire revision text.

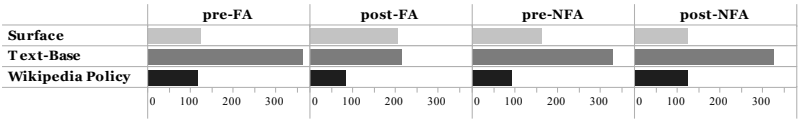


Figure 1: Absolute number of edits C_e for layers in revision groups.

It is not possible to verify the distinction between experienced and unexperienced authors as explained by Sommers (1980) for the collaborative writing process in Wikipedia. As can be seen in Table 5, the number of Surface respective Text-Base edits is higher respective lower for FAs compared to NFAs. This might be due to the fact that not only experienced authors work on FAs and vice versa.

The relationship between the distribution of edit types and quality has earlier been addressed by Jones (2008), who included in his corpus all FA revisions before and after their promotion. Like ours, his analysis shows a high correlation between FAs and NFAs, while pre-FA and post-FA differ significantly, cf. Table 6. Although it is hard to explain the reasons for this difference with his data, our corpus shows a clear difference in the ratio of Surface to Text-Base edits when

[9] http://en.wikipedia.org/w/index.php?title=Wikipedia:Featured_article_criteria&oldid=506642325

comparing post-FA revisions to pre-FA, pre-NFA and post-NFA revisions, cf. Table 5. Hence, even if we cannot find significant differences in the editing history of FAs and NFAs, there is a deviation in the collaborative writing process (in terms of editing behavior) before and after the promotion of FAs. The distinctive behavior of the post-FA revision group as compared to pre-FA and NFA revisions suggests that the nomination and promotion as FA triggers a distinguished type of collaboration. The collaborative writing process in post-FA revisions can be characterized through a relatively high number of surface edits (in particular, Spelling/Grammar corrections) and a low number of changes to the Text-Base. Figure 1 highlights the distinction between different revision groups. The lower number of Text-Base edits and the higher number of copy-edits in post-FA revisions can be interpreted as a sign of stability which FAs show after their promotion.

Conclusion

As explained in the above, there is a need for corpora to analyze the collaborative writing process. To address this problem, we introduced a classification scheme of edits established on previous work of the writing research. We applied this scheme to annotate a sample from the revision history of the English Wikipedia. To verify the reliability of our annotations, we measured and analyzed the inter-annotator agreement across categories. We published our corpus, providing free access to the research community. Furthermore, we compared the edit category distribution in featured and non-featured article revisions. Our findings show that featured articles differ from non-featured articles mainly because of a distinguished process of collaboration after an article achieved featured status. This collaboration process includes a higher number of surface changes and on the opposite a lower number of edits changing the meaning.

Further work should incorporate a deeper analysis of article quality and quality flaws in Wikipedia (Ferschke et al., 2012b). Since revisions in Wikipedia are accompanied by metadata and in particular, user comments, an analysis of the metadata based on edit categories might yield interesting results. Although we analyzed edit categories in the English Wikipedia, our approach (i.e. the classification scheme and the edit segmentation) can be applied to any language version of Wikipedia. Given that other language versions might use existing information in the English Wikipedia and translate it rather than creating completely new content (e.g. to keep the language versions with a smaller set of authors up-to-date), our taxonomy can also be used to distinguish between surface edits and edits which add new information, similar to the approach of Bronner and Monz (2012).

Finally, there remains a need for more data. Our assumptions have to be confirmed on a larger corpus. We will address this issue by augmenting the labeled corpus with an automated approach using Machine Learning on the annotated data.

Acknowledgments

This work has been supported by the Volkswagen Foundation as part of the Lichtenberg-Professorship Program under grant No. I/82806, and by the Hessian research excellence program “Landes-Offensive zur Entwicklung Wissenschaftlich-ökonomischer Exzellenz” (LOEWE) as part of the research center “Digital Humanities”. We thank the anonymous reviewers for their helpful remarks.

References

Adler, B. T., Alfaro, L., Mola-Velasco, S. M., Rosso, P., and West, A. G. (2011). Wikipedia Vandalism Detection: Combining Natural Language, Metadata, and Reputation Features. In Gelbukh, A., editor, *Computational Linguistics and Intelligent Text Processing*, Lecture Notes in Computer Science, pages 277–288. Springer.

Bronner, A. and Monz, C. (2012). User Edits Classification Using Document Revision Histories. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 356–366, Avignon, France. Association for Computational Linguistics.

Chin, S.-C., Street, W. N., Srinivasan, P., and Eichmann, D. (2010). Detecting Wikipedia vandalism with active learning and statistical language models. In *Proceedings of the 4th workshop on Information credibility*, pages 3–10, Raleigh, NC, USA.

Ede, L. and Lunsford, A. (1990). *Singular Text/Plural Authors: Perspectives on Collaborative Writing*. Southern Illinois University Press.

Faigley, L. and Witte, S. (1981). Analyzing Revision. *College Composition and Communication*, 32(4):400.

Ferschke, O., Daxenberger, J., and Gurevych, I. (2012a). A Survey of NLP Methods and Resources for Analyzing the Collaborative Writing Process in Wikipedia. In Gurevych, I. and Kim, J., editors, *The People's Web Meets NLP: Collaboratively Constructed Language Resources*, Theory and Applications of Natural Language Processing, chapter 5. Springer, Heidelberg.

Ferschke, O., Gurevych, I., and Rittberger, M. (2012b). FlawFinder: A Modular System for Predicting Quality Flaws in Wikipedia. In *CLEF 2012 Evaluation Labs and Workshop – Working Notes Papers*, Rome, Italy.

Ferschke, O., Zesch, T., and Gurevych, I. (2011). Wikipedia Revision Toolkit: Efficiently Accessing Wikipedia's Edit History. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies. System Demonstrations*, pages 97–102, Portland, OR, USA.

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

Flöck, F., Vrandečić, D., and Simperl, E. (2012). Revisiting reverts: Accurate revert detection in Wikipedia. In *Proceedings of the 23rd ACM conference on Hypertext and social media*, pages 3–12, Milwaukee, WI, USA.

Fong, P. K.-F. and Biuk-Aghai, R. P. (2010). What did they do? Deriving high-level edit histories in Wikis. In *Proceedings of the 6th International Symposium on Wikis and Open Collaboration*, WikiSym '10, Gdansk, Poland.

Heckel, P. (1978). A technique for isolating differences between files. *Communications of the ACM*, 21(4):264–268.

Jones, J. (2008). Patterns of Revision in Online Writing: A Study of Wikipedia's Featured Articles. *Written Communication*, 25(2):262–289.

Kittur, A. and Kraut, R. E. (2008). Harnessing the wisdom of crowds in wikipedia: quality through coordination. In *Proceedings of the 2008 ACM Conference on Computer Supported Cooperative Work*, CSCW '08, pages 37–46, San Diego, CA, USA.

Krippendorff, K. (1980). *Content Analysis: An Introduction to Its Methodology*. Sage Publications.

Lee, J. and Webster, J. (2012). A Corpus of Textual Revisions in Second Language Writing. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics*, pages 248–252, Jeju Island, Republic of Korea.

Liu, J. and Ram, S. (2011). Who does what: Collaboration patterns in the wikipedia and their impact on article quality. *ACM Transactions on Management Information Systems*, 2(2).

Noel, S. and Robert, J.-M. (2004). Empirical Study on Collaborative Writing: What Do Co-authors Do, Use, and Like? *Computer Supported Cooperative Work*, 13(1):63–89.

Passonneau, R. (2006). Measuring Agreement on Set-valued Items (MASI) for Semantic and Pragmatic Annotation. In *Proceedings of the 5th International Conference on Language Resources and Evaluation*, Genoa, Italy.

Pfeil, U., Zaphiris, P., and Ang, C. S. (2006). Cultural Differences in Collaborative Authoring of Wikipedia. *Journal of Computer-Mediated Communication*, 12(1):88–113.

Potthast, M. (2010). Crowdsourcing a Wikipedia Vandalism Corpus. In *Proceedings of the 33rd Annual International ACM SIGIR Conference on Research & Development on Information Retrieval*, pages 789–790, Geneva, Switzerland.

Sommers, N. (1980). Revision strategies of student writers and experienced adult writers. *College composition and communication*, 31(4):378–388.

Stvilia, B., Twidale, M. B., Smith, L. C., and Gasser, L. (2008). Information quality work organization in wikipedia. *Journal of the American Society for Information Science and Technology*, 59(6):983–1001.

Tsoumakas, G., Katakis, I., and Vlahavas, I. (2010). Mining Multi-label Data. In Maimon, O. and Rokach, L., editors, *Data Mining and Knowledge Discovery Handbook*, chapter 34, pages 667–685. Springer.

Woodsend, K. and Lapata, M. (2011). Learning to Simplify Sentences with Quasi-Synchronous Grammar and Integer Programming. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 409–420, Edinburgh, Scotland, UK.

Zesch, T. (2012). Measuring Contextual Fitness Using Error Contexts Extracted from the Wikipedia Revision History. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 529–538, Avignon, France.

A Computational Cognitive Model for Semantic Sub-network Extraction from Natural Language Queries

Suman DEB ROY Wenjun ZENG

Department of Computer Science
University of Missouri – Columbia, USA.

sdr5x8@mail.missouri.edu, zengw@missouri.edu

ABSTRACT

Semantic query sub-network is the representation of a natural language query as a graph of semantically connected words. Such sub-networks can be identified as sub-graphs in larger ontologies like DBpedia or Google knowledge graph, which allows for domain and concepts identification, especially in noisy queries. In this paper, we present a novel standalone NLP technique that leverages the cognitive psychology notion of semantic *forms* for semantic sub-network extraction from natural language queries. Semantic *forms*, borrowed from cognitive psychology models, are one of the fundamental structures employed by human cognition to construct semantic information in the brain. We propose a computational cognitive model by means of conditional random fields and explore the interaction patterns among such *forms*. Our results suggest that the cognitive abstraction provided by semantic *forms* during labelling can significantly improve parsing and sub-network extraction compared to pure lexical approaches like parts of speech tagging. We conduct experiments on approximately 5000 queries from three diverse datasets to demonstrate the robustness and efficiency of the proposed approach.

KEYWORDS: Cognitive Linguistics, Forms, Query, Search, Subnets, Semantic

Proceedings of COLING 2012: Technical Papers, pages 727–744,
COLING 2012, Mumbai, December 2012.

1 Introduction

The efficiency of natural language (NL) search often depends on detection of keywords in a query, followed by construction of some meaningful connected network comprising of such keywords (Herdagdelen, 2010). This connected network of keywords is called a semantic subnet (Booth, 2009). These keywords together comprise what is called a semantic field (Croft, 2003). The goal of our research is to efficiently recover the semantic sub-network from NL queries.

Prior research suggests three main motivations for extracting semantic subnets from NL queries. Firstly, extracted query subnets can be used to generate a candidate set of concepts within a larger ontology (like of DBpedia RDF network/ Google knowledge graph), which may align to the words in the query subnet and assist domain identification and query expansion (Booth, 2009). Secondly, a query subnet can act as a NL interface to concept graph databases (Popescu, 2003), facilitating semantic information retrieval (Kauffman, 2007) and improved query understanding and semantic search (Hu, 2009). Finally, semantic subnets enable identification of event structures within sentences (McClosky, 2011) and assist higher-level NLP tasks, like Question Answering (QA) (Huang, 2009).

It is possible to detect semantic keywords in NL queries using methods like Named Entity Recognition (NER) or Semantic Role Labelling (SRL) (Collobert, 2011). Both techniques provide a higher level of abstraction than the basic syntax tree. However, our task goes a step further: we aim to find out *how these keywords are semantically connected* in terms of a network. This is very difficult to achieve using NER alone, since detecting the named entities provides limited information about their relations. SRL does a better job at concept level parsing using predicate logic, but is bound by the strict predicate grammar. Therefore, although techniques such as NER and SRL is core to NLP, there is an inherent gap between requirements of intelligent tasks (like QA) and several state-of-the-art NLP techniques (Finkel, 2009).

As search is becoming more collaborative and social, queries turn noisier (Hu, 2009). Often, the conceptual structure of the NL query is difficult to extract using simple (Parts-Of-Speech) POS-based dependency parsing. Imprecision of NL usage is a major obstacle to computation with NL. Therefore, it is necessary to develop a technique that partially relaxes the rigid grammar of the language. While imprecise or varied grammatical constructions are difficult to capture using POS or predicate logic, note that the human cognition can often eliminate such noise to interpret meaning. If we assume that 'meaning' of a NL sentence is captured in its semantic subnet, then it would be logical to conclude that human cognition possesses a more noise-resistant process of extracting semantic subnets. A rational explanation for this is the presence of an improved model for detecting semantics in NL and subsequently constructing semantic information in the brain.

Cognitive psychology has a number of interesting theories on how the human mind deals with imprecision, uncertainty and complexity of language (Chater, 2006). One such theory, called the structure-of-intellect model, proposes that humans perceive concepts contained within the words of a sentence as a semantic *form* (Guilford, 1977). His model has been widely used to study the cognitive intellect and the kinds of information that humans can extract from any observed

argues that human cognition is robust to noise because it dynamically changes the resolution at which data is to be semantically interpreted (Croft, 2004).

Recognizing the potential of cognitive approaches in semantic information modelling, we propose to leverage semantic *forms* in the extraction of semantic sub-networks from NL queries. These semantic *forms*, when connected in some networked pattern, becomes responsible for understanding the scope and context of a concept, and assists functional retrieval of related concepts and question answering/response (Carroll, 1993). Thus, our main insight in modelling semantic *forms* and their interaction patterns in NL is grounded on the idea: *the subsurface form space demonstrates the query intent (expresses semantics) better than superficial (lower) query syntactical features, which might vary depending on diverse query construction.* In other words, the higher is the level of abstraction for labelling, the more robust the extraction should become. This idea of cognitive abstraction provided by semantic *forms* is shown in Fig. 1.

The main contributions of this paper are:

- We propose the use of semantic *forms*, borrowed from cognitive science, as label category for NL sequence labelling tasks.
- We propose a conditional random field based method of implementing the structure of intellect model, by labelling query words with semantic *forms* and analyzing the interconnected patterns in which such *forms* exist within a semantic field.
- We perform experiments on three diverse query datasets consisting of TREC, QA-type and Web queries to justify the robustness of our approach to varying noise levels. Our approach comprehensively outperforms existing works on query subnet detection (Booth, 2009).

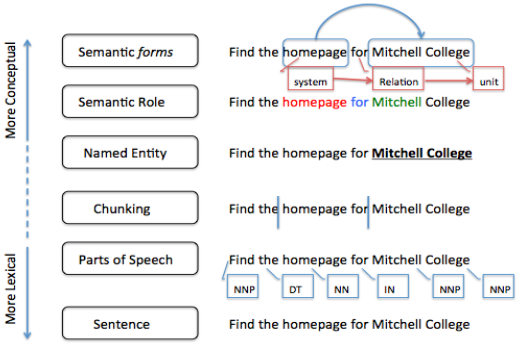


FIGURE 1 – Level of abstraction in different NLP techniques: from lexical to conceptual.

The rest of this paper is organized as follows: in Section 2 we discuss related work and the scope of the paper. Section 3 introduces the notion of semantic *forms* and their interactions. In Section 4, we describe the proposed model for labelling query words with linked semantic *forms*. Section

aids better NL interfaces that precisely map a NL query into a suitable graph database. In the next two sub-sections, we describe the related work and scope of this paper.

2.1 Related Work

Semantic subnet extraction from NL queries is essentially a method of query reformulation, wherein a query is represented in an alternative form that eases its interface with different types of databases for concept detection. A detailed analysis of query reformulation techniques is available in (Clifford, 1990) and (Herdagdelen, 2010). Substantial efforts have been exhausted in trying to enhance the role of NL interfaces in converting a natural language query into a graph database query (Booth, 2009) (Popescu, 2003). A semantic network of Resource Description Format (RDF) concepts (DBpedia) can be considered one such graph database (Auer, 2007). Several problems like word-sense disambiguation (Bruce, 1994), specificity of grammar (Manning, 1999) and keyword (not semantic) based approaches inhibit portability of several existing NLP techniques across systems and domains (Kaufmann, 2007). The closest work to our research is (Booth, 2009), which uses a POS-based approach in extracting subnets from queries. The accuracy of query subnet extraction compared to a human standard can be evaluated using metrics such as Consistency Index (Cardona, 2009). The results stated in (Booth, 2009) are tested on a very limited number of queries (approx. 12), which does not come close to capturing the diversity in human query constructions or web scale. In contrast, we provide empirical results on 5000 queries from three query datasets with different noise levels.

Substantial efforts have also been spent in detecting named entities in sentences. This task, called Named Entity Recognition (NER) seeks to locate and classify words such as names, organizations, places etc. in a NL sentence (Guo, 2009). A similar NLP task is SRL, which involves finding target verbs (predicate) in a sentence that resemble some 'action' (Collobert, 2011). Models have also strived to combine these two techniques (Finkel, 2009). In this paper, we will use Conditional Random Fields (CRF) (Lafferty, 2001) to capture the cognitive model in terms of finite state automata, with each *form* state dependent on the previous *form* state and on the current symbol word being processed. The resulting Markov chain of possible states can be solved using the Viterbi Algorithm, implemented using dynamic programming (Manning, 1999).

J. P. Guilford introduced the structure-of-intellect model in (Guilford, 1977), which covers the notion of semantic *forms* as 'products'. 'Products' are the result of applying some cognitive operation (cognition, retention etc.) on specific content (semantic, symbolic etc.). The model has since been used, studied and analysed substantially in the cognitive science community. A detailed view of human cognitive semantics in linguistics is provided in (Croft, 2004). Probabilistic models of cognitive linguistics are described in (Chater, 2006). An insightful introduction to human cognitive abilities is available in (Carroll, 1993).

2.2 Motivation

We strive to better model the conceptual linkage among words in a query sentence, such that the linked semantic field (query subnet) can be searched for in a larger network of RDF based

We abstract the problem to the level of cognitive semantics, by making use of the concept of semantic *forms*. According to existing cognitive psychology, *forms* are used by the human psyche to process and store semantic information structures in the brain (Carroll, 1993). *To the best of our knowledge, computationally modelling semantic forms borrowed from the domain of cognitive psychology has not been previously used in semantic query understanding in the domain of natural language processing*.

3 The Cognitive Structure-of-Intellect Model

The main hypothesis proposed by the Structure-of-Intellect model is that *human cognition is robust to noisy sentence constructions because it strives to detect semantic information at different levels of granularity*. The noisier the sentence, the coarser is the granularity of semantic information detection employed by the human cognition. In this section, we qualitatively introduce the different semantic *forms* from Guildford's structure-of-intellect cognitive model and describe how *form* interaction patterns play a key role in semantic subnet extraction.

3.1 Granular Hierarchies in Semantic Information

Semantic *forms* consist of five entities that capture the structure of information contained within a natural language sentence as perceived by the human cognition. A remarkable thing about semantic *forms* is that they are structured as granular hierarchies (i.e. one *form* is composed of other *forms*). Following is a description of the semantic *forms* starting with finer granularity:

Unit: Every item of a query sentence can be regarded as part of some chunk, of which *units* are the most basic entities. *Units* will cover most words of a sentence, from intangible ideas like 'love' to tangible objects like 'cars'. For example, the name 'Anna Chakvetadze' is a *unit*. The cognition of semantic *units* has to do with one's vocabulary (Guilford, 1977).

Class: When *units* have one or more attributes in common, they can be grouped in *classes*. *Units* belonging to a *class* will share connectivity to at least one common attribute node. *Classes* can be narrow or broad. For example, the *unit* 'Anna Chakvetadze' can belong to the very broad *class* 'female', a moderately broad *class* 'Russia' or a narrow *class* 'Tennis'. The size of the *class* (narrow/ broad) qualitatively determines the size of the search space for related concept retrieval.

Relation: *Relations* are *kinds* of connections between *units*. When any two entities are connected in the semantic network, there are three items of information involved – two *units* and the *relation* between them. *Relations* between search keywords play an integral role in realizing *class* or *unit* interconnections in the query. For example, 'Steffi Graf' and 'Andre Agassi' could be connected by the *relation*: *married*, while both belonging to the *class: tennis players*.

System: A *system* is the most complex item in semantic information. *Systems* are composed of more than two interconnected *units*. *Systems* may also comprise of overlapping *classes*, multiple interconnecting *units* and diverse *relations*. They often occur as an order or sequence of *units*. Add 'Maria Sharapova' and 'Sasha Vujacic' to the previous example of 'Steffi Graf' and 'Andre Agassi', and we get a *system*: *married sportspersons*.

Word	Thursday	witches	market	driving	mansion	school
Form	*unit*	*class*	*system*	*relation*	*Unit*	*system*

TABLE 1 – Examples of *forms* attached to query words.

4 The Proposed Computational Cognitive model

In our proposed approach, consider each observed symbol as the tuple: {word, POS tag, NP chunk number}. We can employ basic sequence labeling idea here, by considering the chain of *forms* that link the tuples as hidden states. Using the training data, a CRF model (McCallum, 2000) can then assign optimal state chains to samples of observed symbols, from which we learn the kinds of *form* chains (interactions) that exist. Steps for computationally modeling the cognitive notion of semantic *forms* are described in this section.

We begin with formal definitions, followed by describing some pre-processing techniques and finally, the detailed description of model features.

4.1 Formal Definitions

Consider an NL sentence *Q*. Our assumption is that *Q* is a carrier of information. Every word is a linguistic variable in *Q*. It is well known that information is expressible as a restriction (i.e. a constraint) on the values that a variable can take (Zadeh, 1998). By this flow of thought, consider *W* as a constrained variable in *Q*, let *R* be the constraining relation in *Q* and ζ (zeta) represent how *R* constrains W. Then, every NL sentence *Q* can be represented as: $Q \rightarrow W\ \zeta\ R$

It is possible for *W* to be a vector-valued random variable. The primary constraint *R* is a restriction on the *form* values that can be probabilistically assigned to *W*. Hence, *W* can take up values of different *forms* from the set (*unit, class, … , transform*) with probabilities (p_{unit}, p_{class}, …, $p_{transform}$) respectively. Thus, *W* is constrained using the probability distribution *R* as:

$$W\ \zeta\ (p_{unit} \backslash unit + \ p_{class} \backslash class + \cdots + p_{transform} \backslash transform)$$

The singular variable *W* takes values from the universe of discourse *U*, such that values of *W* are singletons in *U*. On the other hand, the semantic *form* of *W* is a variable whose values depend on the granular collections in *U*. Said alternately; *the granular precision of a word in U is expressed through its semantic form*. The type of *form* assigned to a word depends on the cluster size of elements in *U* that have common attributes or behaviour related with the concepts of that word.

The overall process is described at an abstract level in Fig. 2, where ellipses represent the *form* of a word. Consider four key words W1, W2, W3, and W4 in the query (*Q*) that need to be connected as some semantic subnet. Let $Form(W1)$ denote the *form* associated with the word *W1*. In step (i): we are uncertain of the semantic subnet connection among the words. In (ii), our goal is to label the words with semantic *forms* to help extract the query subnet. In (iii), we use the *form* interconnection patterns (described in Section 4.3.3.2) to retrieve the connection among the *forms* for the four words when they exist together in some *Q*. Finally, in (iv), we can connect the words as a query subnet by shadowing the connected *form* pattern that exists among the *forms*.

known Python NL toolkit stop word list. We used the Stanford POS tagger for POS tagging. For long queries, chunking is necessary. The chunking process is inspired by (Huang, 2009).

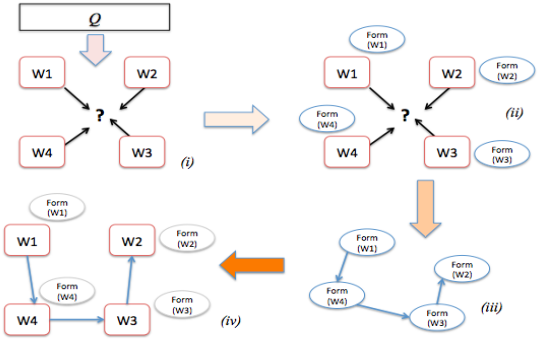


FIGURE 2 – Overview of process flow from receiving input query *Q* to subnet extraction.

Chunking: Consider *Q* to be a query sentence in natural language *L* containing words belonging to the vocabulary set *V*. Let S_Q be the sequence of POS-tagged symbols associated with *Q*, i.e.

$S_Q = \{s_1, s_2, \dots, s_N\}$, where $s_n = \langle w_n, t_n \rangle$, $w_n \in V$, $t_n \in T'$ for *N* words in Q.

Given S_Q we can define the k^{th} chunk ($\mathcal{C}_k$) as: $\mathcal{C}_k = (\langle w_i, t_i \rangle, \langle w_{i+1}, t_{i+1} \rangle, \dots, \langle w_j, t_j \rangle)$ for some $i < j \leq N$ and $1 \leq k \leq M$ for a total of *M* chunks in the query. We assume that no two chunks have any common words, i.e. chunks are non-overlapping. Then, the task involves determining all the *M* chunks based on S_Q, s.t. $\mathcal{C} = \{\mathcal{C}_1, \mathcal{C}_2, \dots \mathcal{C}_k, \dots, \mathcal{C}_M\}$.

This generates the chunked query set: $S_{QC} = \{\langle s_1, \mathcal{C}_1 \rangle, \dots, \langle s_l, \mathcal{C}_1 \rangle, \langle s_{l+1}, \mathcal{C}_2 \rangle, \dots, \langle s_N, \mathcal{C}_M \rangle\}$, where $\mathcal{C}_1 = (s_1, \dots, s_l)$, $s_l \in S_Q$, for some l, $1 \leq l \leq N$. Following similar methods as used in (Huang, 2009) and given $\{s_{i-1}, s_i, s_{i+1}\}$, we can find $p(c_1, c_2, \dots, c_N \mid S_Q)$ as:

$$p(c_1, c_2, \dots, c_N \mid S_Q) = \Psi \prod_{i=1}^{N} p(s_{i-1} | s_i, c_i)\, p(s_{i+1} | s_i, c_i)\, p(s_i | c_i)\, p(c_i) \qquad (1)$$

$$\Psi = 1 / p(s_1, s_2)\, p(s_2, s_3) \dots, p(s_{N-1}, s_N)$$

and c_i represents if s_i is inside, outside or start of some NP chunk. The individual probabilities of Eq. (1) can be estimated from the training set.

4.3 *Form* Tagging Using CRFs

The task of tagging words of a sentence with semantic *forms* from the set of *forms* (*F*) leverages a CRF model. The result is the set S_{QF} of *form* labelled words. First, we briefly describe CRF in the light of our problem, followed by feature functions and learning weights.

A *state feature* is an element of $\boldsymbol{l_f}$ of the structure $state(y, \boldsymbol{x}, i)$ where *i* is the input position, *y* is a label and $\boldsymbol{x}$ is the input sequence. A *transition feature* is an element of l_f of the structure $tran(y, y', \boldsymbol{x}, i)$ where *y*, y' are labels.

The global feature vector for an input sequence $\boldsymbol{x}$ and a label sequence $\boldsymbol{y}$ is:

$$\boldsymbol{F}(\boldsymbol{y}, \boldsymbol{x}) = \sum_i \boldsymbol{f}(\boldsymbol{y}, \boldsymbol{x}, i) \quad (2)$$

Individual feature functions are described in Section 4.3.2. A conditional distribution that obeys the Markov property, which is: $p\left(Y_i \middle| \{Y_j\}_{j \neq i}, \boldsymbol{X}\right) = p(Y_i | Y_{i-1}, Y_{i+1}, \boldsymbol{X})$ can be written as:

$$p_{\boldsymbol{\lambda}}(\boldsymbol{Y}|\boldsymbol{X}) = \frac{\exp\{\boldsymbol{\lambda}.\boldsymbol{F}(\boldsymbol{Y}, \boldsymbol{X})\}}{Z_{\boldsymbol{\lambda}}(\boldsymbol{X})} \quad (3)$$

where $Z_{\boldsymbol{\lambda}}(\boldsymbol{x}) = \sum_{\boldsymbol{y}} \exp\{\boldsymbol{\lambda} . \boldsymbol{F}(\boldsymbol{y}, \boldsymbol{x})\}$.

Note the denominator of Eq. (3) is independent of $\boldsymbol{y}$. Then the most probable sequence of *form* labels ***(y*)*** for the input sequence $\boldsymbol{x}$ is:

$$\boldsymbol{y}^* = \arg max_{\boldsymbol{y}}\, p_{\boldsymbol{\lambda}}(\boldsymbol{y}|\boldsymbol{x}) = \arg max_{\boldsymbol{y}}\, \boldsymbol{\lambda}.\boldsymbol{F}(\boldsymbol{y}, \boldsymbol{x}) \quad (4)$$

Eq. (4) can be solved using the Viterbi Algorithm (McCallum, 2000).

4.3.2 Feature Functions

Feature functions are key components of CRF (see Fig. 3). The general structure of a feature function is $z(f_{i-1}, f_i, \boldsymbol{x}, i)$ which looks at two adjacent states f_{i-1}, f_i, the whole input sequence $\boldsymbol{x}$ where *i* is the current location in this sequence, and assigns some weight. They can be defined in different ways, e.g., we have a feature like: if the current word is 'Nile' and the current state is '*unit*' then we give the feature a positive weight, otherwise not. Each feature function has a binary output and can take as inputs the value of a feature and particular values of the current *form* f_i and the previous *form* f_{i-1}. We use the training corpus queries to build the atomic feature set for the CRF. Let F_u, F_r, F_c, F_s represent *unit, relation, class* and *system respectively*.

In the examples below, a binary value of 1 indicates the presence of the feature, and 0 the lack of the feature. 'λ' denotes logical AND. We implement four types of binary atomic features:

(1) *Simple Feature Function*: A simple feature function depends only on a word and its connected *form*. For example,

$$z(f_{i-1}, f_i, \boldsymbol{x}, i) = \begin{cases} 1, & if\ x_i = ´\ music´ \wedge f_i = F_s \\ 0, & otherwise \end{cases}$$

(2) *Overlapping Feature Function*: An overlapping feature function depends on *form* of a word and on its successor word. Under normal conditions, Hidden Markov Models (HMMs) are unable to realize overlapping features (unlike CRFs). A suitable example would be:

$$z(f_{i-1}, f_i, \boldsymbol{x}, i) = \begin{cases} 1, & if\ x_i = ´\ and´ \wedge f_{i-1} \neq F_r \\ 0, & otherwise \end{cases}$$

734

(3) *Form Transition Feature Function*: A *form* transition feature function depends on successive *forms* such as:

$$z(f_{i-1}, f_i, \boldsymbol{x}, i) = \begin{cases} 1, & if\ f_{i-1} = F_r \wedge f_i = \{F_u, F_c\} \end{cases}$$

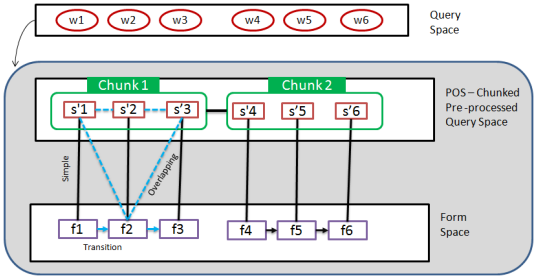


FIGURE 3 – Different features functions in the CRF model: from query words to *form* labeling.

In Fig. 3, each '*s*' element in the POS-chunked pre-processed query space represents a tuple <word, POS, NP Chunk number>. There are 828 atomic features in our system, obtained from words in the vocabulary and shifted-conjugation patterns.

This initial feature set is then grown using feature induction (McCallum, 2003), resulting in a total of 23,713 features. A quasi-Newton method is used to adjust all parameters of the CRF model to increase the conditional likelihood. When training the CRF, we use pre-conditioning to ensure fast convergence of the conjugate gradient method (Sha, 2003). On average, our technique requires 12-13 forward-backward iterations to reach an objective function value, which is in close proximity (~96%) to the maximum.

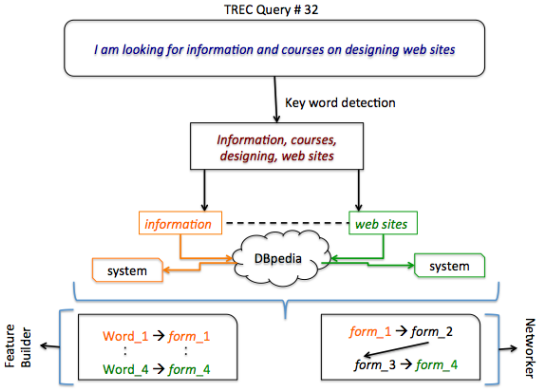


FIGURE 4 – Feature Builder and Networker learning from training queries.

4.3.3 Feature Generation

4.3.3.1 State features

The CRF labeller's state feature set is assorted using a feature builder. The feature builder is trained using a seed set of words and their related *forms* obtained using DBpedia RDF resource (Fig. 4). DBpedia contains structured information collected from Wikipedia and has been extensively used for semantic analysis of content (Auer, 2007). The RDF semantic network of DBpedia is arranged in granular categories. Thus, for every node (which represents a concept word), we can calculate the normalized degree centrality, which gives us an estimate of the generality of the node. The more general concept nodes have higher centralities (Coursey, 2009).

Fig. 4 illustrates how words and their tagged *forms* are collected from a training query. Keywords are identified from a query by stemming and eliminating stop words. Using DBpedia to classify the word into a semantic *form* follows this. The vocabulary containing words → *forms* is updated as more training examples are seen and used by the feature builder.

4.3.3.2 Transition features

Given enough training samples of the sentence Q, the variable W and constraint R, we can deduce the pattern ζ, which identifies how R constrains W. This pattern ζ contains information about the ordering in Q with respect to W (recall W could be vector-valued) such that they are mapped to R. That is to say, every *form* has some specific interaction pattern with other *forms* when they exist together/adjacent in Q. This interaction pattern among *forms* in U is signified by ζ. Interaction patterns provide insight into the question: how are three or more *forms* connected when appearing in Q? For example, if we see words $\{A, B, C\}$ having *forms* {*relation, class, unit}* respectively, then would the query subnet be of the ordering A-B-C, B-A-C or C-A-B?

The networker learns interaction patterns at the chunk level, modelling each chunk as a potential branch for a rooted semantic query subnet. This viewpoint is derived from the observation that branches of most annotated query subnets are composed of individual or contiguous chunks of the original query. The *form* interaction set $\tilde{F}$ is a simple ordered set: $\{(f_i, f_j)\}$, where $f_i, f_j \in F$ representing a complete or part of a directed chain $f_i \rightarrow f_j$. We only use *forms* connected within a chunk to populate the set $\tilde{F}$. Fig. 5 shows results collected using a Trellis diagram.

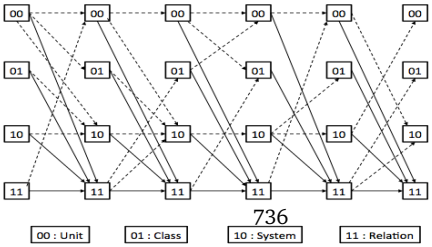


FIGURE 5 – Trellis diagram of possible Viterbi paths representing sequence of labeled *forms*.

measure: $z(f_{i-1}, f_i, \boldsymbol{x}, i) = 1\ if\ f_i = b\ and\ f_{i-1} = a$ and assign weights to this feature based on *K* training subnet samples as:

$$w_{a,b} = 1 + \log[\ \sum_{1 \le k \le K} n(a,b) * |k|/K\ ] \quad (5)$$

where *k* is the *kth* sample in the training set of subnets, *n(a,b)* = 1 if the *form* transition $a \rightarrow b$ appears as some edge of the *kth* sample subnet and $|k|$ is the length of the subnet branch containing $a \rightarrow b$. If the *form* transition $a \rightarrow b$ is not present, then *n(a,b)* = 0.

Eq. (5) achieves a simple goal: it takes the human labelled subnets into consideration while allocating weights for transition feature functions. The more we notice a particular *form* transition repeated, the higher the weight it is given as a potential transition feature.

From the *form* interaction patterns $\tilde{F}$ and the chunk set $\mathcal{C}$, the networker builds a set $S_{QF} = \{\langle s_1, f_1, \rangle, \dots, \langle s_3, f_3, \rangle, \dots, \langle s_6, f_6, \rangle\}$ structured as a tree $G = (S_{QF}, E)$ where $E \in \tilde{F}$. If $|\mathcal{C}_i|$ represents the number of POS-tagged symbols in some chunk $\mathcal{C}_i \in \mathcal{C}$, then the height of subnet is $\le \max(|\mathcal{C}_i|)$, taking into account that consecutive *units* within a chunk may be collapsed into a single node. G is the semantic subnet.

5 Experiments

5.1 Data Description, Evaluation Metrics and Benchmarks

Data: We test our model on each of these three datasets: (a) The TREC 2011 (TREC) web topic dataset has 50 topics (Clarke, 2011). Each topic has 1-7 queries associated with it. All queries within a topic resemble similar search intent. There are a total of 243 queries in the TREC topic dataset. 77% of the queries in the TREC dataset have 11-14 words. (b) The Microsoft Question Answering Corpus (MSQA), which is aimed at querying documents belonging to the Encarta-98 encyclopedia (MSQA, 2008). There are 1365 usable queries in this dataset and 85% of the queries have 5-10 words. (c) The last dataset consists of ~ 3400 raw search query feeds collected from a commercial web search engine (denoted as 'WSE'). Queries containing 4-20 words are chosen for evaluation. The distribution of average number of words per query is shown in Fig. 6.

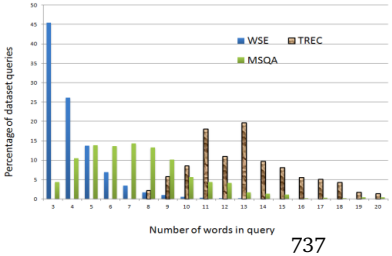


FIGURE 6 – Distribution of avg. number of words per query in the three datasets.

The three datasets represents gradually rising levels of challenge in terms of query construction

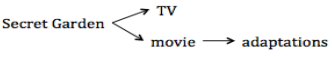
Query ID	Query Sentence	Query Subnet
TREC 43	*Find reviews of the various TV and movie adaptations of The Secret Garden*	

TABLE 2 – Example of query subnet generated from query.

Several other small optimizations are implemented: (a) we collapse consecutive *units* into a single *unit* when creating the subnet. (b) We use a simple root selection algorithm: when only *relation* words are found connecting two chunks $\mathcal{C}_i, \mathcal{C}_j \in \mathcal{C}$, we search $\mathcal{C}_{j+1}$ for *units* or *classes*. If $\mathcal{C}_{j+1}$ lacks a *unit* or *class*, we search $\mathcal{C}_{i-1}$instead. For example, in the query #TREC43 (Table 2), '*various TV'* and '*movie adaptations'* are connected by the conjunction '*and'*. Therefore, we search in $\mathcal{C}_{j+1} = '\ of\ The\ Secret\ Garden'$ and since we find a sequence of two *units* ('Secret', 'Garden'), we collapse it to a single *unit* and represent it as root.

We used annotators to hand label the queries in the datasets to build query subnet trees. The inter-annotator agreement on subnet structure was 72.3%. Disagreements were limited to just 1 node position in 82% disagreed cases. Thus, we consider this hand labelled set as the gold standard for comparing the machine generated subnet.

Metrics: Since our output (query subnet) is a tree where each node belongs to the set of query words, a 'tree-likeness' metric is essential to judge quality of results produced in terms of *structure*. We use *Consistency Index* (*CI*) as a metric to judge the quality of the subnet generated (Cardona, 2009). Mathematically, *CI* can be defined as:

$$CI = node\ position\ matches\ between\ T1\ \ and\ T2/\#\ nodes\ in\ T2$$

where, *T1* represents the query subnet tree generated by a machine algorithm, *T2* is the query subnet tree of the gold standard and # represents the number of nodes. In (Booth, 2009), the authors evaluate their subnets using simple measures like 'nodes correctly resolved' or 'semi-correctly resolved'. However, we believe that *CI* captures the effect of *structural relatedness* more intuitively. Table 3 lists the average *CI* values obtained for various datasets for the proposed approach and the comparison benchmarks.

Benchmarks: We compare the proposed model (*formNet*) against 3 benchmarks. We test our model against (a) the POS based approach introduced in (Booth, 2009) for generating subnets from query sentences (called *posNet*), (b) a non-*form* CRF (denoted as *nfCRF*) used in (Sha, 2003), whose features are based on POS only, and (c) a non-chunked version of our model (denoted as *noChnk*), to compare the gain due to semantic *forms* vs. chunking.

5.2 Test Results

We measure the average CI for queries in each dataset with our technique against the above benchmark techniques. Results are reported in Table 3. For each dataset, we provide a detailed bar graph describing percentage of queries that produced outputs in some particular CI range.

TREC: Fig. 7(A) shows that *formNet* achieves *CI*=1 for 63.1% queries. In fact, only 9.2% of the

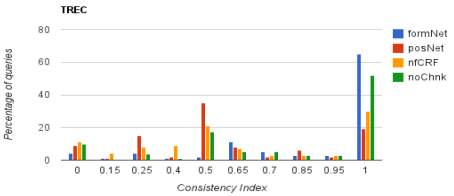


FIGURE 7. (A) – Percentage of queries that yielded some CI for TREC

	posNet	*formNet*	nfCRF	noChnk
TREC	54.8	***83.6***	58.6	74.0
MSQA	53.6	***79.5***	43.3	77.3
WSE	48.2	***73.2***	36.1	68.4

TABLE 3 – Evaluation results for various datasets in terms of average CI (%)

MSQA: Table 3 shows that *formNet* provides average *CI*=0.795 for MSQA queries whereas the benchmark *posNet* produces an average *CI*=0.536. This signifies ~ 49% improvement in performance. Fig. 7(B) shows that *formNet* can retrieve 55% queries with perfect match and produces a *CI*>0.5 for 85% queries in the dataset. In contrast, the benchmark *posNet* could only produce *CI*>0.5 for 38% queries. TREC queries are grammatically richer than MSQA; therefore a drop in overall performance is expected when evaluating MSQA. Interestingly, *forms* seem to be playing a stronger role in MSQA, since a traditional CRF performs poorly in this case.

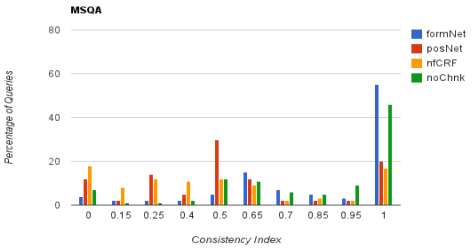


FIGURE 7. (B) – Percentage of queries that yielded some CI for MSQA

WSE: WSE queries are most diverse in construction and number of words. In Fig. 7(C), we see that performance is reduced for all techniques, but *formNet* still performs better than *posNet* by 51.86%. Observe that *noChnk* performs worst for TREC when compared to *formNet* than for any other dataset as indicated in Table 3 (difference between average CI for *formNet* and *noChnk*). This reaffirms our previous observation from the query data: TREC queries consist of longer

substantially better than *noChnk* for MSQA and WSE datasets, whereas no chunking for TREC significantly deteriorates performance. This indicates that chunking has a stronger impact in TREC, a dataset where 77% queries have more than 11 words (Fig. 6). In comparison, only ~10.8 % queries in MSQA and 7.6% queries in WSE have more than 9 words.

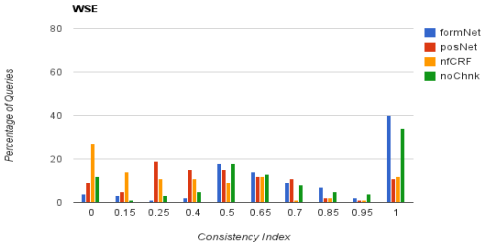


FIGURE 7. (C) – Percentage of queries that yielded some CI for WSE.

Cross Dataset Testing: Different datasets differ in query structure, context and length of query. To ensure robustness to different training environments, we perform cross dataset testing, i.e., train on one dataset and test on another (read TRAIN_TEST). Here, we report *formNet* performance. The average *CI* achieved by *formNet* is as follows: TREC_MSQA: 0.53, TREC_WSE: 0.44, MSQA_TREC: 0.68, MSQA_WSE: 0.58. We can observe that cross dataset testing provides best results when we train on MSQA and test on TREC. This is potentially due to the fact that the TREC dataset query structures are quite limited in construction, which are contained within queries of MSQA. Performance is worst when we train on TREC and test on WSE. This is potentially due to the diverse and noisy queries in WSE not captured during limited training over TREC. Nevertheless, for MSQA_WSE, *formNet* retrieves query subnets with *CI* > 0.5 in 73.1% cases and *CI* > 0.75 in 33% cases, suggesting robustness of *formNet* to web scale.

6 Conclusion

Several papers on computational cognitive psychology dwell on the fact that cognitive psychology models cannot be purely verified on the basis of behavioural experiments (Chater, 2006). For researchers in the domain of NLP, a fascinating possibility is to model cognitive techniques computationally and test their robustness to noise in NL. Natural languages are undeniably imprecise, especially in the realm of semantics. The primary reason of this imprecision is the fuzziness of class boundaries (Zadeh, 1998). Surprisingly, robustness to imprecision is often achieved by slightly relaxing the rigidity imposed by lexical grammar, by means of parsing at a higher abstraction than POS.

In this paper, we reproduce the structure-of-intellect model of cognitive psychology computationally. Exploring the various interactions among the semantic *forms* provides insights

References

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. (2011). Natural Language Processing (Almost) from Scratch. *J. Mach. Learn. Research* (November 2011), 2493-2537.

John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. (2001). Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data. In *Proceedings of the Eighteenth International Conference on Machine Learning* (ICML '01), San Francisco, CA, USA, 282-289.

Andrew McCallum, Dayne Freitag, and Fernando C. N. Pereira. (2000). Maximum Entropy Markov Models for Information Extraction and Segmentation. In *Proceedings of the Seventeenth International Conference on Machine Learning* (ICML '00), San Francisco, CA, USA, 591-598.

Minhua Huang and Robert M. Haralick. (2009). Identifying Patterns in Texts. In *Proceedings of the 2009 IEEE International Conference on Semantic Computing* (ICSC '09). IEEE Computer Society, Washington, DC, USA, 59-64.

Joel Booth, Barbara Di Eugenio, Isabel F. Cruz, and Ouri Wolfson. (2009). Query Sentences as Semantic (Sub) Networks. In *Proceedings of the 2009 IEEE International Conference on Semantic Computing* (ICSC '09). IEEE Computer Society, Washington, DC, USA, 89-94.

Rebecca Bruce and Janyce Wiebe. (1994). Word-sense disambiguation using decomposable models. In *Proceedings of the 32nd annual meeting on Association for Computational Linguistics* (ACL '94). Association for Computational Linguistics, Stroudsburg, PA, USA, 139-146.

Ana-Maria Popescu, Oren Etzioni, and Henry Kautz. (2003). Towards a theory of natural language interfaces to databases. In *Proceedings of the 8th international conference on Intelligent user interfaces* (IUI '03). ACM, New York, NY, USA, 149-157.

E. Kaufmann, A. Bernstein and L. Fisher. (2007). "NLP-Reduce: A "naïve" but domain-independent natural language interface for querying ontologies. In *4th European Semantic Web Conference (ESWC).*

Gabriel Cardona, Francesc Rossello, and Gabriel Valiente. (2009). Comparison of Tree-Child Phylogenetic Networks. *IEEE/ACM Trans. Comput. Biol. Bioinformatics* 6, 4 (October 2009), 552-569.

Fei Sha and Fernando Pereira. (2003). Shallow parsing with conditional random fields. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology - Volume 1* (NAACL '03), Vol. 1. Association for Computational Linguistics, Stroudsburg, PA, USA, 134-141.

Lance A. Ramshaw and Mitchell P. Marcus. (1995). Text chunking using transformation-based

Language Processing. *MIT Press*.

Joy P. Guilford. (1977). Way beyond the IQ. *Creative Education Foundation, NY.*

Soren Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann and Zachary Ives. (2007). DBpedia: A Nucleus for a Web of Open Data. In *6th Inter. Semantic Web Conference*, 11-15.

J Jian Hu, Gang Wang, Fred Lochovsky, Jian-tao Sun, and Zheng Chen. (2009). Understanding user's query intent with wikipedia. In *Proceedings of the 18th international conference on World wide web* (WWW '09). ACM, New York, NY, USA, 471-480.

Jiafeng Guo, Gu Xu, Xueqi Cheng, and Hang Li. (2009). Named entity recognition in query. In*Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval* (SIGIR '09). ACM, New York, NY, USA, 267-274.

Roberto Navigli. (2009). Word sense disambiguation: A survey. *ACM Comput. Surv.* 41, 2, Article 10 (February 2009), 69 pages.

John B. Carroll. (1993). Human Cognitive Abilities. *Cambridge University Press, Cambridge.*

Andrew McCallum. (2002). Efficiently inducing features of conditional random fields. In *Proceedings of the Nineteenth conference on Uncertainty in Artificial Intelligence* (UAI'03), Uffe Kjærulff and Christopher Meek (Eds.). Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 403-410.

William Croft and D. Alan Cruse. (2004). Cognitive Linguistics. *Cambridge University Press.*

N. Chater, J. B. Tenenbaum and A. Yuille. (2006). Probabilistic models of cognition: Conceptual foundations. In *Trends in Cognitive Science,* 10.

A. Herdagdelen, M. Ciaramita, D. Mahler, M. Holmqvist, K. Hall and S. Riezler. (2010). Generalized Syntactic and Semantic Models of Query Reformulation. In *SIGIR*, 283-290.

Lotfi A. Zadeh. (1998). Some reflections on soft computing, granular computing and their roles in the conception, design and utilization of information/intelligent systems. In Soft Computing 2(1): 23-25.

MSQA. (2008). http://research.microsoft.com/en-us/downloads/88c0021c-328a-4148-a158-a42d7331c6cf/

Charles A. Clarke, Nick Craswell, Ian Soboroff and Ellen M. Voorhees. (2011). Overview of the TREC 2011 Web Track. *Text Retrieval Conference*.

Silviu Cucerzan. (2007). Large-Scale Named Entity Disambiguation Based on Wikipedia Data. In *Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pp 708-716, June 2007.

David McClosky, Mihai Surdeanu, and Christopher D. Manning. (2011). Event extraction as dependency parsing. In *Proceedings of the 49th Annual Meeting of the Association for*

Association for Computational Linguistics, Stroudsburg, PA, USA, 326-334.

Kino Coursey and Rada Mihalcea. (2009). Topic identification using Wikipedia graph centrality. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics, Companion Volume: Short Papers* (NAACL-Short '09). Association for Computational Linguistics, Stroudsburg, PA, USA, 117-120.

Extraction of domain-specific bilingual lexicon from comparable corpora: compositional translation and ranking

Estelle DELPECH[1], Béatrice DAILLE[1], Emmanuel MORIN[1], Claire LEMAIRE[2,3]
(1) UNIVERSITÉ DE NANTES – LINA UMR 6241, 2 rue de la Houssinière, BP 92208, 44322 Nantes, Cedex 3, France
(2) UNIVERSITÉ STENDHAL – GRENOBLE – ILCA – BP 25, 38040 Grenoble Cedex 9, France
(3) LINGUA ET MACHINA, c/o Inria Rocquencourt BP 105, Le Chesnay Cedex 78153, France
estelle.delpech@univ-nantes.fr, beatrice.daille@univ-nantes.fr, emmanuel.morin@univ-nantes.fr, claire.lemaire1@e.u-grenoble3.fr

ABSTRACT

This paper proposes a method for extracting translations of morphologically constructed terms from comparable corpora. The method is based on compositional translation and exploits translation equivalences at the morpheme-level, which allows for the generation of "fertile" translations (translation pairs in which the target term has more words than the source term). Ranking methods relying on corpus-based and translation-based features are used to select the best candidate translation. We obtain an average precision of 91% on the Top1 candidate translation. The method was tested on two language pairs (English-French and English-German) and with a small specialized comparable corpora (400k words per language).

TITLE AND ABSTRACT IN ANOTHER LANGUAGE, FRENCH

Extraction de lexiques bilingues spécialisés à partir de corpus comparales : traduction compositionnelle et ordonnancement

Cet article propose une méthode permettant d'extraire des traductions de termes morphologiquement construits à partir de corpus comparables. La méthode se base sur la traduction compositionnelle et exploite des équivalences traductionnelles au niveau morphologique, ce qui nous permet de générer des traductions "fertiles" (des paires de traductions dans lesquelles le terme cible a plus de mots que le terme source). Des méthodes d'ordonnancement s'appuyant sur des traits extraits du corpus et des paires de traduction sont utilisées pour sélectionnner la meilleure traduction candidate. Nous obtenons une précision de 91% sur le Top1 en moyenne. La méthode a été testée sur deux paires de langues (anglais-français et anglais-allemand) et sur un corpus comparable spécialisé de petite taille (400k mots par langue).

KEYWORDS: COMPUTER-AIDED TRANSLATION, MACHINE TRANSLATION, COMPARABLE CORPORA, LEARNING-TO-RANK, COMPOSITIONALITY, TERMINOLOGY

MOTS-CLÉS : TRADUCTION ASSISTÉE PAR ORDINATEUR, TRADUCTION AUTOMATIQUE, CORPUS COMPARABLES, LEARNING-TO-RANK, COMPOSITIONNALITÉ, TERMINOLOGIE

Introduction

Comparable corpora are composed of texts in different languages which are not translations but deal with the same subject matter and were produced in similar situations of communication. They are used in Computer-Aided Translation to provide technical translators with domain-specific bilingual lexicons when there is no parallel data available (e.g. translation memories, multilingual terminologies). This situation happens when translators have to translate texts which deal with emerging technical domains or when the translation is done from/to an under-resourced language. Comparable corpora also have the advantage of containing more idiomatic expressions than parallel corpora do because the target texts do not bear the influence of the source language. Indeed, Baker (1996) observed that translated texts tend to bear features like explicitation, simplification, normalization and levelling out. As a consequence, one of the difficulties with comparable corpora is that the translation of a source term may not be present in its "normalized" or "canonical" form but rather in the form of a morphological or paraphrastic variant (e.g. *post-menopausal* translates to *après la ménopause 'after the menopause'* instead of *post-ménopausique*). Another limitation is that algorithms output, for each source term, a set of candidate translations instead of just one target term. This state of affairs makes it very challenging for translators to use lexicons extracted from comparable corpora in real-life situations (Delpech, 2011).

The solution that consists in increasing the size of the corpus in order to find more translation pairs or to extract parallel segments of text (Fung & Cheung, 2004; Rauf & Schwenk, 2009) is only possible when large amounts of texts are available. In the case of the extraction of *domain-specific* lexicons, we quickly face the problem of data scarcity: in order to extract high-quality lexicons, the corpus must contain text dealing with very specific subject domains and the target and source texts must be highly comparable. If one tries to increase the size of the corpus, one takes the risk of decreasing its quality by adding out-of-domain texts. Studies support the idea that the quality of the corpora is more important than its size. Morin *et al.* (2007) show that the discourse categorization of the documents increases the precision of the lexicon despite the data sparsity. Bo & Gaussier (2010) show that they improve the quality of the extracted lexicon if they improve the comparability of the corpus by selecting a smaller – but more comparable – corpus from an initial set of documents.

This paper proposes methods for ranking and extracting canonical translations as well as translation variants, with a special focus on the extraction of *fertile* translations. In parallel texts processing, the notion of fertility has been defined by Brown *et al.* (1993). They defined the fertility of a source word *e* as the number of target words to which *e* is connected in a randomly selected alignment. Similarly, we call a fertile translation a translation pair in which the target term has more words than the source term. The identification of fertile translations is useful because (i) they frequentlty correspond to non-canonical translations, e.g. paraphrastic variants and (ii) they tend to correspond to vulgarized forms of technical terms (e.g. « cytotoxic » vs. « toxic to the cells ») which are useful when the translator translates lay science texts. Up to now, fertility has received little attention in the field of comparable corpora processing. To our knowledge, only Daille & Morin (2005) and Weller *et al.* (2011) tried to extract translation pairs of different lengths from comparable corpora. Daille & Morin (2005) focus on the specific case of multi-word terms whose meaning is not compositional and tried to align these multi-word terms with either single-word terms or multi-word terms using a context-based approach. Weller

et al. (2011) concentrate on translating noun compounds as noun phrases. Similar to the approach presented here, Claveau & Kijak (2011) use translation equivalences between morphemes to generate translations and can handle fertility. However it is not suited for comparable corpora since it requires domain-specific parallel data (in their case, a multilingual terminology) to learn alignment probabilities.

Our method is based on compositional translation. We chose this approach because: (i) according to Namer & Baud (2007), compositional terms form a major part of the new terms found in technical and scientific domains, this is not restricted to the field of biomedicine as it is generally believed ; (ii) compositionality-based methods have been shown to clearly outperform context-based ones for the translation of terms with compositional meaning, both in terms of translation accuracy and rank of the correct candidate translation (Morin & Daille, 2010) ; (iii) we believe that compositionality-based methods offer the opportunity to generate fertile translations if combined with a morphology-based approach. This method, which we call morpho-compositional translation, consists in: **(i) decomposing** the source term into morphemes: *post-menopause* is split into *post-* + *menopause*[1] ; **(ii) translating** the morphemes to bound morphemes or fully autonomous words: *post-* becomes *post-* or *après, menopause* becomes *ménopause* ; **(iii) recomposing** the translated elements into a target term: *post-ménopause 'post-menopause'*, *après la ménopause 'after the menopause'*. Fertile translations can be generated because we allow bound morphemes to be translated to autonomous lexical items (e.g. prefix *post-* → preposition *après*). The proposed ranking methods exploit various corpus-based and translation-based features.

This paper falls into 4 sections. Section 1 outlines recent research in compositional approaches to bilingual lexicon extraction. Section 2 explains the methods we designed for translation generation and ranking. Section 3 describes our experimental data. Section 4 presents and discusses the results of our experimentations.

1 Compositional approaches to bilingual lexicon extraction

The core of compositional translation consists in generating candidate translations following the principle of compositionality: "*the meaning of the whole is a function of the meaning of the parts*" (Keenan & Faltz, 1985, pp. 24-25). Once the candidate translations have been generated, one generally ranks them and selects the TopN candidate translations. Generation methods are described in section 1.1. Ranking methods are described in section 2.3.

1.1 Generation methods

Compositional translation consists in decomposing the source term into atomic components, translating these components into the target language and recomposing the translated components into target terms. Existing implementations differ on the kind of atomic components they use for translation.

Lexical compositional translation (Baldwin & Tanaka, 2004; Grefenstette, 1999; Morin & Daille, 2009; Robitaille *et al.*, 2006) deals with multi-word term to multi-word term alignment and uses lexical words as atomic components: *rate of evaporation* is translated into French as

[1]We use the following notations: trailing hyphen for prefixes (*a-*), leading hyphen for suffixes (*-a*), both for confixes (*-a-*), no hyphen for autonomous morphemes (*a*) and a plus sign (+) for intra-word morpheme boundaries. The term *confix* is borrowed from Martinet (1979) and refers to neoclassical (Latin or Ancient Greek) roots.

taux d'évaporation by translating *rate* to *taux* and *evaporation* to *évaporation* using dictionary lookup. Recomposition may be done by permuting the translated components (Morin & Daille, 2010) or with translation patterns (Baldwin & Tanaka, 2004).

Sublexical compositional translation deals with single-word term translation. The atomic components are subparts of the source single-word term. Cartoni (2009) translates neologisms created by prefixation with a formalism called Bilingual Lexeme Formation Rules. Atomic components are the prefix and the lexical base: Italian neologism *ricostruire 'rebuild'* is translated into French *reconstruire* by translating the prefix *ri-* to *re-* and the lexical base *costruire* as *construire*. Weller *et al.* (2011) translate two types of single-word term. German single-word terms formed by the concatenation of two neoclassical roots are decomposed into these two roots, then the roots are translated into target language roots and recomposed into an English or French single-word term, e.g. *Kalori*$_1$*metrie*$_2$ is translated as *calori*$_1$*metry*$_2$. German NOUN$_1$+NOUN$_2$ compounds are translated into French and English NOUN$_1$ NOUN$_2$ or NOUN$_1$ PREP NOUN$_2$ multi-word terms, e.g. Elektronen$_{N1}$-mikroskop$_{N2}$ is translated to *electron*$_{N1}$ *microscope*$_{N2}$. Garera & Yarowsky (2008) translate various compound sequences (NOUN$_1$+NOUN$_2$, ADJ$_1$+NOUN$_2$ ...). They generate an English literal gloss of the compounds with the compositional method (for instance, the English gloss for the Albanian word *hekurudhë 'railway'* is *iron path*). Then, they search for entries in *Lx*-to-English dictionaries where the entry in language *Lx* is a word-to-word translation of the English gloss (e.g. *iron path* matches the German entry *Eisenbahn* and the Italian entry *ferrovia*). The final candidate translations are the fluent English translations proposed by the bilingual dictionaries (e.g. *Eisenbahn* and *ferrovia* both translate to *railway* ; *railway* is considered as a potential translation for *hekurudhë*).

1.2 Ranking and selection methods

Generally, compositional translation generates several possible translations for one source term. One has to find a way to rank the translations from the most to the least reliable. Garera & Yarowsky (2008) tried two ranking methods: (i) a probability score $\mathcal{P}$ based on the number of different languages exhibiting the association between the literal gloss and the fluent translation ; (ii) the probability score $\mathcal{P}$ combined with the similarity of the source and target words' contexts using context-based methods like in the work of Rapp (1995) and Fung (1997). Robitaille *et al.* (2006) extract translation pairs from a corpus built by querying a search engine with a set of seed translation pairs. They select the candidate translations which are semantically related to the target seed terms. The semantic similarity measure is based on the number of hits containing the seed term and/or the candidate translation (Jaccard coefficient). Other works simply select the candidate translations which occur in the target corpus (Weller *et al.*, 2001 ; Morin and Daille, 2010) or which are significantly attested on the Web (Cartoni, 2009).

Only Baldwin and Takana (2004) use machine learning. They train a SVM classifier with corpus-based, dictionary-based and translation pattern-based features and use the value returned by the classifier (a continuous value between -1 and +1) to rank the candidate translations. Their approach is tantamount to point-wise approaches in learning-to-rank. To our knowledge, no research work has investigated the possible contribution of advanced learning-to-rank algorithms to candidate translations ranking. Learning-to-rank algorithms are widely used in Information Retrieval for ranking documents from the most to the least relevant to a given query (Li, 2011). They can be easily ported to the problem of ranking the candidate translations of a source term. There exists three families of learning-to-rank algorithms: **point-wise** (for a given a query-

document pair, predict its relevance score or label: ranking is treated as a regression or classification problem), **pair-wise** (for a given query and two documents, indicate which of the two documents is the most relevant) and **list-wise** (given a query and a list of documents, indicate how to order the documents: this last family straightforwardly represents learning-to-rank problem). According to the tests of Liu (2009), list-wise algorithms generally outperform the two other approaches.

1.3 Challenges of compositional translation

Compositional translation faces four main challenges which are: **morphosyntactic variation:** source and target terms' morphosyntactic structures are different: *anti-cancer*$_{NOUN}$ → *anti-cancéreux*$_{ADJ}$ *'anti-cancerous'* ; **lexical variation:** source and target terms contain semantically related - but not equivalent - words: *machine translation → traduction automatique 'automatic translation'* ; **terminological variation:** a source term can be translated to different target terms: *oophorectomy → ovariectomie 'oophorectomy', ablation des ovaires 'removal of the ovaries'* ; **fertility:** the target term has more content words than the source term. Note that fertility can have two origins. In the case of ***surface*** **fertility**, the target term has more words than the source term but source and target terms have the same number of morphemes. Source and target languages differ in the way they concatenate morphemes to form words: *bi-dimensional → deux dimensions 'two dimensions'*. In the case of ***semantic*** **fertility,** the target term has more morphemes than the source term. Source and target languages differ in the way they combine elements of meaning to create new words: *voie de glace 'route of ice' → ice climbing route, aquarelle* (not decomposable) → *water color*.

Solutions to morphosyntactic, lexical and to some extent terminological variation have been proposed in the form of thesaurus lookup (Robitaille *et al.*, 2006), morphological derivation rules (Morin & Daille, 2010), morphological variant dictionaries (Cartoni, 2009) or morphosyntactic translation patterns (Baldwin & Tanaka, 2004; Weller *et al.*, 2011). Although it is not specifically outlined in their paper, the work of Garera & Yarowsky (2008) can theoretically handle semantic fertility and lexical divergence. However, their method depends on a large number of dictionaries with a substantial coverage of compounds. Surface fertility has been addressed by Weller *et al.* (2011) for the specific case of German NOUN+NOUN compounds. The method presented here is able to deal with surface fertility and to generate and rank translations for a large variety of morphologically constructed words.

2 Translation method

2.1 Principle of morpho-compositional translation

The idea of morpho-compositional translation is to apply the principle of compositional translation at the morpheme-level rather than at the lexical level and to allow translation equivalences between bound and autonomous morphemes in order to generate fertile translations. It relies on the assumptions that: **(i) a lexical item can be decomposed into smaller components**. These components may be **free**, i.e. they can occur in texts as autonomous lexical items like *toxicity* in *cardiotoxicity* or **bound**, i.e. they cannot occur as autonomous lexical items, in that case they correspond to bound morphemes like *-cardio-* in *cardiotoxicity* ; **(ii) a bound component can be translated to an autonomous or a bound component:** *-cardio-* can be translated to *-cardio-* or *cœur 'heart'* or *cardiaque 'cardiac'*. Thus, *cardiotoxicity* can be

translated to *toxicité cardiaque 'cardiac toxicity'* or *toxicité pour le cœur 'toxicity to the heart'* or *cardiotoxicité 'cardiotoxicity'*.

Like other sublexical approaches, the main idea behind morpho-compositional translation is to go beyond the word level and work with subword components. In our case, these components are morpheme-like items which either (i) bear referential lexical meaning like confixes (*-cyto-, -bio-*) and autonomous lexical items (*cancer, toxicity*) or (ii) can substantially change the meaning of a word, especially prefixes (*anti-, post-*) and some suffixes (*-less, -like*). Unlike other approaches, morpho-compositional translation is not limited to small set of source-to-target structure equivalences. It takes as input a morphologically constructed single-word term which can be the result of prefixation '*pretreatment*', confixation '*densitometry*', suffixation '*childless*', compounding '*anastrozole-associated*' or any combinations of the four. Its output is a set of single or multi-word candidate translations. For instance, *postoophorectomy* may be translated to *postovariectomie 'postoophorectomy'* or *après l'ovariectomie 'after the oophorectomy'* or *après l'ablation des ovaires 'after the removal of the ovaries'*.

Section 3.2 explains the algorithm for generating candidate translations. Section 3.3 describes different methods for ranking the candidate translations.

2.2 Generation algorithm

The generation method is described in the algorithm 1. A detailed version of the algorithm can be found in the feasibility study of Delpech *et al.* (2012).

Algorithm 1 Generate translations

Require: *source_term, target_corpus*
translations ← *Ø*
for all *{c1, ... ci}* in DECOMPOSE *(source_term)* ***do***
  for all *{e1, ... ej}* in CONCATENATE (*{c1, ... ci}*) **do**
    for all *{t1, ... tk}* in {TRANSLATE (*e1*) × … TRANSLATE (*ej*)} **do**
      if $k \neq j$ **then**
        continue
      for all*{t1, ... tk}* in PERMUTATE (*{t1, ... tk}*) **do**
        for all *{w1, ... wl}* in CONCATENATE (*{t1, ... tk}*) **do**
          for all *match* in MATCH (*{w1, ... wl}, target_corpus*) **do**
            add *match* to *translations*
return *translations*

The DECOMPOSE function splits the source term into minimal components *{c1, ... ci}* by matching substrings of the term with lists of prefixes, confixes, suffixes and lexical items and respecting some length constraints on the substrings. When several splittings are possible, only the ones with the highest number of components are retained.

The CONCATENATE function generates all possible concatenations of a list of components. For example, if the term "*abc*" has been split into 3 components *{a, b, c}*, then there are 4 different concatenations : *{a,bc}, {ab,c}, {a,b,c}, {abc}* (for *n* components, we have 2^{n-1} possible concatenations). When called used after the decomposition, the concatenation of the components increases the chances of matching the entries of the linguistic resources used by the TRANSLATE function. When called after the permutation, the concatenation is used to recreate a set of target words *{w1, ...wl}* from the set of translated components.

The TRANSLATE function uses two kinds of linguistic resources to generate translations. Bilingual resources map elements across languages. Variation resources are used to handle variation at the lexical and morphological level. Hence, the output of TRANSLATE(*e*) correspond to *Trans(e) U Trans(Var^{src}(e)) U Var^{tgt}(Trans(e))* where *Trans* is a bilingual resource, *Var* a variation resource, *src* is the source language and *tgt* is the target language. For example, *toxic* can be translated to *toxique 'toxic', toxicité 'toxicity'* or *vénéneux 'poisonous'*. If one element can not be translated then the translation of the whole fails.

The PERMUTATE function serves to capture the fact that components' order may be different in the source and target language (distortion). As a general rule, *O(n!)* procedures should be avoided but we are permuting small sets (up to 4 items).

The MATCH function returns a series of tokens which occur in the target corpus and whose lemmas match the generated target words *{w_1, ... w_l}*. We allow for 3 stop words between each lemma. For example, if the system generates the target words *{toxique, cellule} 'toxic, cell'* from *cytotoxic,* it will match *"toxique pour les cellules" 'toxic to the cells'*. We consider that two matches are one and the same translation if they correspond to the same series of *(lemma, part-of-speech)* pairs. For example, *toxique pour les cellules* and *toxique pour la cellule 'toxic to the cell'* correspond to the same translation.

2.3 Ranking methods

We have considered four parameters for ranking the translations: frequency of the translation, part-of-speech translation probability, context similarity and the reliability of the resources used for translating the components of the source term. These parameters can be used separately or in a combined manner.

The frequency (FREQ) corresponds to the number of occurrences of the translation in the target corpus divided by the total number of words in the target corpus.

The part-of-speech translation probability (POS) corresponds to *P(y|x)*, the probability that a source term with part-of-speech *x* will be translated to a target term with part(s)-of-speech *y*, e.g. it is more probable that a NOUN is translated by another NOUN or by a NOUN PREP NOUN sequence rather than an ADVERB. The part-of-speech translation probabilities were acquired by running the software ANYMALIGN (Lardilleux, 2008) on the EMEA corpus (Tiedemann, 2009) which had been previously pos-tagged with the linguistic analyzer XELDA[2]. ANYMALIGN outputs a phrase translation table. Each line of the translation table corresponds to an alignment *a = {lem$_s$, pos$_s$, lem$_t$, pos$_t$, p(s|t), p(t|s)}* where *pos$_s$* is the parts-of-speech of the source phrase, *pos$_t$* is the parts-of-speech of the target phrase and *pos(t|s)* is the probability of translating the source phrase to the target phrase. From these alignments, we obtain *P(y|x)* with the following formula:

$$P(y|x) = \frac{\sum_{\{a \in A | pos_s = x,\ pos_t = y\}} p(t|s)}{\sum_{\{a \in A | pos_s = x\}} p(t|s)}$$

The context similarity (CONT) corresponds to the method used for ranking translations in context-based approaches. For each source term and target term we build a *context vector*. This vector indicates the number of times the term co-occurs with each word of the corpus within a

[2] http://www.temis.com

contextual window of 5 words around the term. The number of co-occurrences is normalized with the log-likelihood ratio (Dunning, 1993). Then, the vector of the source term is translated into the target language. Finally, the source vector and the target vector are compared: the most similar the vectors, the most likely the target and source terms are translations of each other. The similarity between source vector *s* and target vector *t* is computed with the weighted jaccard:

$$WeightedJaccard(s,t) = \frac{\sum_{w \in s \cap t} min(c(s,w), c(t,w))}{\sum_{w \in s \cap t} max(c(s,w), c(t,w)) + \sum_{w \in s \setminus t} c(s,w) + \sum_{w \in t \setminus s} c(t,w)}$$

where *c(s, w)*, respectively *c(t, w),* is the normalized number of co-occurrences between the source, respectively target, term and word *w*. Note that for multi-word terms, the context vector corresponds to the union of the context vectors of the lexical words that compose the multi-word term.

The resources score (RESO) corresponds to:

$$RESO(t) = \frac{1}{|T|} \sum_{i=1}^{|T|} rel(T_i)$$

where *T* is the total number of components in the target term *t* and $rel(T_i)$ is the reliability of the target component T_i. The reliability of a component is a float value between 0 and 1 inclusive. It depends on the nature of the component and on the resources which were used to generate it. We defined 8 types of target components: 1) the target component is a lexical item found in a general-language dictionary ; 2) target component is a lexical item which was found by cognate matching ; 3) target component is a lexical item which is a lexical variant of the translation (found in general language dictionary or by cognate matching) ; 4) target component is a lexical item which is a morphological variant of the translation ; 5) target component is a lexical item which is the translation of a bound morpheme ; 6) target component is a prefix ; 7) target component is a confix ; 8) target component is a suffix. We tuned the reliability values associated to these 8 types of target components empirically: we tested several arrangements[3] of reliability values on a training dataset (described in section 3.3) and retained the arrangement that gave the best rankings.

Scores combination (COMBI) is the linear combination of the FREQ, POS, CONT and RESO scores.

Learning-to-rank algorithms (LTR) were also tested. We tried three list-wise algorithms: AdaRank (Li & Xu, 2007), Coordinate Ascend (Metzler & Croft, 2000) and LambdaMart (Wu *et al.*, 2010). We used the implementations available in the RankLib software[4]. The predictive variables are the FREQ, POS, CONT and RESO scores. We trained the models on the training dataset described in section 3.3.

3 Data

We worked with 3 languages: English as source language and French and German as target languages.

[3] all 8-arrangement with repetition of {0, 0.2, 0.4, 0.6, 0.8, 1}

[4] http://people.cs.umass.edu/~vdang/ranklib.html. We set the metric to optimize on the training data to MAP (Manning *et al.*, 2008) ; all other parameters were left to default.

3.1 Comparable corpora

Our corpus is composed of specialized texts from the medical domain dealing with breast cancer. We define specialized texts as texts being produced by domain experts and directed towards either an expert or a non-expert readership (Bowker & Pearson, 2002). The texts were collected from scientific papers portals and from information websites targeted to breast cancer patients and their relatives. Each corpus has approximately 400k words (cf. table 1). All texts were pos-tagged and lemmatized with XELDA. We also computed the comparability of the corpora[5]. The English-French corpus' comparability is 0.71 and the English-German corpus' comparability is 0.45. The difference in comparability can be explained by the fact that German texts on breast cancer were hard to find (especially scientific papers): we had to collect texts in which breast cancer was not the main topic. This may have added out-of-domain words.

	EN	**FR**	**DE**
Expert readership	218.3k	267.2k	197.2k
Non-expert readership	198.2k	184.5k	201.7k
TOTAL	416.5k	451.75k	398.9k

TABLE 1: Composition and size of corpora (nb. of words)

3.2 Resources for generation

Tables 2 and 3 show the size of the resources we used for generation.

General language dictionary: We used the dictionary which is part of the XELDA software. This dictionary was used for generating translations but also for computing the corpus comparability and for translating the context vectors for the context similarity measure (CONT score).

Cognate dictionary: We built this resource automatically by extracting pairs of cognates from the comparable corpora. We used the same technique as Hauer & Kondrak (2011): a SVM classifier trained on examples taken from online dictionaries[6].

Morpheme translation table: this resource was created manually by translators since there exists no publicly available morphology-based bilingual dictionary. This translation table links the English bound morphemes contained in the source terms to their French or German equivalents (which can be bound morphemes or lexical items).

In order to handle the variation phenomena described in section 1.3, we used a **dictionary of synonyms** and lists of **morphologically related words**. The dictionary of synonyms is part of the XELDA software. Morphologically related words were collected by stemming the words of the comparable corpora and the entries of the bilingual dictionary with the algorithm of Porter (1980).

The DECOMPOSE function uses the entries of the morpheme translation table (242 entries) and a list of 85k lexical items composed of the entries of the general language dictionary and English words extracted from the Leipzig Corpus (Quasthoff *et al.*, 2006).

[5]We used the measure defined by Bo & Gaussier (2010) which indicates, given a bilingual dictionary, the expectation of finding, for each word of the source corpus, its translation in the target corpus and *vice-versa.*
[6]http://www.dicts.info/uddl.php

	EN → FR	EN → DE
General language	38k → 60k	38k → 70k
Domain specific	6.7k → 6.7k	6.4k → 6.4k
Morphemes (TOTAL)	242 → 729	242 → 761
Prefixes	50 → 134	50 → 166
Confixes	185 → 574	185 → 563
Suffixes	7 → 21	7 → 32

TABLE 2: Nb. of entries in the multilingual resources

	EN → EN	FR → FR	DE → DE
Synonyms	5.1k → 7.6k	2.4k → 3.2k	4.2k → 4.9k
Morphological families	5.9k → 15k	7.1k → 18k	7.4k → 16k

TABLE 3: Nb. of entries in the monolingual resources

3.3 Datasets for evaluation and training

We extracted morphologically constructed source terms from the English texts in a semi-supervised manner: **(i)** we wrote a short seed list of English bound morphemes. We automatically extracted from the English texts all the words that contained these morphemes. For example, we extracted the words *postchemotherapy* and *poster* because they contained the string *post-* which corresponds to a bound morpheme of English ; **(ii)** The extracted words were sorted: those which were not morphologically constructed were eliminated (like *poster*), and those which were morphologically constructed were kept (like *postchemotherapy*). The morphologically constructed words were manually split into morphemes. For example, *postchemotherapy* was split into *post-*, *-chemo-* and *therapy* ; **(iii)** if some bound morphemes which were not in the initial seed list were found when we split the words during step (ii), we started the whole process again, using the new bound morphemes to extract new morphologically constructed words.

We also added hyphenated terms like *ER-positive* to our list of source terms. With this method, we collected 2025 source terms. Then, we excluded all the source terms which could be translated with the general language dictionary and whose translation was present in the target corpus. Finally for each language pair, we divided the source terms into two groups:

The evaluation dataset contains source terms which could be translated with the UMLS meta-thesaurus (Bodenreider, 2004) and whose translation was in the target corpus – these terms, along with their UMLS translations, constitute the reference lexicon for the evaluation.

	EN → FR	EN → DE
EVALUATION dataset	126 → 163	90 → 104
TRAINING dataset	642 → 1953	584 → 1826

TABLE 4: Size of datasets

The training dataset contains source terms which could not be translated with the general-language dictionary or the UMLS but for which we could generate translations with our method. This generated translations were manually annotated by translators. These terms, along with their annotated translations, were used as training data for learning the ranking models and to tune the reliability values used in the RESO score. We used four classes for the annotation: *exact, acceptable, related* and *wrong*. An exact translation is a canonical translation like *cytoprotection → Zellschutz* (DE), *protection des cellules 'protection of the cells'* (FR). An acceptable translation is a variant of the canonical translation: *cytoprotection → protéger les cellules 'protect the cells', cytoprotecteur 'cytoprotective'.* A related translation is a translation which is only semantically related to the source term: *insecure → ohne Sicherheit 'without safety'.* All other translations are wrong translations. We computed inter-annotator agreement on a set of 100 randomly selected translations. We used the Kappa statistics (Carletta, 1996) and obtained a high agreement (0.77 for English to German translations and 0.71 for English to French).

4 Results

4.1 Related work

Generally, systems are compared using the *TopN* precision: the percentage of source terms with at least one exact translation among the *TopN* candidate translations. Compositional-translation methods tend to give better results when they are applied to general language texts rather than domain-specific texts. Indeed, it is easier to find translations of the components since they belong to the general language and large corpora are also easier to collect. **Working with general language texts,** Baldwin & Tanaka (2004) were able to generate candidate translations for 92% of their source terms and they report 43% (gold-standard) to 84% (silver standard) of correct translations on Top1. Corpus' size exceeds 80M words for each language. Cartoni (2009) works on the translation of prefixed Italian neologisms into French. He finds that between 42% and 94% of the generated neologisms occur more than five times on the Internet. Garera & Yarowski (2008) obtain translations for 13% of the source words and the best precision is 39% for the Top10 candidate translations (for German and Swedish). **Regarding domain-specific translation,** Robitaille *et al.* (2006) collect translation pairs in an incremental manner. They start with a list of 9.6 pairs (on average) with a precision of 92% and end up with a final output of 19.6 pairs on average with a precision of 81%. Morin & Daille (2010) could generate candidate translations for 15% of their source terms and they report 88% of correct translations on Top1. The size of their corpus is 700k words per language. Weller *et al.* (2011) obtained correct English translations for 18% of their German compounds. Their corpus contains approximately 1.5M words per language.

4.2 Generation

We tested several combinations of linguistic resources. Table 5 only shows the results for the best combination. For English to French translation, the best results where obtained with all the combined resources, closely followed by the combination of the general language and the cognate dictionary. For English to German translation, the best results were obtained with the combination of the general language and the cognate dictionary. Morphologically related words and synonyms tend to increase the number of generated translations to the cost of translation accuracy.

	EN → FR	**EN → DE**
# source terms	126	90
# source terms with no translation	40 (32%)	34 (38%)
# source terms with at least one translation	86	56
# nb of translations / source term	2.05	2.6
# at least one reference translation (UMLS)	68 (79%)	40 (71%)
# at least one exact translation (translators or UMLS)	81 (94%)	51 (91%)

TABLE 5: Results of generation

Regarding English to French translations, we were able to generate translations for 86 of the 126 English source terms (68%). Among these 86 source terms, 79% had the UMLS reference translation among their candidate translations. Regarding English to German translations, we were able to generate translations for 56 of the 90 English source terms (62%). Among these 56 source terms, 71% had a reference translation among their candidate translations. We noticed that the algorithm generated translations which were not in the UMLS lexicon but which were exact translations according to the translators. For example, the German reference translation for *mastectomy* is *mastektomie*. The system generated the reference translation *mastektomie* but also translations like *ablation der brust, abschnitt der brust, brustentfernun, entfernung der brust* which are all exact translations. Thus, if we take into account these translations, we find that 94% and 91% of the source terms had at least one exact translation for English to French and English to German respectively. Among all these correct translations 21% and 10% were fertile translations for English to French and English to German respectively.

There are several reasons for untranslated source terms. In 30% of the cases, silence is due to the coverage of the linguistic resources: some of the components could not be translated and the translation of the whole source term failed. Another 30% of target terms do not have a compositional meaning: *breastfeeding* → *allaitement* (FR), *stillen* (DE). The third reason is due to lexical variation (~ 20%), e.g. *radio+sensitivity* translates to *strahlen+toleranz* but *toleranz 'tolerance'* has a different meaning than *sensitivity*. There was also cases of semantic fertility (~ 13%).

Errors were mainly due to problems in word reordering when generating fertile translations, especially with German. Other errors were due to wrong translations in the cognate dictionary and translations which were inappropriate in the context, e.g. the translation of *gynae* to *frau*

'woman' in *gynaecomastia → Frau gegen Brust 'women against breast'*. If we look at the part of fertile translations in the incorrect translations, we find that they constitute half of the English to French incorrect translations and 80% of the English to German incorrect translations. We think it is due to the morphological type of the languages involved in the translation. As a matter of fact, fertile variants are more natural and more frequent in French than in German. English and German are Germanic languages with a tendency to build new words by agglutinating words or morphemes into one single word. Noun compounds such as *anthracycline-containing* or *Anthracyclin-enthaltende* are common in these two languages. Conversely, French is a Romance language which prefers to use phrases composed of two nouns and a preposition rather than a single-noun compound. For example, *anthracycline-containing* would be translated as *comprenant une anthracycline 'containing an anthracycline'*. There is no non-fertile equivalent in French (**anthracycline-contenant* would be ungrammatical). It is the same with the bound/free morpheme alternation. The term *cytotoxic* will be translated into German as *zytotoxisch* whereas in French it can be translated as *cytotoxique* or *toxique pour les cellules 'toxic to the cells'*.

4.3 Ranking

Table 6 indicates the precision on Top1, 2 and 3, i.e. the percentage of source terms which have at least one exact translation (found in the UMLS or according to the translators) on the TopN candidate translations.

	EN → FR			**EN → DE**			**Average**
	Top1	**Top2**	**Top3**	**Top1**	**Top2**	**Top3**	**Top1**
RANDOM	.83	.88	.93	.80	.88	.88	.815
FREQ	.92	.92	.94	.84	.88	.91	.88
POS	.88	.93	.94	**.91**	**. 91**	**.91**	.895
CONT	.90	.91	.93	.82	.88	.88	.86
RESO	.92	.94	.94	.82	.86	.88	.87
COMBI	**.93**	**.94**	**.94**	**.89**	**.89**	**.91**	**.91**
LTR ADARANK	.90	.90	.93	.84	.88	.88	.87
LTR COORDINATE ASCEND	**.93**	**.94**	**.94**	**.89**	**.89**	**.91**	**.91**
LTR LAMBDAMART	.86	.91	.93	.88	.91	.91	.87

TABLE 6: Results of ranking

We tested the ranking methods described in section 2.3: the scores FREQ, POS, CONT and RESO separately, a linear combination of these scores (COMBI) and three learning-to-rank algorithms (LTR). We also randomly ranked the translations to serve as a baseline. On average, the best precision on Top1 (91%) is obtained with the linear combination and Coordinate Ascend. All ranking methods perform better than the baseline. For English to French, the best rankings were obtained with the linear combination and Coordinate Ascend. For English to German, the best

rankings were obtained with the POS parameter alone, closely followed by the linear combination and Coordinate Ascend.

We expected learning-to-rank algorithms to perform much better than simple methods like part-of-speech probabilities or the linear combination of several scores. This might be due to the small size of our training dataset (approx. 600 ranked lists per language). We note that the context similarity score (CONT) is the least performing ranking method. Similarly, Garera and Yarowsky (2008) note only a small performance gain when they use context similarity. This might be due to the fact that context-based methods need the source and target words to be very frequent in the corpora to work properly. The lower quality of the German translations can be explained by the fact that the English-German corpus is much less comparable than the English-French corpus (0.45 vs. 0.71).

Conclusion and perspectives

We have proposed a new compositional translation method for domain-specific bilingual lexicon extraction from comparable corpora. We obtain an average precision of 91% on the Top1 candidate translation. English-to-French translation performs slightly better than English-to-German translation, probably due to the morphological type of the languages and to the lower quality of the German data.

Future work includes the improvement of the identification of morphological variants. The morphological families extracted by the stemming algorithm are too broad for the purpose of translation. For example, the words *desirability* and *desiring* have the same stem but they are too distant semantically to be used to generate translation variants. We need to restrict the morphological families to a smaller set of morphological relations (e.g. noun → relational adjective). Furthermore, some work needs to be done on lexical variation: we used a dictionary of synonyms, but a thesaurus, which contains a large variety of semantic relations, may help us better in tackling lexical variation. Another improvement will be to use translation patterns to recompose the components into a target language term structure instead of using the PERMUTATE and CONCATENATE functions.

Our investigation of the contribution of learning-to-rank algorithms to the problem of candidate translations ranking shows encouraging results and we should further pursue this line of research. Future experiments include: testing other learning-to-rank approaches (pair-wise, point-wise), increasing the number of predictive variables (e.g. source/target term frequency ratio, number of components...) and finding ways to increase the size of the training dataset at a lesser cost.

Acknowledgment

The research leading to these results has received funding from the French National Research Agency under grant ANR-08-CORD-013 and from the European Community's Seventh Framework Programme (FP7/2007-2013) under Grant Agreement no 248005. We would like to thank the company LINGUA ET MACHINA (www.lingua-et-machina.com) for supporting this research as well as Clémence de Baudus and Kiril Isakov for their participation in the annotation task and Van Dang for his help with the RankLib software.

References

Baker, M. (1996). Corpus-based translation studies: The challenges that lie ahead. In *Terminology, LSP and Translation: Studies in Language Engineering in Honour of Juan C. Sager*. Somers H., Amsterdam & Philadelphia, John Benjamins edition.

Baldwin, T., & Tanaka, T. (2004). Translation by Machine of Complex Nominals. *Proceedings of the ACL 2004 Workshop on Multiword expressions: Integrating Processing* (pp. 24-31). Barcelona, Spain.

Bo, L., & Gaussier, E. (2010). Improving Corpus Comparability for Bilingual Lexicon Extraction from Comparable Corpora. *Proceedings of the 23th International Conference on Computational Linguistics* (pp. 23-27). COLING 2010, Beijing, Chine.

Bodenreider, O. (2004). The Unified Medical Language System (UMLS): integrating biomedical terminology. *Nucleic Acids Research*, *32*, 267-270.

Bowker, L., & Pearson, J. (2002). *Working with Specialized Language: A Practical Guide to Using Corpora*. London/New York: Routledge.

Brown, P., Della Pietra, S., Della Pietra, V., & Mercer, R. (1993). The mathematics of statistical machine translation: parameter estimation. *Computational Linguistics*, *19*(2), 263-311.

Carletta, J. (1996). Assessing Agreement on Classification Tasks: The Kappa Statistic. *Computational Linguistics*, *22*(2), 249-254.

Cartoni, B. (2009). Lexical Morphology in Machine Translation: A Feasibility Study. *Proceedings of the 12th Conference of the European Chapter of the ACL* (pp. 130-138). EACL 2009, Athens, Greece.

Claveau, V., & Kijak, E. (2011). Morphological analysis of Biomedical Terminology with Analogy-Based Alignment. *Proceedings of Recent Advances in Natural Language Processing* (pp. 347-354). RANLP 2011, Hissar, Bulgaria.

Daille, B., & Morin, E. (2005). French-English terminology extraction from comparable corpora. *Proceedings of the 2nd International Joint Conference on Natural Language Processing*, Lecture Notes in Computer Science (Vol. 3651, pp. 707-718). Jeju Island, Korea: Springer.

Delpech, E. (2011). Evaluation of terminologies acquired from comparable corpora : an application perspective. *Proceedings of the 18th Nordic Conference of Computational Linguistics* (pp. 66-73). NODALIDA 2011, Riga, Latvia.

Delpech, E., Daille, B., Morin, E. & Lemaire, C. (2012). Identification of Fertile Translations in Medical Comparable Corpora : a Morpho-Compositional Approach. *Proceedings of the 10th Biennal Conference of the Association for Machine Translation in the Americas* (in press). AMTA 2012, San Diego, CA, USA.

Dunning, T. (1993). Accurate Methods for the Statistics of Surprise and Coincidence. *Computational Linguistics*, *19*(1), 61-74.

Fung, P. (1997). Finding Terminology Translations from Non-parallel Corpora. *Proceedings of the 5th Workshop on Very Large Corpora* (pp. 192-202), Hong Kong.

Fung, P., & Cheung, P. (2004). Mining Very-Non-Parallel Corpora: Parallel Sentence And Lexicon Extraction Via Bootstrapping And EM. *Proceedings of Empirical Methods in Natural Language Processing* (pp. 57-63). EMNLP 2004, Barcelona, Spain.

Garera N., & Yarowsky, D. (2008). Translating Compounds by Learning Component Gloss Translation Models via Multiple Languages. *Proceedings of the 3rd International Joint Conference on Natural Language Processing* (pp. 403-410). IJCNLP 2008, Hyderabad, India.

Grefenstette, G. (1999). The world wide web as a resource for example-based machine translation tasks. *Proceedings of the ASLIB Conference on Translating and the Computer*, *21*.

Hauer, B., & Kondrak, G. (2011). Clustering Semantically Equivalent Words into Cognate Sets in Multilingual Lists. *Proceedings of the 5th International Joint Conference on Natural Language Processing* (pp. 865–873). IJCNLP 2011, Chiang Mai, Thailand.

Keenan, E. L., & Faltz, L. M. (1985). *Boolean semantics for natural language*. Dordrecht, Holland: D. Reidel.

Lardilleux, A. (2008). A truly multilingual, high coverage, accurate, yet simple, sub-sentential alignment method. *The 8th conference of the Association for Machine Translation in the Americas* (pp. 125-132). AMTA 2008, Waikiki, Honolulu, USA.

Li, H., & Xu, J. (2007). AdaRank: A Boosting Algorithm for Information Retrieval. *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval* (pp. 391-398). SIGIR '07, Amsterdam, The Netherlands.

Li, H. (2011). Learning to Rank for Information Retrieval and Natural Language Processing. *Synthesis Lectures on Human Language Technology, Lecture 12.* Morgan & Claypool Publishers.

Liu, T.-Y. (2009). WWW 2009 Tutorial on learning to rank for information retrieval. *18th International World Wide Web Conference*. WWW 2009, Madrid, Spain.

Manning, C., Raghavan, P., & Schütze, H. (2008). *Introduction to Information Retrieval.* Cambridge University Press. New York, NY, USA.

Martinet, A. (1979). *Grammaire fonctionnelle du français.* Crédif/Dider. Paris, France.

Metzler, D., & Croft, W. B. (2000). Linear feature-based models for information retrieval. *Information Retrieval*, *10*(3), 257-274.

Morin, E., & Daille, B. (2010). Compositionality and lexical alignment of multi-word terms. *Language Resources and Evaluation (LRE)*, Multiword expression: hard going or plain sailing (Springer Netherlands., Vol. 44, pp. 79–95). Rayson, P. and Piao, S. and Sharoff, S. and Evert, S. and Villada Moirón.

Morin, E., Daille, B., Takeuchi, K., & Kageura, K. (2007). Bilingual terminology mining - using brain, not brawn comparable corpora. *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics* (pp. 664-671). ACL 2007, Prague, Czech Republic.

Namer, F., & Baud, R. (2007). Defining and relating biomedical terms: Towards a cross-language morphosemantics-based system. *International Journal of Medical Informatics*, *76*(2-3), 226-33.

Porter, M. F. (1980). An algorithm for suffix stripping. *Program*, *14*(3), 130-137.

Quasthoff, U., Richter, M., & Biemann, C. (2006). Corpus Portal for Search in Monolingual Corpora. *Proceedings of the fifth international conference on Language Resources and Evaluation* (pp. 1799-1802). LREC 2006, Genoa, Italy.

Rapp, R. (1995). Identifying Word Translations in Non-Parallel Texts. *Proceedings of the 33rd Conference of the Association for Computational Linguistics* (pp. 320-322). ACL'95, Boston, Massachussets, USA.

Rauf, S., & Schwenk, H. (2009). On the use of comparable corpora to improve SMT performance. *Proceedings of the 12th Conference of the European Chapter of the ACL* (pp. 16-23). EACL 2009, Athens, Greece.

Robitaille, X., Sasaki, X., Tonoike, M., Sato, S., & Utsuro, S. (2006). Compiling French-Japanese terminologies from the Web. *Proceedings of the 11th Conference of the European Chapter of the Association for Computational Linguistics* (pp. 225-232). EACL'06, Trento, Italy.

Tiedemann, J. (2009). News from OPUS - A Collection of Multilingual Parallel Corpora with Tools and Interfaces. *Recent Advances in Natural Language Processing*, John Benjamins (Vol. V, pp. 237-248). Amsterdam, Philadelphia: N. Nicolov and K. Bontcheva and G. Angelova and R. Mitkov.

Weller, M., Gojun, A., Heid, U., Daille, B., & Harastani, R. (2011). Simple methods for dealing with term variation and term alignment. *Proceedings of the 9th International Conference on Terminology and Artificial Intelligence* (pp. 87-93). TIA 2011, Paris, France.

Wu, Q., Burges, J. C., Svore, K., & Gao, J. (2010). Adapting Boosting for Information Retrieval Measures. *Journal of Information Retrieval*, *13*(3), 254 - 270.

Twitter Topic Summarization by Ranking Tweets Using Social Influence and Content Quality

DUAN YaJuan[1*] *CHEN ZhuMin*[2] *WEI FuRu*[3]
ZHOU Ming[3] *Heung − Yeung SHUM*[4]
(1) University of Science and Technology of China, No. 96, Jinzhai Road, Hefei, Anhui, P.R.China
(2) Shandong University, No. 27, Shanda South Road, Jinan, Shandong, P.R.China
(3) Microsoft Research Asia, No. 5, Danling Street, Haidian District, Beijing, P.R.China
(4) Microsoft Corporation, Redmond, USA
dyj@mail.ustc.edu.cn, chenzhumin@sdu.edu.cn
{fuwei, mingzhou, hshum}@microsoft.com

ABSTRACT
In this paper, we propose a time-line based framework for topic summarization in Twitter. We summarize topics by sub-topics along time line to fully capture rapid topic evolution in Twitter. Specifically, we rank and select salient and diversified tweets as a summary of each sub-topic. We have observed that ranking tweets is significantly different from ranking sentences in traditional extractive document summarization. We model and formulate the tweet ranking in a unified mutual reinforcement graph, where the social influence of users and the content quality of tweets are taken into consideration simultaneously in a mutually reinforcing manner. Extensive experiments are conducted on 3.9 million tweets. The results show that the proposed approach outperforms previous approaches by 14% improvement on average ROUGE-1. Moreover, we show how the content quality of tweets and the social influence of users effectively improve the performance of measuring the salience of tweets.

TITLE AND ABSTRACT IN ANOTHER LANGUAGE (CHINESE)

基于用户社会影响力和文本质量的推特话题摘要

本文提出了一个基于时间轴的推特话题自动摘要框架。话题按照时间顺序分成子话题，各子话题中根据文本的重要度和多样性对推特文本进行排序和抽取以生成摘要。我们使用相互增强式图模型同时考虑文本内容、作者社会影响力和文本质量进行排序。实验结果表明：1）本文提出的模型在基准模型上ROUGE-1平均提高了14%；2）作者社会影响力和文本质量有效地改进了文本重要度的度量。

KEYWORDS: Topic summarization, Twitter, Timeline summarization, Content quality, Social influence.

KEYWORDS IN CHINESE: 话题摘要，推特，时间轴上的摘要，文本质量，用户社会影响力 .

[*] This work was done when the first author was an intern at Microsoft Research Asia.

Proceedings of COLING 2012: Technical Papers, pages 763–780,
COLING 2012, Mumbai, December 2012.

1 Introduction

Recently, Twitter [1] has become one of the most popular social networking sites. It enables people to freely post short messages (called tweets) up to 140 characters. Twitter has rapidly gained worldwide popularity, with over 140 million active users generating over 340 million tweets daily in March 2012 [2]. The rapid proliferation of Twitter posts presents a big obstacle for efficient information acquisition. It is impossible for a user to get an overview of important topics on Twitter by reading all tweets everyday. In addition, because of information redundancy and the informal writing style, it is time consuming to find useful information about a topic from a huge number of tweets. The tremendous volume of tweets suggests summarization as the key to facilitating the requirements of topic exploration, navigation, and search from hundreds of thousands of tweets. Specifically, a summary that provides representative information of topics with no redundancy and well-written sentences would be preferred.

In this paper, we focus on the problem of topic summarization in Twitter, which aims to provide a short and compact summary for a collection of tweets on the same or similar topics. We take individual tweets as the basic constituents to compose the summary. Here, tweets are to a certain extent analogous to sentences in traditional extractive document summarization, which has been extensively studied in past decades (Ani Nenkova, 2011). However, we argue that summarizing tweets is substantially different from summarizing news documents owing to the following reasons. First, tweets are streamed with detailed time-stamps delivering real-time information of users' continuous updates and comments that, provide temporal information for tracing topic evolution over time. This timeliness feature motivates us to summarize tweets along time line. Second, tweets are commonly expressed in an informal way. Only some of them obey standard grammar requirements, while others are written in arbitrary styles. For instance, tweets may contain many abbreviations, spelling errors, or information fragments. This requires the summarization algorithm to be aware of the content quality of the tweets when ranking and selecting salient tweets as summaries.

Furthermore, tweets are created on and spread through social networks. The authority of the author of the tweets, as well as the social networks (e.g. follower-followee relationship) of the author, usually plays an important role in demonstrating the salience of the tweets. People may be particularly interested in tweets from celebrities or opinion leaders. To be specific, if there are two tweets stating the same information, and one is published by an influential user and the other is not, we assume the former is more important than the latter on account of two interesting observations about Twitter users. First, users with a high influence have a larger audience. Their tweets are apt to be read by more users than those of non-influential users. Second, encouraged by the interactions with their followers, influential users are more likely to publish informative tweets of better readability, less error, and preferable completeness than common users. This guides us to develop a unified framework to simultaneously model the information from the content and the authors of tweets.

We therefore propose modeling and formulating tweet ranking in a unified mutual reinforcement graph, where the social influence (i.e. authority) of users and content quality of tweets are taken into consideration simultaneously in a mutually reinforcing manner. Particularly, we leverage the follower-followee relationship connecting different authors, upon which the social influence of users can be inferred. We define the content quality of tweets, including readability and

[1] http://twitter.com
[2] http://blog.twitter.com/2012/03/twitter-turns-six.html

content richness, as a measure of the regularity of written language and the pointless degree of the content. The above information is jointly employed in a graph-based ranking algorithm. In order to avoid redundancy in the result, the final summary is generated by selecting tweets from the previous ranking results with the traditional Maximal Marginal Relevance(MMR) algorithm (Carbonell and Goldstein, 1998). We conduct experiments on a real data set containing 3.9 million tweets. Compared with two popular graph-based summarization approaches, namely Lexrank (Erkan and Radev, 2004) and phrase graph (Sharifi et al., 2010a), the experimental results show that:

- The reinforcement summarization model integrating social influence and content quality achieves a considerable performance and outperforms the standard LexRank and the phrase graph summarization approaches.
- The social influence of users and the content quality of tweets help to more effectively measure the salience of tweets.

The rest of this paper is organized as follows. Related work is introduced in Section 2. Next, we present a detailed introduction of our approach in Section 3. Section 4 shows the experiments and results, and we conclude this work with directions for future study in section 5.

2 Related work

2.1 Extractive document summarization

A substantial amount of work has been done on extractive text summarization (Lloret and Palomar, 2012). Many text features, such as term frequency, sentence position, query relevance and sentence dependency structure, have been investigated for sentence salience estimation. They are usually weighted automatically by applying certain learning-based mechanisms or tuned experimentally to build a feature-based summarization system (Fuentes et al., 2007) (Wong et al., 2008). Previous research shows that a combination of sentence position, fixed-phrase and sentence length give the best results in learning-based sentence selection (Ani Nenkova, 2011). Meanwhile, feature-based approaches have been widely used in the top five participating systems in DUC[3] 2005-2007.

In addition, different types of links among sentences and documents are employed by graph-based approaches to measure sentence salience, such as LexRank (Erkan and Radev, 2004), TextRank (Mihalcea, 2004), and Mutual Reinforcement Chain(MRC) (Wei et al., 2008). LexRank and TextRank make use of pairwise similarity between sentences, hypothesizing that the sentences similar to most of the other sentences in a cluster are more salient. In contrast to the single level PageRank in LexRank and TextRank, MRC considers both internal and external constraints on three different levels, document, sentence, and term and achieves promising improvement.

2.2 Micro-blog summarization

Recently, researchers have conducted a number of investigations on micro-blog(e.g. Twitter) summarization. Instead of ranking sentences in traditional document summarization, micro-blog posts are ranked to select salient ones for the generation of topic-sensitive and query-sensitive summary. Both feature-based and graph-based approaches are exploited to measure the salience of posts under an extractive summarization framework. Taking into consideration

[3] http://www-nlpir.nist.gov/projects/duc/data.html

the evolutionary characteristic of topics along time line, researchers have also started to explore the evolutionary summarization of events in micro-blog.

In feature-based approaches, a variety of statistical and linguistic features have been extensively investigated, such as, language model (O'Connor et al., 2010), tweet frequency (Shiells et al., 2010), term frequency (Liu et al., 2011) (Takamura et al., 2011) (Parthasarathy, 2012), TF-IDF (Frederking, 2011) (Chakrabarti and Punera, 2011), hybrid TF-IDF (Sharifi et al., 2010b), KL-divergence (Zubiaga et al., 2012), time delay (Takamura et al., 2011), and topic relevance (Long et al., 2011). Among them, simple term frequency has proven to be extremely extraordinary for topic-sensitive micro-blog summarization because of the unstructured and short characteristics of micro-blog posts according to Inouye and Kalita (2011). As for micro-blog summarization, some micro-blog specific features such as text normalization, the content of shared web pages (Liu et al., 2011), and user behavior in conveying relevant content (Harabagiu and Hickl, 2011), have proven useful for result improvement.

The phrase graph algorithm is the most frequently studied graph-based approach in micro-blog summarization. Sharifi et al. (2010a), Beaux Sharifi and Kalita (2010), and Sharifi (2010) propose a phrase reinforcement summarization algorithm on leverage of trending phrase or phrases specified by a user in micro-blog posts. It achieves substantial improvements on ROUGE results by taking advantage of the link structure among words. Nichols et al. (2012) generate journalistic summary for events in world cup games by employing phrase graph algorithm only on the longest sentence in each tweet. Additionally, PageRank-like algorithms such as LexRank and TextRank have also been investigated by Inouye and Kalita (2011).

Evolutionary summarization approach segments post stream into event chains(usually along time line) and produces the final summary by incorporating the summary extracted for each event. A simple and effective method for detecting events from a post stream is to separate the stream according to the bursty period of post volume (Nichols et al., 2012) (Zubiaga et al., 2012). Long et al. (2011) provide an even particular separation according to the topical words that are recognized by their frequency in #hashtags and the entropy in the corpus. Chakrabarti and Punera (2011) argue that different types of events should be detected even when they are temporally close. They utilize a modified Hidden Markov Model to detect the event chain by learning the underlying hidden state representation of repeated events.

3 Topic summarization by time line with mutual reinforcement model

We begin this section with a formal definition of the work presented in this paper. We formulate the problem of topic summarization in Twitter as follows: given a topic τ depicted by a #hashtag [4], we can obtain a tweet collection $T = \{t_1, t_2, \cdots, t_N\}$ containing the #hashtag, where N is the number of tweets in a time span. Each tweet is attached with a time mark. We take a time-line approach to summarize T to fully capture the rapid topic evolution over time. Specifically, the summary for the topic consists of:

- a set of sub-topics $\omega = \{\omega_1, \omega_2, \cdots, \omega_K\}$ based on the distribution of tweets over time, where K is the number of sub-topics;
- at most M salient tweets as the summary for each sub-topic $\omega_i, 1 \leq i \leq K$.

[4] #hashtags are user-annotated topics in tweets.

There are three major steps generating a time-line summary for a topic. First, we conduct a topic segmentation that segments the tweet stream of the topic into sub-topic clusters in terms of the posting time, in which each cluster describes a sub-topic. Second, tweets in each sub-topic cluster are ranked according to tweet salience by a reinforcement ranking model taking advantage of the content quality of tweets and social influence of the authors. Third, we generate the summary for each sub-topic upon the tweet ranking results by removing the redundant tweets at the whole topic level. In the following sub-sections, we will present the three steps respectively.

3.1 Sub-topic segmentation

The key issue of sub-topic segmentation is to detect the breaking point of sub-topics in the tweet stream. We observe that in comparison to the internal portion of a sub-topic, some statistics of the stream make dramatic changes, called bursty, at breaking points, for example, tweet volumes, term frequency, participating users, and variety of #hashtags. Among them, term frequency bursty is most correlated to topic evolution according to our analysis. Therefore, we segment the sub-topics by detecting term frequency bursty. Typically, a sub-topic occurs associated with several words and covers their bursty period. We break sub-topic segmentation down to the bursty period identification of associated words and the lifetime detection for sub-topics.

First, we track every word in the stream to identify their bursty period. Terms without a bursty period will be discarded. We assume that the term frequency satisfies a binomial distribution in the bursty period (Fung et al., 2005).

$$tf(w) = \binom{I}{i} p(w)^i (1-p(w))^{I-i} \tag{1}$$

$$p(w) = \sum_{i=1}^{I} \frac{tf_i(w)}{Count_i(tweet)} \cdot \frac{1}{I}$$

Where I is the total length of time in days in the stream, $p(w)$ represents the bursty probability of word w, and $tf_i(w)$ is the term frequency of w at i^{th} time. $Count_i(tweet)$ denotes the number of tweets at i^{th} time. We compute the mean value of $tf(w)$. The period in which the term frequency is larger than $2 * mean$ is a bursty period of the corresponding word.

The lifetime of sub-topics is detected depending on the bursty period of associated words. Terms whose half bursty period overlap with each other are recognized as a set of sub-topic-associated words, denoted by w_a. The lifetime of the sub-topic covers a continuous period where

$$tf_i(w_a) > \alpha \cdot mean_a + \beta \cdot var_a \tag{2}$$

$$tf_i(w_a) = \sum\nolimits_{w \in w_a} tf_i(w)$$

where $tf_i(w_a)$ is the term frequency of associated words at i^{th} time. $mean_a$ and var_a denote the expectation and variance of term frequency respectively by blending words in associated words set during the overlapped time period. α and β adjust the length of the sub-topic lifetime, which are selected empirically. In the experiment, we set α as 1.4 and β as 0.4. The sub-topic consists of tweets published in the corresponding lifetime. Time periods contain no bursty words and those with an associated term frequency below the threshold are regarded as noisy periods and kept out from the later summary generation.

3.2 Mutual reinforcement model based sub-topic summarization

We define salient tweets as those similar to most of the tweets in the sub-topic, published by influential users, and presented in a good writing style. The first is consistent with the salience in document summarization. We emphasize the second because Twitter is a social network, in which influential users have a wider audience and more affirmative interaction with others. Tweets published by influential users are more likely to dominate the topic. The last is set against the informal writing style of Twitter.

Inspired by Wei et al. (2008), we propose a unified mutual reinforcement summarization model taking advantage of relations among tweets, words, and users for tweet salience measurement. Figure 1 shows the overview of the proposed model. The similarity of tweets to the sub-topic benefits from both the content similarity among tweets and the word coverage in the sub-topic cluster. In document summarization, the contribution of relationships among sentences to the performance improvements has been recognized (Wan et al., 2007). Sharifi et al. (2010a) found that sequences of words that encompassed the topic phrase highly overlapped when considering a large number of tweets for a single topic. The social influence of users contributes to salience measurement via author relation. And the content quality of tweets is incorporated at the tweet level.

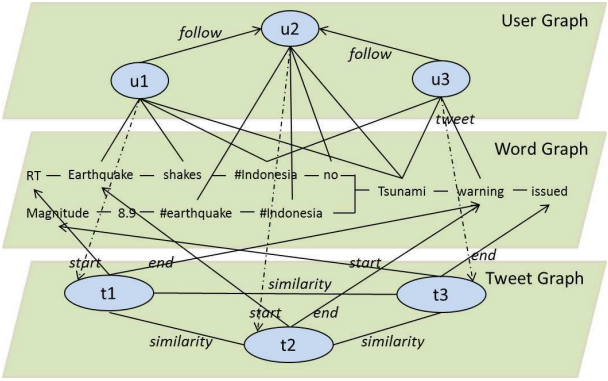


Figure 1: The Unified Mutual Reinforcement Graph Model

The mutual reinforcement model is formed with three PageRank-like models for word, tweet, and user respectively, but in a unified and interrelated way. The ranking of one of them is derived not only from the relationships with instances of itself, but is also affected by the other two. In the model, tweets connect to each other through syntactic similarities. If two tweets have a non-zero cosine similarity, there is a link between them. Words are linked through their co-occurrence in the same tweet. And users are naturally associated by following-followee relationship. If user u_i follows user u_j, a direct edge from u_i to u_j is created. Furthermore, we create edges among users, words, and tweets through authorship and consisting relationship. If user u_i published tweet t_j, we connect u_i with t_j and all words in t_j according to authorship. And t_j is linked to every word it consists of. In Figure 1, we illustrate the connection between

tweet and words through *start* and *end* links to avoid too many lines. *Start* specifies the first word of a tweet and *end* denotes the last word.

3.2.1 Mutual reinforcement based tweet ranking

We can then formulate the ranking algorithm for the mutual reinforcement model.

A tweet is salient if it is published by an influential user, written in regular language style, contains important terms and connects to other salient tweets. The ranking score of a tweet in sub-topic collection ω_k is defined as,

$$\begin{aligned} Score^{(r+1)}(t_i) = \alpha_1 \cdot [(1-d) \cdot \frac{quality(t_i)}{\sum_{t_j \in \omega_k} quality(t_j)} + d \cdot \sum_{t_j \in adj[t_i]} \frac{Sim(t_i, t_j)}{\sum_{t \in adj[t_j]} Sim(t_j, t)} \cdot Score^{(r)}(t_j)] \\ + \beta_1 \cdot \sum_{w \in t_i} Score^{(r)}(w) + \gamma_1 \cdot Score^{(r)}(u_i) \end{aligned} \tag{3}$$

where $Score^{(r)}(t)$, $Score^{(r)}(w)$, and $Score^{(r)}(u)$ denote the ranking score of tweet t, word w and user u in r^{th} iteration respectively. $adj[t_i]$ represents tweets connecting to t_i directly, $Sim(t_i, t_j)$ denotes the cosine similarity between t_i and t_j, u_i is the author of t_i, and $quality(t_i)$ refers to the content quality of t_i which will be detailed in Section 3.2.2. In order to assign high scores for tweets of good quality, we diversify the random walk probabilities (Brin and Page, 1998) of tweets using content quality. High quality tweets will be selected at high probabilities, while low quality ones are selected at low probabilities. Here, d is the damping factor, set to 0.85 as described in Brin and Page (1998).

A user is influential if he publishes salient tweets, uses important terms, and connects to other influential users. The ranking score of a user is defined as,

$$\begin{aligned} Score^{(r+1)}(u_i) = \alpha_2 \cdot \sum_{t \in \omega_k \cap T_{u_i}} Score^{(r)}(t) + \beta_2 \cdot \sum_{w \in t, t \in \omega_k \cap T_{u_i}} Score^{(r)}(w) \\ + \gamma_2 \cdot [(1-d) \cdot \frac{F_{static}(u_i)}{\sum_{u \in U_{\omega_k}} F_{static}(u)} + d \cdot \sum_{u_j \in flw[u_i]} \frac{1}{|frd[u_j]|} \cdot Score^{(r)}(u_j)] \end{aligned} \tag{4}$$

where T_{u_i} denotes the tweets published by u_i, and U_{ω_k} refers to all users who published tweets in ω_k. $flw[u_i]$ represents followers of u_i, and $frd[u_j]$ refers to the users u_j follows. $F_{static}(u_i)$ is regarded as a topic-irrelevant static influence of user u_i. It is predicted by a linear SVM model using features derived from several statistics, including the number of followers, the number of messages, the number of lists, the follower/following ratio, the zombie followers in proportion to all followers, and the number of mentions and retweets the user received. Similar to content quality, we use the static influence of users to differentiate the random walk probabilities of users with the purpose of assigning high scores for users with high static influence.

A phrase term is important if it is contained in salient tweets, used by an influential user, and connects to other important terms.

$$\begin{aligned} Score^{(r+1)}(w_i) = \alpha_3 \cdot \sum_{t \in \omega_k \cap T_{w_i}} Score^{(r)}(t) + \gamma_3 \cdot \sum_{u \in U_{\omega_k} \cap U_{w_i}} Score^{(r)}(u) \\ + \beta_3 \cdot [(1-d) \cdot \frac{df(w_i)}{\sum_{w_j \in t, t \in \omega_k} df(w_j)} + d \cdot \sum_{w_j \in adj[w_i]} \frac{1}{|adj[w_j]|} \cdot Score^{(r)}(w_j)] \end{aligned} \tag{5}$$

where T_{w_i} denotes tweets containing word w_i and U_{w_i} refers to users who used w_i. $df(w_i)$ is the document frequency of term w_i. And $adj[w_i]$ represents the words connnecting to w_i. The random walk probability of terms is initialized by their normalized document frequency with the purpose of assigning high scores for words with high document frequency. α_i, β_i, and γ_i are used to balance the relative weight of tweets, words, and users. They are selected in accordance with Wei et al. (2008)'s work. Wei et al. (2008) proved that the three level unified mutual reinforcement model will converge on unique $Score(t_i)$, $Score(u_i)$, and $Score(w_i)$. We will not repeat the demonstration here. Finally, a ranked tweet list $\omega_k^{'}$ is formed by ranking tweets in ω_k according to the score assigned by the model.

3.2.2 Content quality estimation

We employ a logistic regression model to estimate the content quality used in Section 3.2.1. The content quality of tweets is estimated according to two aspects: readability and content richness. The former measures how easy a tweet is to read. In particular, tweets have high readability if they are written in regular language style using ordinary vocabulary, and have low readability if not. The latter quantifies how much useful information a tweet contains. Instead of calculating the absolute readability and the content richness, we learn models to compare the values of two tweets. Given tweets t_i and t_j, we set their relative quality as

$$quality(t_i, t_j) = \begin{cases} 1, quality(t_i) > quality(t_j) \\ 0, quality(t_i) = quality(t_j) \\ -1, quality(t_i) < quality(t_j) \end{cases} \quad (6)$$

Naturally, for a given group of tweets, if the relative quality of any two tweets has been identified, a ranking list of those tweets with respect to the quality can be generated, and vice visa. Consequently, the relative quality estimation problem is converted into quality ranking. We learn a unified logistic regression model to rank the quality of tweets according to their regression score. The model considers readability and content richness simultaneously. Table 1 and Table 2 present the feature set we used.

Feature	Description
OOV words	The proportion of OOV words in tweet
#hashtags	The proportion of #hashtags in tweet
Mentions	The number of user names appearing in the tweet
Capital letter	Normalized length of capital letters fraction
Punctuation	Normalized length of punctuation character
Emoticon	The number of emoticons
Ellipsis	The number of ellipsis
Stop word	Normalized length of stop word fraction
Character per word	Average character per word
Length	Number of characters in tweet

Table 1: Features for readability estimation

Feature	Description
Celebrity	Normalized word length of celebrities
URL	Share URL or not
URL rank	Alexa rank of shared URL
InfoToNoise	Normalized word length after removing stop word
Word length	The number of words
NE length	Normalized word length of named entities

Table 2: Features for content richness estimation

3.3 Summary generation by removing redundancy

Finally, we will generate a summary for each sub-topic from its corresponding ranked tweets. A straightforward method would be to select the top-ranked tweets of the sub-topics. However, this method would generate a redundant summary both at the tweet and sub-topic levels. At the tweet level, using a similarity link between tweets when measuring the salience of a tweet leads to close ranking scores being assigned to similar tweets. Tweets with high similarity may be chosen simultaneously. At the sub-topic level, the tweet stream is segmented into sub-topics and noisy tweet collections along time line. Two sub-topics describing the same content may be separated by several noisy collections. To avoid redundancy, we employ the MMR algorithm to generate the final summary.

$$S_k = \underset{t_i \in \omega_k' \setminus S_k}{argmax} [\lambda \cdot Sim_1(t_i, w_{a,k}) - (1-\lambda) \cdot \underset{t_j \in S_k \cap S_{k-1}}{max} Sim_2(t_i, t_j)] \quad (7)$$

Where S_k is the summary generated for sub-topic ω_k. Sim_1 represents the cosine similarity between current tweets and the associated terms $w_{a,k}$ of ω_k. And Sim_2 is the cosine similarity between current tweet and tweets selected in S_k and S_{k-1}, based on the observation that a topic changes continuously. λ is the weight emphasizing the importance of the two types of similarities, which is set as 0.5 in the experiment.

4 Experiment

4.1 Data set

We take earthquake event as an example in our experiment. Note that our approach does not leverage topic (e.g. earthquake) specific features. It can generate summary for topics beyond earthquake as well. We obtain 12.7 million English tweets containing the keyword *earthquake* published between September 2010 and April 2012 using the public Twitter API [5]. As #hashtags are regarded as user annotated topics in Twitter, we select the 30 most popular #hashtags related to earthquake as the topics from the collected tweet corpus. Then, we pick out tweets relevant to each topic using two methods of keyword matching. First, tweets containing the corresponding #hashtag are chosen. Second, if the #hashtag is a compound word concatenated of several words, it split into a phrase. Tweets containing the phrase are also selected. In total, we obtain 3.9 million tweets for the 30 topics, which are segmented into 84 sub-topics. Each sub-topic contains 66,416 tweets on average because of the co-occurrence of the #hashtags.

We ask two Ph.D. students(who have not participated in this work) to label 10 salient tweets as

[5] http://windowsphone.uservoice.com

reference for each sub-topic. Given that it is fairly difficult to select 10 tweets from thousands, we filter some tweets beforehand if they satisfy one of the following conditions:

- The length of the word is less than 3.
- It contains no words other than topic words, keyword *earthquake* and stop words.
- It contains no words other than URLs

After filtering, we are left with 26,874 tweets on average for each sub-topic. To make it easy for labeling, the remaining tweets are ranked according to syntactic similarity with the sub-topic. We further remove repetition in the ranking list in case different users post identical tweets. Finally, there are 10,616 tweets for each sub-topic in the ranking list. Annotators select the top tweet in the ranking list, and decide if it should be added to the summary, until 10 tweets have been chosen. We merge their annotation by maintaining the same tweets as the final reference summary. The inter-annotator agreement of annotation is 73%.

In the experiment, we remove spam tweets and extremely short tweets to clear the data. Some users post very similar tweets for certain kinds of promotion. For example, identical content was posted with different URLs attached for a backpack promotion. If a user continuously posts tweets with a content similarity of more than 0.9, all those tweets published by him or her are regarded as spam tweets and removed. We also ignore extremely short tweets less than 3 words in length because they are too short to deliver complete information. In addition, to get a better estimation of similarity between tweets, we conduct some preprocessing before the similarity computation: 1) all words in tweets are stemmed; 2) the topic terms and keyword *earthquake* are neglected due to their occurrence in nearly every tweet; 3) some functional #hashtags and news agencies such as #fb and #BBC are excluded because they are usually irrelevant to the topics and only provide an information source; 4) the stop words are removed, including standard stop words and stop word abbreviations commonly used in Twitter, for instance, *you are* shortens to *ur*.

In the following sections, we present the experimental results on a real-life data set to verify the effectiveness of our summarization approach. We conduct four experiments to evaluate the performance of the reinforcement summarization model, the contribution of social influence and content quality to the model, the correctness of sub-topic segmentation, and the accuracy of the content quality estimation model. The performance of the social influence of users has not been evaluated due to the difficulty of data acquisition. It is hard to label the influence of users, or to collect such information automatically. Therefore, in this paper, we evaluate the end-to-end contribution of social influence under the whole summarization framework.

4.2 Evaluation metric

We evaluate the performance of our summarization system using ROUGE (Lin and Hovy, 2003), which is widely-used in summarization evaluation. It measures the overlap of N-grams between the predicted summary and the reference, which is defined as,

$$ROUGE = \frac{\Sigma_{t \in S_{ref}} \Sigma_{gram_n \in t} Count_{match}(gram_n)}{\Sigma_{t \in S_{ref}} \Sigma_{gram_n \in t} Count(gram_n)}$$

where n is the word length of n-gram, S_{ref} denotes the reference summary, $Count(gram_n)$ is the number of n-grams comprising sentences in the reference summary and $Count_{match}(gram_n)$ computes the maximum number of n-grams appearing both in the summary generated by our system and the reference summary.

4.3 Evaluation of the reinforcement model

In order to evaluate the effectiveness of the reinforcement model, we construct two baselines, namely the phrase graph model (Sharifi et al., 2010a) and the LexRank algorithm (Erkan and Radev, 2004). Both of them are graph-based algorithms, homologous with the reinforcement model. And they are currently the most popular two graph-based algorithms for summarization. Phrase graph measures sentence similarity depending on word frequency as well as the word distance from the topical word in a sentence. LexRank is a PageRank-like summarization algorithm, which calculates sentence salience using the random walk model. Table 3 shows the comparison results of our approach and the baseline methods in terms of ROUGE-1 values. In

Algorithm	ROUGE-1
Phrase graph	0.4286
LexRank	0.3865
Our approach	**0.4617**

Table 3: A comparison of three summarization algorithms[6]

the experiment, α_i, β_i, and γ_i are set as $[\alpha_1, \alpha_2, \alpha_3] = [1, 0.5, 0.25]$, $[\beta_1, \beta_2, \beta_3] = [0.5, 1, 0.5]$, and $[\gamma_1, \gamma_2, \gamma_3] = [0.25, 0.5, 1]$, the same as that in Wei et al. (2008). As seen from Table 3, the proposed approach outperforms both the LexRank and phrase graph methods. It obtains a relative improvement of 20% and 8% compared with the two baselines respectively. We also find that LexRank and phrase graph tend to assign high salience scores to long sentences containing several #hashtags. Some of them are pointless tweets and irrelevant to the topic. For example: *@lightskintess77 =o you could check to see if you had an earthquake by looking it up here: 4 minutes ago #TAB##TAB#5.8 #TAB##TAB#Virginia*. It contains #hashtag *#Virginia*, but the content is not very relevant to the main topic *Virginia earthquake*. It is more like a personal message.

Because LexRank accumulates the information from neighboring sentences (or tweets), its similarity estimation mainly depends on commonly used words, not words with very high IDF scores, especially in short sentences, which increase the chances of selecting pointless information such as personal messages. In comparison to LexRank, phrase graph performs better because it computes the salience of tweets relying on the topical words in the cluster and takes word order into consideration.

Pointless and low quality tweets hurt the performance of both of the baseline models. In comparison to them, our proposed reinforcement model incorporating content quality and social influence emphasizes tweet salience by not only considering the similarity, but also the influence of its author, its readability, and its content richness. These factors penalize the pointless tweets and tweets of low quality when measuring their salience and achieve substantial results.

4.4 Evaluation of social influence and content quality for summarization

We further investigate the effectiveness of the social influence of authors and the content quality of tweets for tweet summarization in detail. To verify the usefulness of content quality, we test

[6]Our approach outperforms LexRank with a significance level of p=0.07. The improvement on Phrase graph is not significant

its contributions in the reinforcement model, LexRank, and phrase graph separately. We remove content quality from the reinforcement model by replacing the random walk probabilities of tweets with a unified weight. The more the performance decreases, the more useful the content quality is. Meanwhile, the content quality is integrated into LexRank and phrase graph. In LexRank, we initialize the random walk probability with the quality score of tweets. And in phrase graph, we change the weight of each word into the weighted summation of its term frequency in every tweet it appears, with the content quality of the tweet as the weight. Table 4 shows the results.

Similar to content quality, we evaluate the contribution of social influence by removing it from the reinforcement model. The salience of tweets is then inferred in the two-level PageRank-like iteration derived from related tweets and words. γ_1, γ_2, γ_3, α_2, and β_2 in section 3.2.1 are set as zero. We set $[\alpha_1,\alpha_3] = [1,0.5]$, and $[\beta_1,\beta_3] = [0.5,1]$ according to Wei et al. (2009). We have not integrated social influence into the LexRank and phrase graph as verification due to the high complexity. Table 5 shows the results.

Model	ROUGE-1
Reinforcement model	0.4617
w/o content quality	**0.4503(-2.5%)**
LexRank	0.3865
LexRank + content quality	**0.3956(+2.4%)**
Phrase graph	**0.4286**
Phrase graph + content quality	0.3959(-7.7%)

Table 4: Effectiveness of content quality

Model	ROUGE-1
Reinforcement model	0.4617
w/o social influence	**0.4310(-6.6%)**

Table 5: Effectiveness of social influence

We can see from Table 4 that the content quality of tweets proves useful in both the reinforcement model and LexRank. The ROUGE-1 is improved by about 2.4% by integrating content quality. We observe in the result that tweets selected by every model pair overlap except for individual pointless tweets. For example, *OMG! This footage on TV now of the earthquake and ensuing tsunami in #Japan is so crazy. My thoughts are with everyone there! Stay strong!*. Content quality excludes such pointless tweets from the final summary by reducing the transition probability. However, the ROUGE-1 in the phrase graph model decreases after it is combined with content quality. It is probably because of the rough assignment of the quality score to words. The content quality of tweet is a measurement based on all words contained in the tweet. It is not so fair to distribute this quality to every word equally.

Table 5 indicates that social influence is an important part of the reinforcement model. The performance decreases rapidly in ROUGE-1 after removing it. The comparison between the results(before and after removing social influence) suggests that more tweets posted by influential users are selected by the former.

4.5 Evaluation of sub-topic segmentation

It is difficult for editors to segment a tweet stream containing hundreds of thousands of tweets into several sub-topics. We have noticed that each sub-topic is a cluster of tweets. Adjacent sub-topics are required to focus on different subjects, which make it a special text clustering task. Thus, we treat each sub-topic as a cluster and automatically compute the average cosine similarity between adjacent ones as an indicator of segmentation performance. The lower the average cosine similarity is, the better performance the segmentation achieves. Table 6 shows the results.

Approach	Cosine similarity
Uniform segmentation	0.835
Nichols et al. (2012)	0.674
Ours	**0.670**

Table 6: Evaluation of sub-topic segmentation

We compare our method with uniform segmentation and the method in Nichols et al. (2012). Uniform segmentation separates the tweet stream into identical length time slots. Tweets in each time slot form a sub-topic. The number of sub-topics equals the number generated by our method. Nichols et al. segment the tweet stream by detecting the extreme changes in update volume per minute. The moment in which the update volume per minute exceeds 3*median(update volume) is a segment. In the experiment, 2.8 sub-topics are produced on average for each topic by our method, and 3 by Nichols et al. (2012). From Table 6 we can see that our method achieves comparable result to Nichols et al. (2012). Both of them lead to lower average cosine similarities between adjacent sub-topics in comparison to uniform segmentation.

4.6 Evaluation of content quality

In this sub-section, we evaluate the performance of the content quality classifiers. We randomly select 3,000 tweets from an additional tweet corpus for content quality training and evaluation. One of two labels, H(high quality) and L(low quality), are assigned to every tweet. Tweets that have good readability and cover rich content are labeled H, otherwise L. A tweet is regarded as having good readability if it does not include too many abbreviations, ellipses, spelling errors, #hashtags, or mentions. In addition, a tweet is viewed as containing rich content if it 1) describes an event or an interesting topic, 2) contains crisp, clear, and effective text that is easy to understand, 3) provides some insight about the event or topic beyond simply stating that it occurred, or 4) reflects a useful opinion on some topics. We learn two logistic regression models using features described in section 3.2.2 and unigrams respectively, denoted as LRO and LRU. Five-fold cross-validation is conducted to evaluate the performance. Table 7 shows the results as well as the result of the Random classification model.

The LRO model obtains promising results for both high and low quality tweets. It achieves substantial improvements in comparison to the LRU model and random classification model. The result of random classification reflects the imbalanced distribution of data. Tweets of high quality in a tweet stream are rare, leading to a proportion of 13% in the annotated data. Low quality tweets account for 87%, which makes it difficult to achieve a very good prediction. However, our concern is the performance on tweets of high quality because we want to select good tweets for summary generation. The precision of high quality tweets increases by 0.429

Method	Data	Precision	Recall	F-score	Accuracy
Random	H	0.130	0.500	0.206	0.500
	L	0.870	0.500	0.635	
LRU	H	0.321	0.636	0.427	0.765
	L	0.931	0.786	0.852	
LRO	H	**0.559**	**0.714**	**0.627**	**0.883**
	L	**0.952**	**0.910**	**0.931**	

Table 7: Evaluation of Content Quality

and 0.238 separately, which dramatically increases the probability of high quality tweets being selected.

A deep analysis of the features reveals several interesting facts. Whether a tweet includes a URL is highly valued by the model. This is mainly because tweets with URLs are usually news titles or shared information, such as videos and games from corresponding websites. They are written in formal language and deliver rich information. Another highly useful feature is the number of ellipsis. Ellipsis is used frequently in Twitter. They are usually used to leave out a part of sentences, or as a break between two irrelevant sentences. Tweets involving more than one ellipsis become difficult to read. In addition, the word length of tweets, the number of emoticons, and the stop word ratio are useful for measuring content quality.

4.7 Discussion

To make it easier for annotators to label the reference, we rank all the tweets before annotation. The ranking procedure used in selecting the reference tweets makes tweets most similar to the corpus have a higher priority for selection. However, rank is not the only aspect the annotators considered (e.g. tweets with low readability or content richness will not be selected). It is just used to reduce the annotation workload. Annotators go through the top N tweets in the ranked list until 10 reference tweets are selected (N may be 100, 1000, or more). A summarization algorithm considering similarity will benefit from this corpus. But both the mutual reinforcement model and the two baselines have not integrated the heuristics to rank tweets to create the corpus. They are compared in fair head start.

5 Conclusion

In this paper, we propose summarizing tweet streams with regard to topics along time line to produce an overview of topic evolution, which is expressed by sub-topics in chronological order. For each sub-topic, a set of salient tweets is selected to produce the summary by ranking them according to salience. Different from traditional documents, tweets suffer a great deal from pointless information and irregular writing style. We thus model the salience of a tweet by using a unified mutual reinforcement graph to incorporate the social influence of users and the content quality of tweets. The experimental results show that the proposed approach achieves substantial improvements in comparison to LexRank and phrase graph. Furthermore, the content quality and the social influence show great effectiveness in measuring the salience of tweets. In the future, we plan to exploit more specific relationships among tweets, such as retweeting in the reinforcement model to rank tweets for summary generation.

References

Ani Nenkova, K. M. (2011). *Automatic Summarization*, pages 103–233.

Beaux Sharifi, M.-A. H. and Kalita, J. (2010). Automatic summarization of twitter topics. In *National Workshop on Design and Analysis of Algorithm*.

Brin, S. and Page, L. (1998). The anatomy of a large-scale hypertextual web search engine. In *Proceedings of the seventh international conference on World Wide Web 7*, WWW7, pages 107–117, Amsterdam, The Netherlands, The Netherlands. Elsevier Science Publishers B. V.

Carbonell, J. and Goldstein, J. (1998). The use of mmr, diversity-based reranking for reordering documents and producing summaries. In *In Research and Development in Information Retrieval*, pages 335–336.

Chakrabarti, D. and Punera, K. (2011). Event summarization using tweets. In *ICWSM'11*, pages –1–1.

Erkan, G. and Radev, D. R. (2004). Lexrank: graph-based lexical centrality as salience in text summarization. *J. Artif. Int. Res.*, 22(1):457–479.

Frederking, K. D. R. R. S. B. L. A. G. R. (2011). Topical clustering of tweets. In *Proceedings of the ACM SIGIR 3rd Workshop on Social Web Search and Mining*.

Fuentes, M., Alfonseca, E., and Rodríguez, H. (2007). Support vector machines for query-focused summarization trained and evaluated on pyramid data. In *Proceedings of the 45th Annual Meeting of the ACL on Interactive Poster and Demonstration Sessions*, ACL '07, pages 57–60, Stroudsburg, PA, USA. Association for Computational Linguistics.

Fung, G. P. C., Yu, J. X., Yu, P. S., and Lu, H. (2005). Parameter free bursty events detection in text streams. In *Proceedings of the 31st international conference on Very large data bases*, VLDB '05, pages 181–192. VLDB Endowment.

Harabagiu, S. M. and Hickl, A. (2011). Relevance modeling for microblog summarization. In Adamic, L. A., Baeza-Yates, R. A., and Counts, S., editors, *ICWSM*. The AAAI Press.

Inouye, D. and Kalita, J. K. (2011). Comparing twitter summarization algorithms for multiple post summaries. In *SocialCom/PASSAT*, pages 298–306. IEEE.

Lin, C.-Y. and Hovy, E. (2003). Automatic evaluation of summaries using n-gram co-occurrence statistics. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology - Volume 1*, NAACL '03, pages 71–78, Stroudsburg, PA, USA. Association for Computational Linguistics.

Liu, F., Liu, Y., and Weng, F. (2011). Why is "sxsw" trending?: exploring multiple text sources for twitter topic summarization. In *Proceedings of the Workshop on Languages in Social Media*, LSM '11, pages 66–75, Stroudsburg, PA, USA. Association for Computational Linguistics.

Lloret, E. and Palomar, M. (2012). Text summarisation in progress: a literature review. *Artif. Intell. Rev.*, 37(1):1–41.

Long, R., Wang, H., Chen, Y., Jin, O., and Yu, Y. (2011). Towards effective event detection, tracking and summarization on microblog data. In *Proceedings of the 12th international conference on Web-age information management*, WAIM'11, pages 652–663, Berlin, Heidelberg. Springer-Verlag.

Mihalcea, R. (2004). Graph-based ranking algorithms for sentence extraction, applied to text summarization. In *Proceedings of the ACL 2004 on Interactive poster and demonstration sessions*, ACLdemo '04, Stroudsburg, PA, USA. Association for Computational Linguistics.

Nichols, J., Mahmud, J., and Drews, C. (2012). Summarizing sporting events using twitter. In *Proceedings of the 2012 ACM international conference on Intelligent User Interfaces*, IUI '12, pages 189–198, New York, NY, USA. ACM.

O'Connor, B., Krieger, M., and Ahn, D. (2010). Tweetmotif: Exploratory search and topic summarization for twitter. In *ICWSM'10*, pages –1–1.

Parthasarathy, X. Y. A. G. Y. R. S. (2012). A framework for summarizing and analyzing twitter feeds. In *In Proceedings of the ACM International Conference on Knowledge Discovery and Data Mining*, SIGKDD.

Sharifi, B., Hutton, M.-A., and Kalita, J. (2010a). Summarizing microblogs automatically. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, HLT '10, pages 685–688, Stroudsburg, PA, USA. Association for Computational Linguistics.

Sharifi, B., Hutton, M.-A., and Kalita, J. K. (2010b). Experiments in microblog summarization. In *Proceedings of the 2010 IEEE Second International Conference on Social Computing*, SOCIALCOM '10, pages 49–56, Washington, DC, USA. IEEE Computer Society.

Sharifi, B. P. (2010). Automatic microblog classification and summarization. In *Master's thesis*.

Shiells, K., Alonso, O., and Lee, H. J. (2010). Generating document summaries from user annotations. In *Proceedings of the third workshop on Exploiting semantic annotations in information retrieval*, ESAIR '10, pages 25–26, New York, NY, USA. ACM.

Takamura, H., Yokono, H., and Okumura, M. (2011). Summarizing a document stream. In *Proceedings of the 33rd European conference on Advances in information retrieval*, ECIR'11, pages 177–188, Berlin, Heidelberg. Springer-Verlag.

Wan, X., Yang, J., and Xiao, J. (2007). Towards an iterative reinforcement approach for simultaneous document summarization and keyword extraction. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 552–559, Prague, Czech Republic. Association for Computational Linguistics.

Wei, F., Li, W., Lu, Q., and He, Y. (2008). Query-sensitive mutual reinforcement chain and its application in query-oriented multi-document summarization. In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '08, pages 283–290, New York, NY, USA. ACM.

Wei, F., Li, W., Lu, Q., and He, Y. (2009). Applying two-level reinforcement ranking in query-oriented multidocument summarization. *J. Am. Soc. Inf. Sci. Technol.*, 60(10):2119–2131.

Wong, K.-F., Wu, M., and Li, W. (2008). Extractive summarization using supervised and semi-supervised learning. In *Proceedings of the 22nd International Conference on Computational Linguistics - Volume 1*, COLING '08, pages 985–992, Stroudsburg, PA, USA. Association for Computational Linguistics.

Zubiaga, A., Spina, D., Amigó, E., and Gonzalo, J. (2012). Towards real-time summarization of scheduled events from twitter streams. In *Proceedings of the 23rd ACM conference on Hypertext and social media*, HT '12, pages 319–320, New York, NY, USA. ACM.

S-restricted monotone alignments: Algorithm, search space, and applications

Steffen Eger
Goethe University Frankfurt, Grueneburg-Platz 1, 60323 Frankfurt am Main, Germany
steffen.eger@yahoo.com

Abstract
We present a simple and straightforward alignment algorithm for monotone many-to-many alignments in grapheme-to-phoneme conversion and related fields such as morphology, and discuss a few noteworthy extensions. Moreover, we specify combinatorial formulas for monotone many-to-many alignments and decoding in G2P which indicate that exhaustive enumeration is generally possible, so that some limitations of our approach can easily be overcome. Finally, we present a decoding scheme, within the monotone many-to-many alignment paradigm, that relates the decoding problem to restricted integer compositions and that is, putatively, superior to alternatives suggested in the literature.

Title and Abstract in German

S-beschränkte monotone Alignierungen: Algorithmus, Suchraum und Anwendungen

Wir präsentieren einen einfachen Alignment-Algorithmus für monotone 'many-to-many' Alignierungen im Bereich Graphem-zu-Phonem-Konversion und verwandten Gebieten wie z.B. Morphologie, und besprechen sinnvolle Erweiterungen. Darüber hinaus geben wir kombinatorische Formeln im Bereich der monotonen 'many-to-many' Alignierungen und im Bereich des Decoding in G2P an, die suggerieren, dass vollständige Enumeration hier im Allgemeinen möglich ist, sodass ein paar Einschränkungen unseres Ansatzes leicht behoben werden können. Schließlich präsentieren wir ein Decoding-Schema, innerhalb des Paradigmas von 'many-to-many' Alignierungen, dass das Decoding-Problem mit beschränkten Zahlenkompositionen in Beziehung setzt und das in der Literatur vorgeschlagenen Alternativen vermeintlich überlegen ist.

Keywords: many-to-many alignments, monotone alignments, string transduction, restricted integer compositions, grapheme-to-phoneme conversion.

Keywords in German: many-to-many Alignierungen, monotone Alignierungen, String-Überführungen, beschränkte Zahlenkompositionen, Graphem-zu-Phonem-Konversion.

Proceedings of COLING 2012: Technical Papers, pages 781–798,
COLING 2012, Mumbai, December 2012.

1 Introduction

Grapheme-to-phoneme conversion (G2P) is the problem of transducing, or converting, a grapheme, or letter, string $\mathbf{x}$ over an alphabet Σ_x into a phoneme string $\mathbf{y}$ over an alphabet Σ_y. A crucial first step thereby is finding *alignments* between grapheme and phoneme strings in training data. The classic alignment paradigm has assumed alignments that were

(i) *one-to-one* or *one-to-zero*; i.e. one grapheme character is mapped to at most one phoneme character; this assumption has probably been a relic of both the traditional assumptions in machine translation (cf. (Brown et al., 1990)) and in biological sequence alignment (cf. (Needleman and Wunsch, 1970)). In the field of G2P such alignment models are also called ϵ-scattering models (cf. (Black et al., 1998)).

(ii) *monotone*, i.e., the order between characters in grapheme and phoneme strings is preserved.

It is clear that, despite its benefits, the classical alignment paradigm has a couple of limitations; in particular, it may be unable to explain certain grapheme-phoneme sequence pairs, a.o. those where the length of the phoneme string is greater than the length of the grapheme string such as in

exact igzækt

where $\mathbf{x}$ has length 5 and $\mathbf{y}$ has length 6. In the same context, even if an input pair can be explained, the one-to-one or one-to-zero assumption may lead to alignments that, linguistically, seem nonsensical, such as

p	h	o	e	n	i	x
f	–	iː	n	i	k	s

where the reader may verify that, no matter where the ϵ is inserted, some associations will always appear unmotivated. Moreover, monotonicity appears in some cases violated as well, such as in the following,

centre sentər

where it seems, linguistically, that the letter character *r* corresponds to phonemic r and graphemic word final *e* corresponds to ə.

Fortunately, better alignment models have been suggested to overcome these problems. For example, (Jiampojamarn et al., 2007) and (Jiampojamarn and Kondrak, 2010) suggest 'many-to-many' alignment models that address issue (i) above. Similar ideas were already present in (Baldwin and Tanaka, 1999), (Galescu and Allen, 2001) and (Taylor, 2005). (Bisani and Ney, 2008) likewise propose many-to-many alignment models; more precisely, their idea is to *segment* grapheme-phoneme pairs into non-overlapping parts ('co-segmentation'), calling each segment a *graphone*, as in

ph	oe	n	i	x
f	iː	n	i	ks

which consists of five graphones.

The purpose of the present paper is to introduce a simple, flexible and general monotone many-to-many alignment algorithm (in Section 3) that competes with the approach suggested in (Jiampojamarn et al., 2007).[1] Thereby, our algorithm is an intuitive and straightforward generalization of the classical Needleman-Wunsch algorithm for (biological or linguistic) sequence alignment. Moreover, we explore valuable extensions of the presented framework, likewise in Section 3, which may be useful e.g. to detect latent classes in alignments, similar to what has been done in e.g. (Dreyer et al., 2008). We also mention limitations of our procedure, in Section 4, and discuss the naive brute-force approach, exhaustive enumeration, as an alternative; furthermore, by specifying the search space for monotone many-to-many alignments, we indicate that exhaustive enumeration appears generally a feasible option in G2P and related fields. Next, in Section 6.1 we briefly mention how we perform training for string transductions in the monotone many-to-many alignment case. Then, a second contribution of this work is to suggest an alternative decoding procedure when transducing strings $\mathbf{x}$ into strings $\mathbf{y}$, within the monotone many-to-many alignment paradigm (in Section 6.2). We thereby relate the decoding problem to restricted integer compositions, a field in mathematical combinatorics that has received increased attention in the last few years (cf. (Heubach and Mansour, 2004), (Malandro, 2012), (Eger, 2012a)). Finally, we demonstrate the superiority of our approach by applying it to several data sets in Section 7.

It must be mentioned, generally, that we take G2P only as an (important) sample application of monotone many-to-many alignments, but that they clearly apply to other fields of natural language processing as well, such as transliteration, morphology/lemmatization, etc. and we thus also incorporate experiments on morphology data. Moreover, as indicated, we do not question the premise of monotonicity in the current work, but take it as a crucial assumption of our approach, leading to efficient algorithms. Still, 'local non-monotonicities' as exemplified above can certainly be adequately addressed within our framework, as should become clear from our illustrations below (e.g. with higher-order 'steps').

2 S-restricted monotone paths and alignments

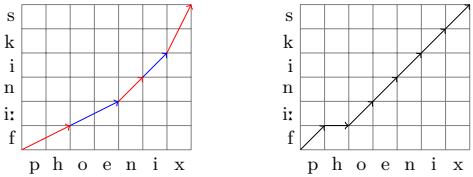


Figure 1: Monotone paths in two-dimensional lattices corresponding to the monotone alignments between $\mathbf{x}$ = phoenix and $\mathbf{y}$ = fi:niks given in Section 1. In the left lattice, we have arbitrarily (but suggestively) colored each step in either red or blue.

Denote by $\mathbb{Z}$ the set of integers, by $\mathbb{N}$ the set of non-negative integers, and by $\mathbb{R}$ the set of real numbers. Consider the two-dimensional lattice $\mathbb{Z}^2$. In $\mathbb{Z}^2$, we call an ordered list of pairs $(\alpha_0, \beta_0) = (0,0), \ldots, (\alpha_k, \beta_k) = (m,n)$ a *path* from $(0,0)$ to (m,n), and we call

[1]The many-to-many alignment algorithm designed in (Jiampojamarn et al., 2007) is an extension of a one-to-one stochastic transducer devised in (Ristad and Yianilos, 1998). Moreover, (Brill and Moore, 2000) learn the weighted edit distance between string pairs where edit operations may encompass arbitrary subsequences of strings, a setting also closely related to our problem of monotone many-to-many alignments.

$(a_i, b_i) := (\alpha_i, \beta_i) - (\alpha_{i-1}, \beta_{i-1})$, $i = 1, \ldots, k$, *steps*. Moreover, we call a path λ in the lattice $\mathbb{Z}^2$ from $(0,0)$ to (m,n) *monotone* if all steps (a,b) are non-negative, i.e. $a \geq 0$, $b \geq 0$, and we call the monotone path λ *S-restricted* for a subset S of $\mathbb{N}^2$ if all steps lie within S, i.e. $(a,b) \in S$.

Note that S-restricted monotone paths define *S-restricted monotone alignments*, between strings $\mathbf{x}$ and $\mathbf{y}$. For example, the two paths in Figure 1 correspond to the two monotone alignments between $\mathbf{x}$ = phoenix and $\mathbf{y}$ = fi:niks illustrated above. Thus, we identify S-restricted monotone paths with S-restricted monotone alignments in the sequel.

Moreover, note that the set and number of S-restricted monotone paths allow simple recursions. To illustrate, the number $T_S(m,n)$ of S-restricted monotone paths from $(0,0)$ to (m,n) satisifies

$$T_S(m,n) = \sum_{(a,b) \in S} T_S(m-a, n-b), \tag{1}$$

with initial condition $T_S(0,0) = 1$ and $T_S(m,n) = 0$ if $m < 0$ or $n < 0$. As will be seen in the next section, under certain assumptions, *optimal* monotone alignments (or, equivalently, paths) can be found via a very similar recursion.

3 An algorithm for S-restricted monotone alignments

Let two strings $\mathbf{x} \in \Sigma_x^*$ and $\mathbf{y} \in \Sigma_y^*$ be given. Moreover, assume that a set S of allowable steps is specified together with a real-valued similarity function $\mathsf{sim} : \Sigma_x^* \times \Sigma_y^* \to \mathbb{R}$ between characters of Σ_x and Σ_y. Finally, assume that the *score* or value of an S-restricted monotone path $\lambda = (\alpha_0, \beta_0), \ldots, (\alpha_k, \beta_k)$ is defined additively linear in the similarity of the substrings of $\mathbf{x}$ and $\mathbf{y}$ corresponding to the steps (a,b) taken, i.e.

$$\text{score}(\lambda) = \sum_{i=1}^{k} \mathsf{sim}(x_{\alpha_{i-1}+1}^{\alpha_i}, y_{\beta_{i-1}+1}^{\beta_i}), \tag{2}$$

where by $x_{\alpha_{i-1}+1}^{\alpha_i}$ we denote the subsequence $x_{\alpha_{i-1}+1} \ldots x_{\alpha_i}$ of $\mathbf{x}$ and analogously for $\mathbf{y}$. Then it is not difficult to see that the problem of finding the path (alignment) with maximal score can be solved efficiently using a very similar (dynamic programming) recursion as in Equation (1), which we outline in Algorithm 1. Moreover, this algorithm is obviously a straightforward generalization of the classical Needleman-Wunsch algorithm, which specifies S as $\{(0,1),(1,0),(1,1)\}$.

Note, too, that in Algorithm 1 we include two additional quantities, not present in the original sequence alignment approach, namely, firstly, the 'quality' q of a step (a,b), weighted by a factor $\gamma \in \mathbb{R}$. This quantity may be of practical importance in many situations. For example, if we specify sim as log-probability (see below), then Algorithm 1 has a 'built-in' tendency to substitute 'smaller', individually more likely steps (a,b) by larger, less likely steps because in the latter case fewer negative numbers are added; if sim assigns strictly positive values, this relationship is reversed. We can counteract these biases by factoring in the *per se* quality of a given step. Also note that if q is added linearly, as we have specified, then the dynamic programming recursion is not violated.

Secondly, we specify a function $\mathsf{L} : \left(\Sigma_x^* \times \Sigma_y^*\right) \times \mathsf{colors} \to \mathbb{R}$, where colors is a finite set of 'colors', that encodes the following idea. Assume that each step $(a,b) \in S$ appears in

C, $C \in \mathbb{N}$, different 'colors', or states. Then, when taking step (a, b) with color $c \in \mathsf{colors}$ (which we denote by the symbol $(a, b)^c$ in Algorithm 1), we assess the 'goodness' of this decision by the 'likelihood' L that the current subsequences of $\mathbf{x}$ and $\mathbf{y}$ selected by the step (a, b) 'belong to'/'are of' color (or state) c. As will be seen below, this allows to very conveniently identify (or postulate) 'latent classes' for character subsequences, while increasing the algorithm's running time only by a constant factor.

To summarize our generalizations over the traditional sequence alignment approach, (i) we allow arbitrary non-negative steps S corresponding to S-restricted monotone alignments, (ii) we include a goodness measure q that evaluates the 'quality' of a given step $(a, b) \in S$ taken, and (iii) we color each step in C different colors and assess the goodness of color c for the subsequences of $\mathbf{x}$ and $\mathbf{y}$ selected by the current step (a, b) as the 'likelihood' L that these subsequences are of color c. Finally, we define the score of a monotone path as an additive linear combination of all three components discussed so that an efficient dynamic programming recursion applies. Note that the algorithm's running time is $O(C|S|mn)$ and is thus linear in the number of colors, the size of S, and the string lengths m and n.[2]

Algorithm 1 Generalized Needleman-Wunsch (GNW)

procedure GNW($x_1 \dots x_m, y_1 \dots y_n$; S, $\mathsf{sim}, \mathsf{q}, \mathsf{L}$)
  $M_{ij} \leftarrow -\infty$ for all $(i, j) \in \mathbb{Z}^2$ such that $i < 0$ or $j < 0$
  $M_{00} \leftarrow 0$
  for $i = 0 \dots m$ **do**
    for $j = 0 \dots n$ **do**
      if $(i, j) \neq (0, 0)$ **then**
        $M_{ij} \leftarrow \max_{(a,b)^c \in S} \{M_{i-a,j-b} + \mathsf{sim}(x^i_{i-a+1}, y^j_{j-b+1}) + \gamma \mathsf{q}(a, b) + \chi \mathsf{L}\big((x^i_{i-a+1}, y^j_{j-b+1}), c\big)\}$
      end if
    end for
  end for
  return M_{mn} ▷ M_{mn} holds value of path with maximal score
end procedure

Algorithm 2 (Hard) EM Training

procedure EM($\{(\mathbf{x}_i, \mathbf{y}_i) \mid i = 1, \dots, N\}$; S, T, $\hat{\mathsf{sim}}_0, \hat{\mathsf{q}}_0, \hat{\mathsf{L}}_0$)
  $t \leftarrow 0$
  while $t < T$ **do**
    for $i = 1 \dots N$ **do**
      $(\mathbf{x}^a_i, \mathbf{y}^a_i) \leftarrow$ GNW($\mathbf{x}_i, \mathbf{y}_i; S, \hat{\mathsf{sim}}_t, \hat{\mathsf{q}}_t, \hat{\mathsf{L}}_t$)
        ▷ $(\mathbf{x}^a_i, \mathbf{y}^a_i)$ denotes the alignment between $\mathbf{x}_i$ and $\mathbf{y}_i$
    end for
    $\hat{\mathsf{sim}}_{t+1}, \hat{\mathsf{q}}_{t+1}, \hat{\mathsf{L}}_{t+1} \leftarrow f(\{\mathbf{x}^a_i, \mathbf{y}^a_i \mid i = 1, \dots, N\})$
      ▷ The function f extracts (count) updates from the aligned data
    $t \leftarrow t + 1$
  end while
end procedure

[2] But also note the dependence of the running time on the definition of sim, q and L.

As to the similarity measure sim employed in Algorithm 1, a popular choice is to specify it as the (logarithm of the) joint probability of the pair $(\mathbf{u}, \mathbf{v}) \in \Sigma_x^* \times \Sigma_y^*$, but a multitude of alternatives is conceivable here such as the χ^2 similarity, pointwise mutual information, etc. (see for instance the overview in (Hoang et al., 2009)). Also note that $\mathsf{sim}(\mathbf{u}, \mathbf{v})$ is usually initially unknown but can be iteratively estimated via application of Algorithm 1 and count estimates in an EM-like fashion (cf. (Dempster et al., 1977)), see Algorithm 2.[3] As concerns q and L, we can likewise estimate them iteratively from data, specifying their abstract forms via any well-defined (goodness) measures. The associated coefficients γ and χ can be optimized on a development set or set exogenously.

4 Exhaustive enumeration and alignments

In the last section, we have specified a polynomial time algorithm for solving the monotonic S-restricted string alignment problem, under the following restriction; namely, we defined the score of an alignment additively linear in the similarities of the involved subsequences. This, however, entails an independence assumption between successive aligned substrings that oftentimes does not seem justified in linguistic applications. If, on the contrary, we specified the score, $\mathrm{score}(\lambda)$, of an alignment λ between strings $\mathbf{x}$ and $\mathbf{y}$ as e.g.

$$\mathrm{score}(\lambda) = \sum_{i=1}^{k} \log \Pr\big((x_{\alpha_{i-1}+1}^{\alpha_i}, y_{\beta_{i-1}+1}^{\beta_i}) \,|\, (x_{\alpha_{i-2}+1}^{\alpha_{i-1}}, y_{\beta_{i-2}+1}^{\beta_{i-1}})\big)$$

(using joint probability as similarity measure) — this would correspond to a 'bigram scoring model' — then Algorithm 1 would not apply.

To address this issue, we suggest *exhaustive enumeration* as a possibly noteworthy alternative — enumerate *all* S-restricted monotone alignments between strings $\mathbf{x}$ and $\mathbf{y}$, score each of them individually, taking the one with maximal score. This brute-force approach is, despite its simplicity, the most general approach conceivable and works under all specifications of scoring functions. Its practical applicability relies on the sizes of the search spaces for S-restricted monotone alignments and on the lengths of the strings $\mathbf{x}$ and $\mathbf{y}$ involved.

We note the following here. By Equation (1), for the choice $S = \{(1,1),(1,2),(1,3),(1,4),(2,1)\}$, a seemingly reasonable specification in the context of G2P (see next section), the number $T_S(n,n)$ of S-restricted monotone alignments is given as (for explicit formulae for specific S, cf. (Eger, 2012b))

$$1, 1, 3, 7, 16, 39, 95, 233, 572, 1406, 3479, 8647$$

for $n = 1, 2, \ldots, 12$ and e.g. $T_S(15,15) = 134,913$. Moreover, for the distribution of letter string and phoneme string lengths we estimate Poisson distributions (cf. (Wimmer et al., 1994)) with parameters $\mu \in \mathbb{R}$ as listed in Table 4 for the German Celex (Baayen et al., 1996), French Brulex (Content et al., 1990) and English Celex datasets, as used in Section 7. As the table and the above numbers show, there are on average only a few hundred or few thousand possible monotone many-to-many alignments between grapheme and phoneme string pairs, for which exhaustive enumeration appears, thus, quite feasible; moreover, given enough data, it usually does not harm much to exclude a few string pairs, for which alignment numbers are too large.

[3] The variant of EM that we describe is sometimes called *hard* EM while e.g. (Jiampojamarn et al., 2007) present a soft EM version; but see the discussion in (Samdani et al., 2012).

Dataset	μ_G	μ_P	$P_{[G>15]}$	$P_{[P>15]}$
German-Celex	9.98	8.67	4.80%	1.62%
French-Brulex	8.49	6.71	1.36%	0.15%
English-Celex	8.21	7.39	1.03%	0.40%

Table 1: Avg. grapheme and phoneme string lengths in resp. data set, and probabilities that lengths exceed 15.

5 Choice of S

Choice of the set of steps S is a question of *model selection*, cf. (Zucchini, 2000). Several approaches are conceivable here. First, for a given domain of application one might specify a possibly 'large' set of steps Ω capturing a preferably comprehensive class of alignment phenomena in the domain. This may not be the best option because it may provide Algorithm 1 with too many 'degrees of freedom', allowing it to settle in unfavorable local optima, and thus may lead to suboptimal alignments (we find appropriate step restriction to have dramatic effects on alignment quality, which we investigate more thoroughly in subsequent research). A better, but potentially very costly, alternative is to exhaustively enumerate all possible subsets S of Ω, apply Algorithm 1 and/or Algorithm 2, and evaluate the quality of the resulting alignments with any choice of suitable measures such as alignment entropy (cf. (Pervouchine et al., 2009)), average log-likelihood, Akaike's information criterion (Akaike, 1974) or the like. Another possibility would be to use a comprehensive Ω, but to penalize unlikely steps, which could be achieved by setting γ in Algorithm 1 to a 'large' real number and then, in subsequent runs, employ the remaining steps $S \subseteq \Omega$; we outline this approach in Section 7.

Sometimes, specific knowledge about a particular domain of application may be helpful, too. For example, in the field of G2P, we would expect most associations in alignments to be of the type M-to-1, i.e. one or several graphemes encode a single phoneme. This is because it seems reasonable to assume that the number of phonetic units used in language communities typically exceeds the number of units in alphabetic writing systems — 26 in the case of the Latin alphabet — so that one or several letters must be employed to represent a single phoneme. There may be 1-to-N or even M-to-N relationships but we would consider these exceptions. In the current work, we choose $S = \{(1,1),(2,1),(3,1),(4,1),(1,2)\}$ for G2P data sets, and for the morphology data sets we either adopt from (Eger, 2012b) or use a comprehensive Ω with 'largest' step $(2,2)$.

6 Decoding

Decoding is the process of generating $\hat{\mathbf{y}} \in \Sigma_y^*$ given $\mathbf{x} \in \Sigma_x^*$. Below, we explain how we perform this process, within the S-restricted monotone many-to-many alignment framework.

6.1 Training a string transduction model

We first generate monotone many-to-many alignments between string pairs with one of the procedures outlined in Sections 3 and 4. Then, we train a linear chain conditional random field (CRF; see (Lafferty et al., 2001)) as a graphical model for string transduction on the aligned data. The choice of CRFs is arbitrary; any transduction procedure tr would do, but we decide for CRFs because they generally have good generalization properties. In all cases, we use window sizes of three or four to predict $\mathbf{y}$ string elements from $\mathbf{x}$ string elements.

6.2 Segmentation

Our overall decoding procedure is as follows. Given an input string $\mathbf{x}$, we exhaustively generate all possible segmentations of $\mathbf{x}$, feed the segmented strings to the CRF for transduction and evaluate each individual resulting sequence of 'graphones' with an n-gram model learned on the aligned data, taking the $\mathbf{y}$ string corresponding to the graphone sequence with maximal probability as the most likely transduced string for $\mathbf{x}$. We illustrate in Algorithm 3. As to the size of the search space that this procedure entails, any segmentation

Algorithm 3 Decoding

procedure decode($\mathbf{x} = x_1 \dots x_m$; $\hat{k}$, α, β, tr)
  $Z \leftarrow \emptyset$
  for $s \in \mathscr{C}(m, \hat{k}, \alpha, \beta)$ **do** ▷ $\mathscr{C}(m, \hat{k}, \alpha, \beta)$: the set of all integer compositions of m with $\hat{k}$ parts, each between α and β
    $\hat{\mathbf{y}} \leftarrow \mathsf{tr}(s)$
    $z_{\hat{\mathbf{y}}} \leftarrow \text{ngramScore}(\mathbf{x}, \hat{\mathbf{y}})$
    $Z \leftarrow Z \cup \{z_{\hat{\mathbf{y}}}\}$
  end for
  $z_{\hat{\mathbf{y}}^*} \leftarrow \max_{z_{\hat{\mathbf{y}}}} Z$
  return $\hat{\mathbf{y}}^*$
end procedure

of a string $\mathbf{x}$ of length m with k parts uniquely corresponds to an *integer composition* (a way of writing m as a sum of non-negative integers) of the integer m with k parts, as in,

		ph		oe		n		i		x
7	=	2	+	2	+	1	+	1	+	1

It is a simple exercise to show that there are $\binom{m-1}{k-1}$ integer compositions of m with k parts, where by $\binom{m}{k}$ we denote the respective binomial coefficient. Furthermore, if we put restrictions on the maximal size of parts — e.g. in G2P a reasonable upper bound l on the size of parts would probably be 4 — we have that there are $\binom{k}{m-k}_l$ integer compositions of m with k parts, each between $\alpha = 1$ and $\beta = l$, where by $\binom{k}{m}_{l+1}$ we denote the respective *polynomial coefficient* (Comtet, 1974). To avoid having to enumerate segmentations for all possible numbers k of segment parts of a given input string $\mathbf{x}$ of length m — these would range between 1 and m, entailing $\sum_{k=1}^{m} \binom{m-1}{k-1} = 2^{m-1}$ possible segmentations in total in the case without upper bound[4] — we additionally train a 'number of parts' prediction model with which to estimate k as $\hat{k}$; we call this in short *predictor model*.

To illustrate the number of possible segmentations with a concrete example, if $\mathbf{x}$ has length $m = 15$, a rather large string size given the values in Table 4, there are

$$2472, 2598, 1902, 990, 364, 91, 14, 1$$

possible segmentations of $\mathbf{x}$ with $k = 8, 9, 10, 11, 12, 13, 14, 15$ parts, each between 1 and 4.

For the sake of completeness, we note that our above discussion presumed that there are no 'empty' parts in integer compositions, that is, that all parts in the integer composition

[4] In the case of upper bounds, (Malandro, 2012) provides asymptotics for the number of restricted integer compositions, which are beyond the scope of the present work, however.

2PKE. abbreche**t**, entgegentrete**t**, zuzieh**t**
z. ab**zu**brechen, entgegen**zu**treten, zu**zu**ziehen
rP. redet, **reibt**, tr**eibt**, verbind**et**
pA. **ge**redet, **ge**r**ieben**, **ge**tr**ieben**, verb**u**nd**en**

Table 2: String pairs in morphology data sets 2PKE and rP (omitting 2PIE and 13SIA for space reasons) discussed by (Dreyer et al., 2008). Changes from one form to the other are in bold (information not given in training). Adapted from (Dreyer et al., 2008).

are integers between 1 and the upper bound l. When converting graphemes to phonemes, we find it unlikely that a sound would be uttered without there being a corresponding letter that gives rise to this sound,[5] i.e. our assumption seems justified. In the general monotone alignment case, however, the zero case would have to be included, e.g. when converting phonemes to graphemes, or in the morphology data sets discussed below, where e.g. segmentations as in

$$\begin{array}{ccccccccccccc} & & \emptyset & & \mathrm{m} & & \mathrm{a} & & \mathrm{c} & & \mathrm{h} & & \mathrm{t} \\ 5 & = & 0 & + & 1 & + & 1 & + & 1 & + & 1 & + & 1 \end{array}$$

seem justified to convert German third person verb form *macht* into participle form *gemacht*. Analogously as above, we find that there are $\binom{k}{m}_{l+1}$ integer compositions of m with k parts, each between 0 and l. To illustrate again, when $m = 15$, there are $37080, 142749, 831204, 2268332, \ldots$ possible segmentations of $\mathbf{x}$ with $k = 8, 9, 10, 11, \ldots$ parts, each between 0 and 4. Obviously, these numbers are much larger than those where all parts are ≥ 1, which is problematic not only from the point of view of computing resources but may also affect accuracy results because more alternatives are provided from which to select. Luckily, as illustrated below, it should usually be possible to specify modeling choices where zero parts do not occur.

7 Experiments

We conduct our experiments on three G2P data sets, the German Celex (G-Celex) and French Brulex data set (F-Brulex) taken from the Pascal challenge (van den Bosch et al., 2006), and the English Celex dataset (E-Celex); and on the four German morphology data sets discussed in (Dreyer et al., 2008), which we refer to, in accordance with the named authors, as rP, 2PKE, 13SIA and 2PIE, respectively. Both for the G2P and the morphology data, we hold monotonicity, by and large, a legitimate assumption so that our approach would appear justified. As to the morphology data sets, we illustrate in Table 7 a few string pair relationships that they contain, as indicated by (Dreyer et al., 2008).

7.1 Alignments

We generate alignments for our data sets using Algorithms 1 and 2 and, as a comparison, we implement an exhaustive search bigram scoring model as indicated in Section 4 in an EM-like fashion similar as in Algorithm 2, employing the CMU SLM toolkit (Clarkson and Rosenfeld, 1997) with Witten-Bell smoothing as n-gram model. For Algorithm 1, which we also refer to as unigram model in the following, we choose steps S as shown in Table

[5] As an exception might be considered e.g. extra terminal vowel sounds like in Italian *sport*, pronounced as s p o r t ə. As pointed out by a reviewer, other such exceptions might include short vowels in Arabic or Hebrew script that are generally not graphemically represented.

E-Celex	$\{(1,1),(2,1),(3,1),(4,1),(1,2)\}$
rP	$\{(0,2),(1,1),(1,2),(2,1),(2,2)\}$
2PKE	$\{(0,2),(1,1),(2,1),(2,2)\}$
13SIA	$\{(1,1),(1,2),(2,1),(2,2)\}$
2PIE	$\{(1,1),(1,2)\}$

Table 3: Data set and choice of S. For all three G2P data sets, we select the same S, exemplarily shown for E-Celex. The choice of S for rP and 2PKE is taken from (Eger, 2012b). For 13SIA and 2PIE we use comprehensive Ω's with largest step $(2,2)$ but the algorithm ends up using just the outlined set of steps.

	Perplexity	$H(L\,\vert\,P)$
2PKE-Uni	7.002 ± 0.04	$\mathbf{0.094 \pm 0.001}$
2PKE-Bi	$\mathbf{6.865 \pm 0.02}$	0.141 ± 0.003
rP-Uni	9.848 ± 0.09	$\mathbf{0.092 \pm 0.003}$
rP-Bi	$\mathbf{9.796 \pm 0.05}$	0.107 ± 0.006
Brulex-Uni	22.488 ± 0.35	$\mathbf{0.706 \pm 0.002}$
Brulex-Bi	$\mathbf{22.215 \pm 0.21}$	0.725 ± 0.003

Table 4: Conditional entropy vs. n-gram perplexity ($n = 2$) of alignments for different data sets. In bold: Statistically best results. $K = 300$ throughout.

3. As similarity measure sim, we use log prob with Good-Turing smoothing and for q we likewise use log prob; we outline the choice of L below. Initially, we set γ and χ to zero. As an alignment quality measure we consider conditional entropy $H(L\,|\,P)$ (or $H(P\,|\,L)$) as suggested by (Pervouchine et al., 2009). Conditional entropy measures the average uncertainty of a (grapheme) substring L given a (phoneme) substring P; apparently, the smaller $H(L\,|\,P)$ the better the alignment because it produces more consistent associations.

In the following, all results are averages over several runs, 5 in the case of the unigram model and 2 in the case of the bigram model. Both for the bigram model and the unigram model, we select K, where $K \in \{50, 100, 300, 500\}$, training samples randomly in each EM iteration for alignment and from which to update probability estimates.

In Figure 2, we show learning curves over EM iterations in the case of the unigram and bigram models, and over training set sizes. We see that performance, as measured by conditional entropy, increases over iterations both for the bigram model and the unigram model (in Figure 2), but apparently alignment quality decreases again when too large training set sizes K are considered in the case of the bigram model (omitted for space reasons); similar outcomes have been observed when similarity measures other than log prob are employed in Algorithm 1 for the unigram model, e.g. the χ^2 similarity measure (cf. (Eger, 2012b)). To explain this, we hypothesize that the bigram model (and likewise for specific similarity measures) is more susceptible to overfitting when it is trained on too large training sets so that it is more reluctant to escape 'non-optimal' local minima. We also see that, apparently, the unigram model performs frequently better than the bigram model.

The latter results may be partly misleading, however. Conditional entropy, the way (Pervouchine et al., 2009) have specified it, is a 'unigram' assessment model itself and may therefore be incapable of accounting for certain contextual phenomena. For example, in the 2PKE and rP data, we find alignment possibilities of the following types,

–	g	e	b	t		g	e	–	b	t
ge	g	e	b	en		g	e	ge	b	en

where we list the linguistically 'correct', due to the prefixal character of *ge* in German, alignment on the left and the 'incorrect' alignment on the right. By its specification, Algorithm 1 *must* assign both these alignments the same score and can hence not distinguish between them; *the same holds true for the conditional entropy measure*. To address this issue, we evaluate alignments by a second method as follows. From the aligned data, we extract a random sample of size 1000 and train an n-gram graphone model (that can account for 'positional associations') on the residual, assessing its perplexity on the held-out set of size 1000. Results are shown in Table 4. We see that, in agreement with our visual impression at least for the morphology data, the alignments produced by the bigram model seem to be slightly more consistent in that they reduce perplexity of the n-gram graphone model, whereas conditional entropy proclaims the opposite ranking.

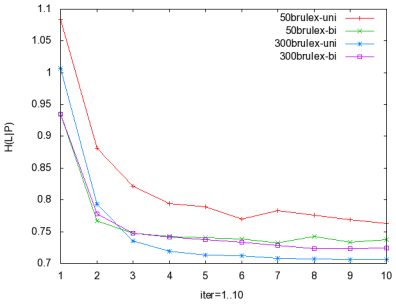


Figure 2: Learning curves over iterations for F-Brulex data, $K = 50$ and $K = 300$, for unigram and bigram models.

7.1.1 Quality q of steps

In Table 5 we report results when experimenting with the coefficient γ of the quality of steps measure q. Overall, we do not find that increasing γ would generally lead to a performance increase, as measured by e.g. $H(L\,|\,P)$. On the contrary, when choosing as set of steps a comprehensive Ω as in Table 5, where we choose $\Omega = \{(a,b)\,|\,a \leq 4, b \leq 4\}\backslash\{(0,0)\}$, for $\gamma = 0$, we find values of 0.278, 0.546, 0.662 for $H(L\,|\,P)$ for G-Celex, F-Brulex and E-Celex, respectively, while corresponding values for $\gamma = 10$ are 0.351, 0.833, 1.401. Contrarily, $H(P\,|\,L)$, the putatively more indicative measure for transduction from $\mathbf{x}$ to $\mathbf{y}$, has 0.499, 0.417, 0.598 for $\gamma = 0$ and 0.378, 0.401, 1.113 for $\gamma = 10$, so that, except for the E-Celex data, $\gamma = 10$ apparently leads to improved $H(P\,|\,L)$ values in this situation, while $\gamma = 0$ seems to lead to better $H(L\,|\,P)$ values.

In any case, from a model complexity perspective,[6] increasing γ may certainly be beneficial. For example, Table 5 shows that with $\gamma = 0$, Algorithm 1 will select up to 15 different steps for the given choice Ω, most of which seem linguistically questionable. On the contrary, with a large γ, Algorithm 1 employs only four resp. five different steps for the G2P data; most

[6]Taking into model complexity is e.g. in accordance with Occam's razor or Akaike's information criterion.

importantly, among these are $(1,1)$, $(2,1)$ and $(3,1)$, all of which are in accordance with linguistic reasoning as e.g. outlined in Section 5. Thus, we can think of q as a 'regularization term' that prevents the algorithm from 'overfitting' the data.

	$(1,1)$	$(2,1)$	$(3,1)$	$(4,1)$	$(1,2)$	$(1,0)$	$(2,3)$	$(3,2)$	$(3,3)$	$(4,2)$	$(4,3)$	$(4,4)$	$(2,2)$	$(0,1)$	$(1,3)$
G-Celex	86.50	11.61	1.77	-	0.10	-	-	-	-	-	-	-	-	-	-
	86.14	8.17	1.63	0.02	0.00	2.56	0.10	0.04	0.01	0.09	0.91	0.28	-	-	-
F-Brulex	78.85	15.08	5.85	-	-	-	0.20	-	-	-	-	-	-	-	-
	75.64	13.80	2.52	0.36	0.07	5.07	0.29	0.10	0.02	0.38	1.01	0.68	-	-	-
E-Celex	88.87	6.58	3.05	-	-	-	-	-	-	-	-	-	-	1.29	0.18
	75.54	8.45	0.75	0.04	1.48	4.57	0.41	0.03	0.16	0.44	2.03	3.03	0.00	2.87	0.12

Table 5: Steps and their frequency masses in percent for different data sets for $\gamma = 10$ (top rows) and $\gamma = 0$ (bottom rows), averaged over two runs. We include only steps whose average occurrence exceeds 10.

7.1.2 Colors

We briefly discuss here a possibility to detect latent classes via the concept of colored paths. Assume that a corpus of colored alignments is available and let each color be represented by the contexts (graphones to the left and right) of its members; moreover, define the 'likelihood' L that the pair $p_{x,y} := (x_{\alpha_{i-1}+1}^{\alpha_i}, y_{\beta_{i-1}+1}^{\beta_i})$ is of color c as the (document) similarity (in an information retrieval sense) of $p_{x,y}$'s contexts with color c, which we can e.g. implement via the cosine similarity of the context vectors associated with $p_{x,y}$ and c. For number of colors $C = 2$, we then find, under this specification, the following kinds of alignments when running Algorithms 1 and 2 with $\gamma = 0$ and $\chi = 1$,

a	nn	**u**	al		ph	**o**	n	**e**	me
&	n	**jU**	l		f	**@U**	n	**i**	m

where we use bold font to distinguish the two color classes, and use original E-Celex notation for phonemic characters. It is clear that the algorithm has detected some kind of consonant/vowel distinction on a phonemic level here. We find similar kinds of latent classes for the other G2P data sets, and for the morphology data, the algorithm learns (less interestingly) to detect word endings and starts, under this specification.

7.2 Transductions

We report results of experiments on transducing $\mathbf{x}$ strings to $\mathbf{y}$ strings for the G2P data and the morphology data sets. We exclude E-Celex because training the CRF with our parametrizations (e.g. all features in window size of four) did regularly not terminate, due to the large size of the data set ($> 60{,}000$ string pairs). Likewise for computing resources reasons,[7] we do not use ten-fold cross-validation but, as in (Jiampojamarn et al., 2008), train on the first 9 folds given by the Pascal challenge, testing on the last. Moreover, for the G2P data, we use an ϵ-scattering model with steps $S = \{(1,0),(1,1)\}$ as a predictor model from which to infer the number of parts $\hat{k}$ for decoding and then apply Algorithm 3.[8] For alignments, we use in all cases Algorithms 1 and 2 with $\gamma = 0$ and $\chi = 0$. As reference for the G2P data, we give word accuracy rates as announced by (Bisani and Ney,

[7] E.g. a single run of the CRF on the G-Celex data takes longer than 24 hours on a standard PC.

[8] We train the ϵ-scattering model on data where all multi-character phonemes such as *ks* are merged to a single character, as obtained from the alignments as given by Algorithms 1 and 2.

	CRF-3	CRF-4	CRF-4*	DSE-F	DSE-FL	Mos3	Mos15	M-M+HMM	BN	MeR+A*
F-Brulex		93.7	**94.6**					90.9	93.7	86.7
G-Celex		91.1	**92.6**					89.8		90.2
2PKE	79.8	80.9		74.7	**87.4**	67.1	82.8			
rP	74.1	77.2		69.9	**84.9**	67.6	70.8			
13SIA	85.6	86.5		82.8	**87.5**	73.9	85.3			
2PIE	**94.6**	94.2		88.7	93.4	92.0	94.0			

Table 6: Data sets and word accuracy rates in percent. **DSE-F**: (Dreyer et al., 2008) using 'pure' alignments and features. **DSE-FL**: (Dreyer et al., 2008) using alignments, features and latent classes. **Mos3, Mos15**: Moses system with window sizes of 3 and 15, resp., as reported by (Dreyer et al., 2008). **M-M+HMM**: Many-to-many aligner with HMM and instance-based segmenter for decoding as reported by (Jiampojamarn et al., 2007). **BN**: (Bisani and Ney, 2008) using a machine translation motivated approach to many-to-many alignments. **MeR+A***: Results of Moses system on G2P data as reported by (Rama et al., 2009). **CRF-3** Our approach with window size of 3 and 3-gram scoring model (see Algorithm 3). **CRF-4**: Our approach with window size of 4 and 3-gram scoring model. **CRF-4***: Our approach with window size of 4 and 4-gram scoring model and 2-best lists (i.e. in Algorithm 3, obtain $\hat{\mathbf{y}}_1$ and $\hat{\mathbf{y}}_2$ as the two most probable transductions of s). In bold: Best results (no statistical tests). Underlined: best results using 'pure' alignments.

2008), (Jiampojamarn et al., 2007), and (Rama et al., 2009), who gives the Moses 'baseline' (Koehn et al., 2007).

For the morphology data we use exactly the same training/test data splits as in (Dreyer et al., 2008). Moreover, because (Dreyer et al., 2008) report all results in terms of window sizes of 3, we do likewise for this data. For decoding we do not use a (complex) predictor model here but rely on simple statistics; e.g. we find that for the class 13SIA, k is always in $\{m-2, m-1, m\}$, where m is the length of $\mathbf{x}$, so we apply Algorithm 3 three times and select the best scoring $\hat{\mathbf{y}}$ string. To avoid zeros in the decoding process (see discussion in Section 6.2), we replace the $(0, 2)$ steps used in the rP and 2PKE data sets by a step $(1, 3)$.

Results are shown in Table 6. For the G2P data, our approach always outperforms the best reported results for pipeline approaches (see below), while we are significantly below the results reported by (Dreyer et al., 2008) for the morphology data in two out of four cases. Contrarily, when 'pure' alignments are taken into consideration — (Dreyer et al., 2008) learn very complex latent classes with which to enrich alignments — our results are clearly better throughout. In almost all cases, we significantly beat the Moses 'baseline'.

8 Discussion

We believe our alignment procedure to be superior to the one presented in (Jiampojamarn et al., 2007) (and likewise for the 'machine translation motivated' approach outlined by (Bisani and Ney, 2008)) from a number of perspectives. First, it is more flexible and general in that it allows the specification of arbitrary non-negative steps S and arbitrary similarity measures sim. Moreover, as we have shown, our approach can very easily and conveniently be adapted to incorporate step quality measures, which may turn out to be very useful in detecting the 'right' choice of S (i.e. as a 'regularization term'); and our approach can also easily be generalized to incorporate the modeling of latent classes as e.g. done in (Dreyer et al., 2008), within a polynomial running time framework; further generalizations such as semi-ring specifications (Mohri, 2002) are obvious but not discussed in the current

work (cf. (Eger, 2012b)). Secondly, our algorithm appears very simple and intuitive, while being computationally equivalently tractable and making the same sorts of independence assumptions as in (Jiampojamarn et al., 2007).

As regards decoding, (Jiampojamarn et al., 2007) use a grapheme segmentation module where each grapheme letter can form a chunk with its neighbor or stand alone, a decision that is based on local context and instance-based learning. We hold this approach to be insufficient because (besides the obvious drawback that, as we have shown, larger chunks than two seem appropriate for G2P) it unecessarily restricts the search space for grapheme segmentation; once a decision is made to join two letters, it cannot be reversed and alternative segmentations are not considered. The same holds true for the phrasal decoder approach outlined in (Jiampojamarn et al., 2008), the critique of which is already uttered in (Dreyer et al., 2008), namely, that the input string is segmented into substrings which are transduced *independently* of each other, ignoring context. Contrarily, for decoding $\mathbf{x}$, we compute *all* possible segmentations of $\mathbf{x}$ and score them (in conjunction with the transduced $\hat{\mathbf{y}}$ strings) with higher order n-gram models, which is clearly superior to the named approaches because it takes both context into account and does not restrict search space. Moreover, *given an adequate predictor model*, we found that enumerating all possible restricted integer compositions is so fast that no further investigation of restricting search space is necessary.[9]

While we thus believe our individual components for alignment and decoding to be superior to the mentioned approaches, our modeling of string transductions adheres to a pipeline approach — in (Jiampojamarn et al., 2008)'s words — that, as they suggest, is inferior to a unified framework, as they present it. All our components can be integrated within such a framework, which is scope for future research. In contrast with (Dreyer et al., 2008), we believe our alignments (*per se*) to be more adequate (they use one-to-one and one-to-zero alignments), which the performance measures corroborate, while their idea to enrich alignments with a multitude of latent classes (more complex than representable in our framework) obviously outperforms our method on certain data sets such as those encountered in morphology, where e.g. latent *word* classes may be of great importance.

Conclusion

We have presented a simple and general framework for generating monotone many-to-many alignments that competes with (Jiampojamarn et al., 2007)'s alignment procedure. Moreover, we have discussed crucial independence assumptions and, thus, limitations of this algorithm and shown that exhaustive enumeration (among other methods) can overcome these problems — in particular, due to the relatively small search space — in the field of monotone alignments. Additionally, we have discussed problems of standard alignment quality measures such as conditional entropy and have suggested an alternative decoding procedure for string transduction within the monotone many-to-many alignment framework that addresses the limitations of the procedures suggested by (Jiampojamarn et al., 2007) and (Jiampojamarn et al., 2008). In future work, we intend to explore more extensively, in particular, the effects of appropriate step restriction and regularization upon alignment quality.

[9] We used the algorithm presented in (Updyke, 2010).

References

Akaike, H. (1974). A new look at the statistical model identification. *IEEE Transactions on Automatic Control*, 19(6):716–723.

Baayen, H., Piepenbrock, R., and Gulikers, L. (1996). *The CELEX2 lexical database*. Linguistic Data Consortium, Philadelphia.

Baldwin, T. and Tanaka, H. (1999). Automated Japanese grapheme-phoneme alignment. In *Proc. of the International Conference on Cognitive Science*, pages 349–354.

Bisani, M. and Ney, H. (2008). Joint-sequence models for grapheme-to-phoneme conversion. *Speech Communication*, 50(5):434–451.

Black, A., Lenzo, K., and Pagel, V. (1998). Issues in building general letter to sound rules. In *The Third ESCA/COCOSDA Workshop (ETRW) on Speech Synthesis. ISCA*.

Brill, E. and Moore, R. (2000). An improved error model for noisy channel spelling correction. In *Proceedings of the 38th Annual Meeting on Association for Computational Linguistics*, pages 286–293. Association for Computational Linguistics.

Brown, P., Cocke, J., , Pietra, S. D., Pietra, V. D., Jelinek, F., Lafferty, J., Mercer, R., and Roossin, P. (1990). A statistical approach to machine translation. *Computational Linguistics*, 16(2):79–85.

Clarkson, P. and Rosenfeld, R. (1997). Statistical language modeling using the CMU-Cambridge toolkit. In *Proceedings ESCA Eurospeech*.

Comtet, L. (1974). *Advanced Combinatorics*. D. Reidel Publishing Company.

Content, A., Mousty, P., and Radeau, M. (1990). Une base de données lexicales informatisée pour le francais écrit et parlé. In *L'Année Psychologique*, pages 551–566.

Dempster, A., Laird, N., and Rubin, D. (1977). Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)*, 39(1):1–38.

Dreyer, M., Smith, J., and Eisner, J. (2008). Latent-variable modeling of string transductions with finite-state methods. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1080–1089, Honolulu, Hawaii.

Eger, S. (2012a). On S-restricted f-weighted integer compositions and extended binomial coefficients. Submitted.

Eger, S. (2012b). Sequence alignment with arbitrary steps and further generalizations, with applications to alignments in linguistics. Submitted.

Galescu, L. and Allen, J. (2001). Bi-directional conversion between graphemes and phonemes using a joint n-gram model. In *Proc. 4th ISCA Tutorial and Research Workshop on Speech Synthesis*, Perthshire, Scotland.

Heubach, S. and Mansour, T. (2004). Compositions of n with parts in a set. *Congressus Numerantium*, 164:127–143.

Hoang, H., Kim, S., and Kan, M.-Y. (2009). A re-examination of lexical association measures. In *MWE '09 Proceedings of the Workshop on Multiword Expressions: Identification, Interpretation, Disambiguation and Applications*.

Jiampojamarn, S., Cherry, C., and Kondrak, G. (2008). Joint processing and discriminative training for letter-to-phoneme conversion. In *46th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL-08: HLT)*, pages 905–913.

Jiampojamarn, S. and Kondrak, G. (2010). Letter-phoneme alignment: An exploration. In *The 48th Annual Meeting of the Association for Computational Linguistics (ACL 2010)*, pages 780–788.

Jiampojamarn, S., Kondrak, G., and Sherif, T. (2007). Applying many-to-many alignments and Hidden Markov models to letter-to-phoneme conversion. In *Proceedings of the Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL-HLT 2007)*, pages 372–379, Rochester, NY.

Koehn, P., Hoang, H., Birch, A., Callison-Burch, C., Federico, M., Bertoldi, N., Cowan, B., Shen, W., Moran, C., Zens, R., Dyer, C., Bojar, O., Constantin, A., and Herbst, E. (2007). Moses: Open source toolkit for statistical machine translation. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, Prague, Czech Republic. Association for Computational Linguistics.

Lafferty, J., McCallum, A., and Pereira, F. (2001). Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proc. 18th International Conf. on Machine Learning*, pages 282–289.

Malandro, M. (2012). Asymptotics for restricted integer compositions. Preprint available at `http://arxiv.org/pdf/1108.0337v1`.

Mohri, M. (2002). Semiring frameworks and algorithms for shortest-distance problems. *Journal of Automata, Languages and Combinatorics*, 7(3):321–350.

Needleman, S. and Wunsch, C. (1970). A general method applicable to the search for similarities in the amino acid sequence of two proteins. *Journal of Molecular Biology*, 48:443–453.

Pervouchine, V., Li, H., and Lin, B. (2009). Transliteration alignment. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 136–144.

Rama, T., Kumar, A., and Kolachina, S. (2009). Modeling letter to phoneme conversion as a phrase based statistical machine translation problem with minimum error rate training. In *NAACL HLT 2009 Student Research Workshop*, pages 90–95, Colorada, USA.

Ristad, E. and Yianilos, P. (1998). Learning string-edit distance. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(5):522–533.

Samdani, R., Chang, M.-W., and Roth, D. (2012). Unified expectation maximization. In *HLT-NAACL*, pages 688–698.

Taylor, P. (2005). Hidden Markov models for grapheme to phoneme conversion. In *Proceedings of the 9th European Conference on Speech Communication and Technology 2005*.

Updyke, J. (2010). A unified approach to algorithms generating unrestricted and restricted integer compositions and integer partitions. *Journal of Mathematical Modelling and Algorithms*, 9(1):53–97.

van den Bosch, A., Chen, S., Daelemans, W., Damper, R., Gustafson, K., Marchand, Y., and Yvon, F. (2006). Pascal letter-to-phoneme conversion challenge. `http://www.pascalnetwork.org/Challenges/PRONALSYL`.

Wimmer, G., Köhler, R., Grotjahn, R., and Altmann, G. (1994). Towards a theory of word length distribution. *Journal of Quantitative Linguistics*, 1:98–106.

Zucchini, W. (2000). An introduction to model selection. *Journal of Mathematical Psychology*, 44:41–61.

Mining words in the minds of second language learners: learner-specific word difficulty

Yo Ehara[1,2] *Issei Sato*[3] *Hidekazu Oiwa*[1,2] *Hiroshi Nakagawa*[3]
(1) Graduate School of Information Science and Technology, the University of Tokyo, Tokyo, Japan
(2) JSPS Research Fellow, Tokyo, Japan
(3) Information Technology Center, the University of Tokyo, Tokyo, Japan
{ehara,sato,oiwa}@r.dl.itc.u-tokyo.ac.jp, nakagawa@dl.itc.u-tokyo.ac.jp

ABSTRACT
While there have been many studies on measuring the size of learners' vocabulary or the vocabulary they should learn, there have been few studies on what kind of words learners actually know. Therefore, we investigated theoretically and practically important models for predicting second language learners' vocabulary and propose another model for this vocabulary prediction task. With the current models, the same word difficulty measure is shared by all learners. This is unrealistic because some learners have special interests. A learner interested in music may know special music-related terms regardless of their difficulty. To solve this problem, our model can define a learner-specific word difficulty measure. Our model is also an extension of these current models in the sense that these models are special cases of our model. In a qualitative evaluation, we defined a measure for how learner-specific a word is. Interestingly, the word with the highest learner-specificity was "twitter". Although "twitter" is a difficult English word, some low-ability learners presumably knew this word through the famous micro-blogging service. Our qualitative evaluation successfully extracted such interesting and suggestive examples. Our model achieved an accuracy competitive with the current models.

KEYWORDS: Learner-specificity, vocabulary prediction, Rasch model.

Proceedings of COLING 2012: Technical Papers, pages 799–814,
COLING 2012, Mumbai, December 2012.

1 Introduction

When learning second languages, vocabulary knowledge is as important as, or sometimes more important, than grammar. The importance of vocabulary knowledge has been a main focus in the last decade in the field of second language acquisition (SLA).

Studies regarding vocabulary knowledge of second language learners have been mainly focusing on two major tasks: devising methods for measuring the size of the second language vocabulary of learners for testing purposes (Schmitt et al., 2001; Laufer and Nation, 1999; Nation, 1990) and determining the words that the learners *should* learn (Nation, 2006). However, there have been few studies on what kind of words learners *actually* know. This is the basic research question for our research.

To study what words second language learners actually know, we focused on the *vocabulary prediction* task. In this task, we aim to build a model that predicts, given a word and a learner, whether or not the learner knows the word. As far as we know, Ehara et al. (2010) is the only study that dealt directly with the vocabulary prediction task. They applied this task to a reading support user interface for second language learners that automatically identifies the words unfamiliar to the learner on a Web page.

The vocabulary prediction task is important for both theory and application. From the theoretical point of view, this task is interesting in that it mines the words second language learners know and creates a model on what kinds of words learners actually know. From the model, we can interpret the patterns or tendency of the learners' process of memorizing second language words. Studying the vocabulary prediction task may also lead to determining if learners actually learn words that SLA experts recommend.

From the application point of view, this task can be used in user-adaptation for reading and writing applications to support second language learners. Ehara et al. (2010)'s model is of this type. They successfully showed the effectiveness of their system. With the increase in Web-based language learning environments, possible data sources for learners' vocabulary knowledge are also increasing. Studying the vocabulary prediction task can shed light on these data sources, and they can be used to further understand the vocabulary knowledge of second language learners.

By using machine learning terminology, the vocabulary prediction task can be categorized as a binary classification task: given a word and a learner, it predicts whether or not the learner knows the word. Therefore, a number of machine learning methods, such as a support vector machine (SVM) for the binary classification task, can be used as predictors. However, to answer our research question, what kind of words learners *actually* know, we want predictors to be able to do more than just predict. Rather, we want predictors that are practical and useful for analysis. Specifically, we list the following properties we want predictors to have.

interpretable weight vector Most predictors use weight vectors trained with data. Weight vectors of some models can be interpreted as quantitative measures of word difficulty and learner ability. Interpretable weight vectors are essential for analysis to find the patterns or tendency of learners' process of memorization, and to further understand the basic research question: what kind of words do second language learners actually know?

out-of-sample Settings in the vocabulary prediction task can be divided into two for handling new words: *in-matrix* and *out-of-sample*. The in-matrix setting does NOT support new

words, i.e., there is at least one training dataset for all the words appearing in the test data. This can be seen as filling in the blanks of a learner-word matrix. In contrast, the *out-of-sample* setting support new words, i.e., some or all words in the test data are missing in the training data. To create the training data, we need to ask learners whether or not they know the words. Thus, creation of the training data is very financially costly and burdensome for learners. In a realistic setting, we can ask learners about only a small subset of words, and the predictors usually have to predict all the rest. The out-of-sample setting is more difficult but more realistic than the in-matrix setting.

learner-specific word difficulty This is the core beneficial property of the proposed model. Some interpretable weight vectors can determine word difficulty. However, the perceived difficulty of a word differs from learner to learner. For example, a learner interested in music may know music-related words that even high-level learners may not be familiar with. For another example, suppose that normally difficult words are used in the names of well known commercial products and services. In this case, again, low-ability learners may know these words through the product names. Thus, it is preferable for a model to be able to detect this kind of learner specialty.

	weight vector is interpretable	out-of-sample	learner-specific word difficulty
Rasch	✓	-	-
Ehara et al. (2010)	✓	✓	-
Proposed	✓	✓	✓

Table 1: Properties of models. The proposed model supports all preferred properties. Ordinary binary classifiers only can classify: their weight vectors are not interpretable as *word difficulty* and *learner ability* as those of the other models listed here.

Table 1 summarizes the models explained in this paper. We can see that only the proposed model supports all the properties. Although ordinary binary classifiers, such as SVMs, can be used for the vocabulary prediction task, their weight vectors cannot be used to determine word difficulty and learner ability that we want for analysis. Thus, we ruled out typical binary classifiers.

The structure of this paper is as follows. We first focus on extending the basic interpretable model: the Rasch model (Rasch, 1960; Baker and Kim, 2004). Although the Rasch model lacks many of the preferred properties, it provides a rough idea for the vocabulary prediction task. To explain why the Rasch model lacks many of these properties, we then introduce *the general form* of the likelihood of the Rasch model. This generalization provides a way of supporting the preferred properties. Through this generalization, we can derive the Rasch model, the model proposed by Ehara et al. (2010), and the proposed model.

The contributions of this paper are as follows:

- We introduce the general form of likelihood of the Rasch model that can explain the reason this model lacks the desired properties.
- We propose a model that supports all desired properties using this general form.
- In an evaluation, our model successfully detected the specialties of second language learners, which the current models cannot detect.

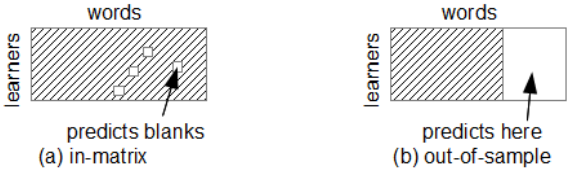


Figure 1: Two problem settings; (a) in-matrix, (b) out-of-sample.

2 Problem setting

Let U be a set of learners, and V be a set of vocabulary. We denote the number of learners as $|U|$ and the number of words as $|V|$. A datum can be expressed using the triplet (y, u, v). Here, $y \in \{0, 1\}$ is the label denoting whether or not learner u knows word v, $(1, u, v)$ means that learner u knows word v, and $(0, u, v)$ means he/she does not know word v. Using these notations, the vocabulary prediction task is defined to predict the label y given (u, v). We denote a dataset of N data as $\mathcal{D} = \{(y_1, u_1, v_1), \ldots, (y_N, u_N, v_N)\}$.

For simplicity, we assume that for one learner $u \in U$ and word $v \in V$ pair, there exists only one label y. This restriction enables us to depict the data set in a matrix form, as shown in Figure 1. The rows of the matrix correspond to learners and the columns of the matrix correspond to words. Under this assumption, for one row (learner) and one column (word), there is only one cell; thus, only one label y. With this restriction, N is the number of cells in the matrix.

The dataset we used in the evaluation agrees with this restriction; however, we cannot always assume this restriction in a realistic setting. This is the reason we did not directly jump to matrix-based prediction methods such as low-rank approximation using singular value decomposition. For example, in a realistic dataset, such as word-click logs in a reading support system, contradiction and repetition are common. For contradiction, if both $(1, u, v)$ and $(0, u, v)$ appear in the dataset, it may mean these two datasets are unreliable. Repetition of multiple $(1, u, v)$ may mean that learner u is more familiar with word v than just one $(1, u, v)$. All the models that we explain in the later sections of this paper can handle these cases.

Figure 1 explains the *in-matrix* and *out-of-sample* settings. The hashed areas denote the training data, and the blank areas denote the test data. In the *in-matrix* setting, the test data are randomly placed in the matrix.

3 Rasch model

Although the vocabulary prediction task is quite novel, there have been a substantial amount of work in SLA about which words a learner *should* learn first. Many studies recommend learners to learn words according to word frequency in general corpora because word frequency can be used as a rough measure of word difficulty. Of course, the learner does not necessarily learn the words in this recommended order. As stated in the introduction, it is one of our research questions to check if learners *actually* learn in this order.

Still, we can come up with the idea that the difficulty of words determines the learners' knowledge of second language words. This idea leads to a very simple model of vocabulary prediction shown in Figure 2. With this model, we predict a learner's vocabulary with the following steps:

1. We rank words according to a measure of word difficulty.
2. We decide the threshold for a learner.
3. Words with greater difficulty than the threshold are predicted to be unfamiliar to the learner, and vice versa.

Although this model seems too simple, it is the core idea of the Rasch model, which has been widely used in language testing.

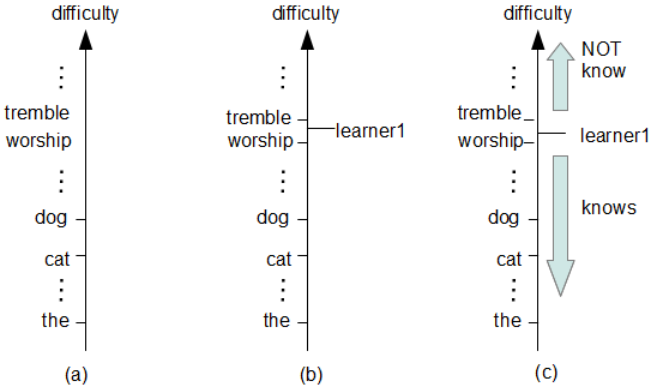


Figure 2: Simple vocabulary prediction model. (a) First, assume there is a difficulty measure that maps each word to a point on the axis of the measure. (b) Second, each learner's ability is also mapped to a point on the same axis. (c) Third, the words with the greatest difficulty to the point designating the learner's ability is predicted to be unfamiliar to the learner, and vice versa.

Given learner u and word v, the Rasch model models the probability of learner u knowing word v as follows:

$$P\left(y=1|u,v\right)=\sigma\left(a_u-d_v\right), \tag{1}$$

where $\sigma\left(t\right)=\left(1+\exp(-t)\right)^{-1}$ denotes the logistic sigmoid function. There are two kinds of parameters to be trained:

d_v the difficulty of word v,
a_u the ability of learner u.

In the Rasch model, the subtraction of two parameters $a_u - d_v$ in Eq. (1) denotes exactly the same mechanism as the simple vocabulary prediction in Figure 2. Here, d_v maps each word v into a point on the axis, and a_u works as a threshold. When $P(y=1|u,v)\geq 0.5$, we can assume learner u knows word v. Due to the logistic sigmoid function, $P(y=1|u,v)\geq 0.5$ holds true if and only if $a_u - d_v \geq 0$, that is, $a_u \geq d_v$. Therefore, the Rasch model determines that learner u knows all words whose word difficulty d_v is lower than the learners' ability a_u. Note that not only the ability of learner a_u but also the difficulty of word d_v is estimated from the data in the Rasch model.

The priors for the parameters are usually set as follows:

$$P\left(a_u|\eta_a\right) = \mathcal{N}\left(0,\eta_a^{-1}\right) \quad (\forall u\in U), \tag{2}$$
$$P\left(d_v|\eta_d\right) = \mathcal{N}\left(0,\eta_d^{-1}\right) \quad (\forall v\in V), \tag{3}$$

where $\mathcal{N}$ denotes the probability distribution function of the normal distribution. Frequently, the hyper parameters η_a and η_d are set as $\eta_a = \eta_d$. If $\eta_a = \eta_d$, the parameters, d_v and a_u of the Rasch model can be obtained using a standard log-linear model solver.

One of the notable problems with the Rasch model is that it does not take into account the *out-of-sample* setting. That is, it cannot predict words that do not appear in the training set. For example, if there is a new word in a document in a reading support system, we need to re-create the training set with the new word for the system to be able to predict that word as well. This restriction makes the application systems using the vocabulary prediction task impractical.

4 General form of likelihood

In the previous section, we stated that the Rasch model does work under the out-of-sample setting, which frequently occurs in a realistic setting. This section attempts to locate the fundamental reason the out-of-sample problem arises by generalizing the likelihood of the Rasch model.

Let us discuss the difficulty parameter d_v of the Rasch model from another perspective. If we define a function as $f(v) = d_v$, we can understand that d_v is a function that takes word v as its argument and returns the difficulty of word v. This means that we do not need to allocate the number of variables $|V|$ to determine the difficulty of a word as the Rasch model does. Instead, all that we need is a function that returns word difficulty for given word v.

We can further extend f to be the form $f(u, v)$: a function that takes learner u and word v as its argument and returns the difficulty of word v for learner u. By using $f(u, v)$, we can generalize the likelihood function of the Rasch model as follows:

$$P\left(y = 1|u, v\right) = \sigma\left(a_u - f\left(u, v\right)\right). \tag{4}$$

The Rasch model is a special version of Eq. (4) where we set $f(u, v) = d_v$. We can see that the fundamental cause of the out-of-sample problem in the Rasch model comes from this poorly designed f. There is a 1-to-1 mapping between parameters and words in this design of f. Therefore, if some words are missing in the training set, parameters arise that are not trained.

Note that Eq. (4) generalizes only the likelihood of the Rasch model. Of course, to fully define a model, we must define priors as well. Moreover, the priors must be designed carefully; otherwise, a model can produce poor results regardless of the design of f.

One may think of extending the learner ability parameter a_u to be a function as well. Of course, we can do this extension in theory. However, unlike word difficulty parameters, little information is practically available for learners. Therefore, it is preferable for a model to require as little information from learners as possible. Since the complex design of f may require much information, we kept the learner ability parameter a_u simple.

4.1 Shared difficulty model

By redesigning f in the general form of likelihood, we can cope with the out-of-sample setting. One way to design f to be able to do this is to set it as $f(u, v) = \mathbf{w}^\top \boldsymbol{\phi}(v)$. Here, $\boldsymbol{\phi} : V \to \mathcal{R}^K$ is a feature function. Given word v, it returns a feature vector for it. Let K be the dimension of the feature space. Typically, frequencies from large corpora can be used as features.

Even if there is a new word in the test data and there are words in the training data that share the same features with the new word, the word difficulty of the new word can be obtained by calculating $\mathbf{w}^\top \phi(v)$. The full form of the likelihood becomes the following.

$$P\left(y=1|u,v;\mathbf{w}\right)=\sigma\left(a_u-\mathbf{w}^\top\phi\left(v\right)\right). \tag{5}$$

Priors for the likelihood Eq. (5) are set as follows. We call this model the shared difficulty model.

$$P\left(a_u|\eta_a\right) = \mathcal{N}\left(0,\eta_a^{-1}\right) \quad (\forall u\in U), \tag{6}$$

$$P\left(\mathbf{w}|\eta_w\right) = \mathcal{N}\left(\mathbf{0},\eta_w^{-1}I\right), \tag{7}$$

where I denotes the $K\times K$-sized identity matrix. If we set $\eta_w=\eta_a$, this model reduces to a simple l2-norm-regularized logistic regression as Ehara et al. (2010) used. However, they did not mention the out-of-sample setting or the general likelihood.

5 Proposed model

One problem in both the Rasch and shared difficulty models is that all learners share a single word difficulty measure. This means that the same ranking of a word is shared by all the learners, e.g., the word "tremble" is more difficult than *worship* according to all the learners. Thus, the Rasch and shared difficulty models cannot take into account a leaner's specialty.

In reality, it is common that even low-ability learners know difficult words with the help of their interests in a specific topic. For example, learners who are interested in music are likely to have a large vocabulary of music-related words in second languages regardless of the difficulty of the words. Modeling this kind of learner specialty is essential in designing user-adaptive supports for second language learners.

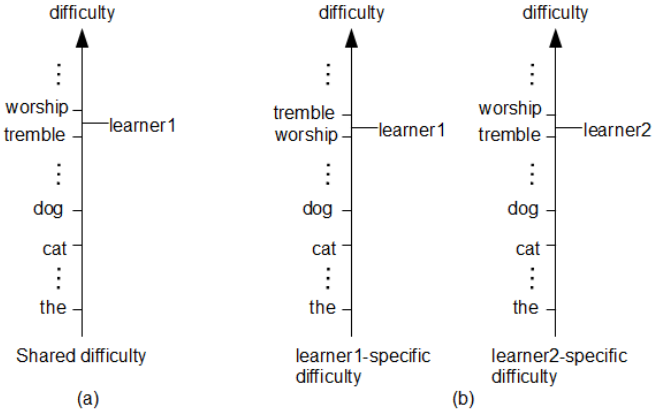


Figure 3: Learner-specific word difficulty.

Figure 3 illustrates the difference between the shared word difficulty and learner-specific word difficulty. On the left side of the difficulty axis, words are plotted according to difficulty. On the

Name	Design of f	Priors	Notes
Rasch	$f(u,v) = d_v$	$P\left(a_u|\eta_a\right) = \mathcal{N}\left(0, \eta_a^{-1}\right)$ $P\left(d_v|\eta_d\right) = \mathcal{N}\left(0, \eta_d^{-1}\right)$	-
Shared difficulty model (Ehara et al., 2010)	$f(u,v) = \mathbf{w}^\top \phi(v)$	$P\left(a_u|\eta_a\right) = \mathcal{N}\left(0, \eta_a^{-1}\right)$ $P\left(\mathbf{w}|\eta_w\right) = \mathcal{N}\left(\mathbf{0}, \eta_w^{-1} I\right)$	Reduced to Rasch model if $\phi(v)$ is 1-dimensional and $\phi(v) = 1$.
Proposed	$f(u,v) = \mathbf{w}_u^\top \phi(v)$	$P\left(a_u|\eta_a\right) = \mathcal{N}\left(0, \eta_a^{-1}\right)$ $P\left(\mathbf{w}_0\right) = \mathcal{N}\left(\mathbf{0}, \eta_w^{-1} I\right)$ $P\left(\mathbf{w}_u|\mathbf{w}_0\right) = \mathcal{N}\left(\mathbf{w}_0, \lambda^{-1} I\right)$	Reduced to the shared difficulty model if we set $\mathbf{w}_u = \mathbf{w_0}$ $(\forall u \in U)$.

Table 2: Summary of models explained so far. The Rasch model is a special case of the shared difficulty model, and the shared difficulty model is a special case of the proposed model.

right side of the axis, learner thresholds are plotted according to the learners' ability parameters a_u. The predictor determines that a learner does not know all the words above his/her threshold. In Figure 3 (a), all three learners share the same word difficulty. Therefore, the model cannot represent a learner who knows the word "worship" but does not know the word "tremble". This problem can be solved by introducing a difficulty axis for every learner as Figure 3 (b) does. In (b), "learner 1" is modeled as knowing the word "worship" but not the word "tremble", while "learner 2" is modeled as knowing the word "tremble" but not the word "worship". This kind of flexible modeling is impossible in the Rasch and shared difficulty models.

With the general model explained above, we can easily explain the fundamental cause of this problem: in the current models, $f(u,v)$ depends only on v, and does not depend on u. Therefore, tackling this problem is simple: let $f(u,v)$ depend on u as well. In the proposed model, we define $f(u,v) = \mathbf{w}_u^\top \phi(v)$. The full form of the likelihood is shown as follows.

$$P\left(y = 1|u, v; \mathbf{w}_u\right) = \sigma\left(a_u - \mathbf{w}_u^\top \phi(v)\right). \tag{8}$$

This likelihood has far more parameters to be trained than the current models. Since the dimension size of the feature space is K, $\mathbf{w}_u$ is a K-dimension vector. Since we have $|U|$ learners, we have $K|U|$ parameters to tune in total. Priors must be carefully designed to tune this large number of parameters. We designed the priors as follows:

$$P\left(a_u|\eta_a\right) = \mathcal{N}\left(0, \eta_a^{-1}\right) \quad (\forall u \in U), \tag{9}$$

$$P\left(\mathbf{w}_0\right) = \mathcal{N}\left(\mathbf{0}, \eta_w^{-1} I\right), \tag{10}$$

$$P\left(\mathbf{w}_u|\mathbf{w}_0\right) = \mathcal{N}\left(\mathbf{w}_0, \lambda^{-1} I\right). \tag{11}$$

Eq. (11) is an important prior that does not appear in the current models. This prior makes $\mathbf{w}_u$ close to $\mathbf{w}_0$ and makes $\mathbf{w}_u$ dependent on each other. The larger the λ, the stronger this effect.

Note that both the shared difficulty model discussed by Ehara et al. (2010) and the Rasch model are actually special cases of the proposed model; we *extended* the Rasch and shared difficulty models into the proposed model. The constraints to reduce the proposed model into these models are summarized in Table 2.

6 Estimation of model parameters

This section describes methods for estimating the model parameters. We use maximum-a-posteriori (MAP) estimation for all three models: Rasch, shared difficulty, and proposed. As we explained, the shared difficulty and Rasch models are special cases of the proposed model. Therefore, we first explain the optimization of the proposed model.

The negative log of the negative log posterior of the proposed model takes the following form:

$$l\left(\mathbf{W},\mathbf{a},\mathbf{w_0}\right) = \sum_{i=1}^{N} nll\left(y_i,u_i,v_i\right) + \frac{\lambda}{2}\sum_{u\in U}\left\|\mathbf{w}_u-\mathbf{w_0}\right\|^2 \quad (12)$$

$$+ \frac{\eta_w}{2}\left\|\mathbf{w}_0\right\|^2 + \frac{\eta_a}{2}\sum_{u\in U} a_u^2 \ . \quad (13)$$

We define the negative log likelihood function of the proposed model as $nll\left(y,u,v\right) \stackrel{\text{def}}{=} \log\left(1+\exp\left(-y\left(a_u-\mathbf{w}_u^\top\phi\left(v\right)\right)\right)\right)$. We define $\mathbf{W}$ and $\mathbf{a}$ as follows for concise notation: $\mathbf{W}=\{\mathbf{w}_u|\forall u\in U\}$, $\mathbf{a}=\{a_u|\forall u\in U\}$. This function $l\left(\mathbf{W},\mathbf{a},\mathbf{w_0}\right)$ is convex (Kajino et al., 2012) over all the variables $\mathbf{W},\mathbf{a},\mathbf{w_0}$. Thus, the MAP model parameters $\hat{\mathbf{W}},\hat{\mathbf{a}}$, and $\hat{\mathbf{w}_0}$ can be estimated by minimizing $l\left(\mathbf{W},\mathbf{a},\mathbf{w_0}\right)$ w.r.t. $\mathbf{W}$, $\mathbf{a}$, and $\mathbf{w}_0$.

Based on Kajino et al. (2012), we minimize $l\left(\mathbf{W},\mathbf{a},\mathbf{w}_0\right)$ iteratively as follows:

minimizing w.r.t. W, a We fix $\mathbf{w}_0$ and minimize $l\left(\mathbf{W},\mathbf{a},\mathbf{w}_0\right)$ w.r.t. $\mathbf{W}$ and $\mathbf{a}$. Kajino et al. (2012) used the Newton method for this optimization. Using the Newton method requires $O(K^2)$ memory, where K is the dimension of $\mathbf{w}_u$ and $\mathbf{w}_0$. This is problematic when K increases. To tackle this problem, we used L-BFGS (Liu and Nocedal, 1989), which requires only $O(K)$ memory, for this optimization instead. Specifically, we used the library liblbfgs (Okazaki, 2007).

minimizing w.r.t. $\mathbf{w}_0$ We fix $\mathbf{W}$ and $\mathbf{a}$ to minimize l w.r.t. $\mathbf{w}_0$. This minimization can be achieved analytically as follows:

$$\mathbf{w}_0 = \frac{\lambda}{\eta_w+|U|\lambda}\sum_{u\in U}\mathbf{w}_u \ . \quad (14)$$

We repeated these two minimizations iteratively until convergence.

Both the Rasch and shared difficulty models are special cases of the proposed model when $\mathbf{w}_u=\mathbf{w_0}$ $(\forall u\in U)$. This means that the second minimization is unnecessary for the Rasch and shared difficulty models. Thus, the parameters, i.e., the weight vector, of the Rasch and shared difficulty models can be obtained by simply performing the first minimization.

7 Evaluation

7.1 Dataset

We used the same dataset as Ehara et al. (2010) used. The dataset was created in Japan in January 2009. Sixteen English as a second language learners participated in the creation of this dataset. Most were graduate students of the University of Tokyo, and Japanese was the native language of most of them.

Corpus name	Type of English	Size (in token)	Description
British National Corpus (BNC) (The BNC Consortium, 2007)	British	100 mil.	General corpus
The Corpus of Contemporary American English (COCA) (Davies, 2011)	American	450 mil.	General corpus
Open American National Corpus (OANC) (Ide and Suderman, 2007)	American	14 mil.	General corpus
Brown corpus (Francis and Kucera, 1979)	American	1 mil.	General corpus
Google 1-gram (Brants and Franz, 2006)	Mixed	1,024,948 mil.	Huge, but not general

Table 3: Feature sources.

This dataset was designed to be quite exhaustive. Every learner was handed a randomly sorted questionnaire comprising 12,000 words and asked to answer how well he/she knew the words in the questionnaire based on a five-point scale. We regarded level 5 as only $y = 1$; the learner knows the word. Otherwise we regarded $y = 0$; the learner does not know the word. Out of the 12,000 words, 1 word was a pseudo-word, i.e., it looks like an English word but actually is not.

Fifteen learners were paid, and 1 learner was not. Since we found that the unpaid learner's data were too noisy, we used only the data of the 15 paid learners. We had $|V| = 11,999$ words $\times$ $|U| = 15$ learners; 179,985 data points in total.

The negative log of the 1-gram probabilities of each word in each corpus is used as features for training. The collected corpora for feature sources are compiled in Table 3. Ehara et al. (2010) used one large corpus, Google-1gram. However, from the perspective of SLA, it is typically not justified because it is not a general corpus; thus, its frequencies could be biased. To avoid being biased, we collected many general corpora and used them as features.

When training, hyper parameters were chosen by grid search and 5-fold cross validation within the training set. The set of hyper parameters that performed best in this cross validation was selected. Then, we trained the model with all the training sets using the selected hyper parameters. We then applied the model to the test set to obtain the results. For the Rasch and shared difficulty models, each hyper parameter, η_d, η_a, and η_w, was chosen by grid search from $\{0.01, 2^{-3}, 2^{-2}, 2^{-1}, 1.0, 2^1, 2^2, 2^4\}$. For the proposed model, each hyper parameter, η_a, η_w, and λ, was chosen by grid search from $\{2^{-2}, 2^{-1}, 1.0, 2^1, 2^2\}$.

7.2 Evaluation of learner-specificity

Unlike the current models, the proposed model was designed to support learner-specific word difficulty. It is interesting to see which words are the most learner-specific.

For a measure of learner-specificity, we introduce the *variance* of learner-specific word difficulty. In the proposed model, the learner-specific difficulty $f(u,v)$ of word v for learner u is defined as $f(u,v) = \mathbf{w}_u^\top \phi(v)$. Unlike the current models that assign single word difficulty for all learners, we can naturally define the variance of word difficulty over learners. Given the set of estimated weight vectors for all $|U|$ learners, $\{\hat{\mathbf{w}}_u \mid u \in U\}$, for word $v \in V$, we define $Mean(v)$ and $Var(v)$ as follows:

$$Mean(v) \stackrel{\text{def}}{=} \frac{1}{|U|}\sum_{u\in U} f(u,v) = \frac{1}{|U|}\sum_{u\in U} \hat{\mathbf{w}}_u^\top \phi(v), \tag{15}$$

$$Var(v) \stackrel{\text{def}}{=} \frac{1}{|U|}\sum_{u\in U} \Big(f(u,v) - Mean(v)\Big)^2 = \frac{1}{|U|}\sum_{u\in U} \Big(\hat{\mathbf{w}}_u^\top \phi(v) - Mean(v)\Big)^2. \tag{16}$$

Table 4 lists the words with largest variances $Var(v)$ in descending order. $Var(v)$ increases when some low-ability learners know the words and some high-ability learners do not. In other words, it increases when low-ability learners know the word for some reason other than the easiness of the word, and vice versa. Table 4 is constructed from the weight vectors of the proposed model. The weight vectors are trained in the in-matrix setting. Out of 179,985 data points, 177,985 were used for training. Features and hyper parameter tuning are explained in §7.1. 2,000 data points were used to check the accuracy, which was 83.40%.

For example, it is very interesting and noteworthy that the word "twitter" comes at the top of the list of Table 4. This is presumably due to the famous micro-blogging service, Twitter. The word "twitter" itself is a rare word. For example, in the British National Corpus, the frequency of the word "twitter" is merely 17 while the word "the" is 6,043,900. The words whose frequency is the same with the word "twitter" are: "abet", "beguile", and "coddle". Since these three words are in the dataset as well, the rareness of words only cannot explain the large variance of the word "twitter". This dataset was created in Japan in January 2009 when Twitter was not as predominant as it is today. Therefore, some low-level learners knew the word "twitter" through the name of the service while some high-level learners did not. Additionally, Table 4 ranks another similar example at the third: "kindle". The first Amazon Kindle was released in the United States in 2007.

Likewise, we annotated presumable reasons $Var(v)$ increased in the rightmost column of Table 4. Although these reasons are speculation, it is difficult to find the correct reason learners know a word, even for learners themselves, because we usually do not remember how we learned foreign words. Our speculations are intuitive and understandable for Japanese-native English as a Second Language (ESL) learners.

Product name The words "twitter" and "kindle" correspond to this case. When a difficult word is used as the name of a famous product, it is possible that even low-ability learners would know the word through the name of the product, which makes the variance larger.

Loanwords in L1 Some words in the second language are borrowed by the learners' native language, or L1, i.e., *loanwords*. However, the spelling of loanwords in L1 can differ from its original. For example, in the case of the word "mantle", the corresponding loanword in Japanese, the native language for most of learners of the dataset used, is spelled as "mantoru". Therefore, the difficulty has little influence on whether or not learners know the word in this case. Rather, whether or not the learner can perceive the loanword in spite of spelling difference has more influence. Thus, even low-ability learners can

$Var(v)$	word	presumed cause of learner-specificity
0.993	twitter	product name
0.886	waltz	topic specific: music, loanword in L1
0.849	kindle	product name
0.833	rink	homophone in L1 with “link”
0.827	launder	loanword in L1
0.825	bass	topic specific: music
0.823	ultraviolet	topic specific: cosmetics
0.818	chime	topic specific: music
0.804	asphalt	loanword in L1
0.802	harry	homophone in L1 with “hurry”
0.793	wooded	-
0.776	mantle	loanword in L1
0.767	trombone	loanword in L1
0.766	modulate	topic specific: computer programming
0.763	homeroom	loanword in L1
0.760	harness	-
0.760	bog	-
0.755	hearth	confused with “health”
0.750	convent	-
0.748	hurdle	loanword in L1
0.733	parson	homophone in L1 with “person”
0.732	vector	loanword in L1
0.731	haven	homophone in L1 with “heaven”
0.719	gadget	loanword in L1
0.714	lizard	-
0.713	smelt	homonym in English: past particle of “smell”
0.709	shin	homophone in L1 with “sin”
0.708	placebo	loanword in L1
0.707	lagoon	-
0.702	aha	-

Table 4: Top 30 words with largest variances $Var(v)$ in descending order. Large $Var(v)$ suggests large learner-specificity. Japanese is the native language (L1) of this dataset.

perceive the meaning of the word through its corresponding loanword in L1, which makes the variance larger.

Homophones in L1 If there are two words that are homophones in the learners' native language, and one of the two words is easier than the other, a low-ability learner may mistake the difficult one for the easy one. For example, a large variance of the word “rink” is caused by low-ability learners' mistake for the word “link” because the Japanese language does not distinguish “l” and “r”. For example, Japanese has no distinction between “par” and “per”, the large variance of the word “parson” is presumably due to some learners mistaking this word for the word “person”.

Topic specific Low-ability learners interested in a topic are likely to know the words of that topic regardless of the words' difficulty.

Homonyms in English "smelt" is a verb that means extracting metals by heat. Yet, it is also the past participle of the word "smell". Although the conjugated forms were removed from this dataset, some low-ability learners presumably did not notice it and thought that they were asked if they knew the word "smelt" as the past participle of the word "smell". Some high-ability learners presumably knew that the word "smelt" has a meaning other than the past participle of "smell" and not asked about "smelt" as the past participle. If they did not know what was the meaning other than the past participle of "smell", they answered no in the dataset.

Note that the variance of the learners' response y for a word in the raw data *cannot* produce an interesting listing as in Table 4 because y is binary, 0 or 1. It trivially lists words of which half the learners in the dataset know. For example, if there are 15 learners in a data set, it is trivial to determine the words with the highest variance of y as those that 8 learners knew and 7 learners did not, or 7 learners knew and 8 learners did not. This means that many words have the highest y variance. In this dataset, $1,408$ of $11,999$ words had the highest y variance. Therefore, y variance does not produce any interesting results.

In contrast to Table 4, the words with smallest $Var(v)$ are trivial. They are words all the learners knew or all the learners did not know. The 30 words with the smallest variances were: am, beach, doll, during, eastern, equal, excellent, green, handwriting, hungry, important, logic, love, luck, marine, paradise, shop, technical, writing, pet, unknown, loose, maker, acquittal, arduous, cot, exchequer, hindsight, innuendo, and purr.

Finally, we investigated the accuracy in the out-of-sample setting. We split the $11,999$ words into $2,000$ words for the test set and the rest for the training set. The size of training data was $149,985$ and the size of test data was $30,000$. Hyper parameter tuning and feature set were the same as we stated in §7.1. The Rasch model achieved 66.32%, the shared difficulty model (Ehara et al., 2010) achieved 77.67%, and the proposed model achieved 77.81%.

8 Related Work

The proposed model is mathematically very similar to those proposed by Evgeniou and Pontil (2004) and Kajino et al. (2012). However, these models are for totally different purposes than ours: Evgeniou and Pontil (2004) aimed at multi-task learning and Kajino et al. (2012) aimed at crowd-sourcing. As the Rasch model is rarely used for these purposes, they did not mention the relationship between the Rasch and proposed models, let alone the generalization of the likelihood of the Rasch model. Strictly speaking, these two models differ from our model in that they do not include the Rasch and shared difficulty models (Ehara et al., 2010) as special cases while our proposed model does.

We extended word difficulty to learner-specific word difficulty by focusing on the analysis of the vocabulary knowledge of adult second language learners. Aside from second languages, study of vocabulary knowledge is also important for the analysis of child development in terms of native language. In computational linguistics, Kireyev and Landauer (2011) proposed an extension of word difficulty called "word maturity" by focusing on the analysis of child development in terms of native language. Their extension was aimed at "track the degree of knowledge of each word at different stages of language learning" using latent semantic analysis. Thus, both their purpose and method of extending word difficulty differ from ours.

While few have studied the vocabulary prediction task, prediction of text readability has been of great focus (François and Fairon, 2012; Feng et al., 2010; Kate et al., 2010) in computational

linguistics. The relationship between vocabulary knowledge and text readability has been thoroughly studied by educational experts (Nation, 2006).

A substantial amount of work has been done by mainly SLA experts in estimating vocabulary *size*. Two major testing approaches have been proposed: *multiple-choice*, (Nation, 1990), and *Yes/No* (Meara and Buxton, 1987). For *Yes/No* tests, Eyckmans (2004) studied the validity and relation to readability prediction.

In the field of psychology, the shared difficulty model (Ehara et al., 2010) is almost mathematically identical to the linear logistic test model (LLTM) (Fischer, 1983). Also, the vocabulary that humans memorize is studied as "mental lexicon" (Amano and Kondo, 1998), although most of the mental-lexicon work is not aimed at predicting vocabulary.

Conclusion

We proposed a model for the vocabulary prediction task. Although there have been few studies on it, it is interesting from both theoretical and practical points of views.

We introduced three preferred properties for predictors for this task: *interpretable weight vector*, *out-of-sample setting*, and *learner-specific word difficulty*. Typical machine-learning classifiers, such as SVMs, lack the first property, interpretable weight vector. Although the Rasch model has this property, it lacks the latter two properties.

To understand why the Rasch model lacks the latter two properties, we introduced the general form of the Rasch model. From this general form, we derived our proposed model, which supports the latter two properties.

In the qualitative evaluation, we wanted to see which words are the most learner-specific. Therefore, we introduced the variance of learner-specific word difficulty and listed the top 30 words with largest variances. The results exhibited social aspects of the learners. For example, "twitter" and "kindle" came first and third, which suggests that some low-ability learners know these words through service and product names, although they are usually difficult English words. Note that this analysis is possible because the proposed model supports the third property, learner-specific difficulty. Since the current models do not support this property, this analysis is impossible with these models. Moreover, the proposed model achieved accuracy competitive with the current models under the out-of-sample setting, which is more realistic than the in-matrix setting.

Future work includes using topic models to determine learners' specialties. We also plan to introduce a sparse prior, such as Laplace prior, instead of Gaussian prior on the user-specific weight vector in Eq. (11) to obtain a more concise model in which the weights specific to each user only deviate from the overall weights.

References

Amano, S. and Kondo, T. (1998). Estimation of mental lexicon size with word familiarity database. In *Fifth International Conference on Spoken Language Processing*.

Baker, F. B. and Kim, S.-H. (2004). *Item Response Theory: Parameter Estimation Techniques*. Marcel Dekker, New York, second edition.

Brants, T. and Franz, A. (2006). Web 1T 5-gram Version 1. LDC2006T13.

Davies, M. (2011). N-grams data from the corpus of contemporary american english (coca). Downloaded from `http://www.ngrams.info` on June 23, 2012.

Ehara, Y., Shimizu, N., Ninomiya, T., and Nakagawa, H. (2010). Personalized reading support for second-language web documents by collective intelligence. In *Proceedings of the 15th international conference on Intelligent user interfaces (IUI 2010)*, pages 51–60, Hong Kong, China. ACM.

Evgeniou, T. and Pontil, M. (2004). Regularized multi–task learning. In *Proceedings of the 10th ACM SIGKDD international conference on Knowledge discovery and data mining (KDD 2004)*, pages 109–117. ACM.

Eyckmans, J. (2004). *Measuring receptive vocabulary size : reliability and validity of the yes/no vocabulary test for French-speaking learners of Dutch*. PhD thesis, Radboud University Nijmegen.

Feng, L., Jansche, M., Huenerfauth, M., and Elhadad, N. (2010). A comparison of features for automatic readability assessment. In *Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010): Posters*, pages 276–284, Beijing, China. Coling 2010 Organizing Committee.

Fischer, G. (1983). Logistic latent trait models with linear constraints. *Psychometrika*, 48(1):3–26.

François, T. and Fairon, C. (2012). An "ai readability" formula for french as a foreign language. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL 2012)*, pages 466–477, Jeju Island, Korea. Association for Computational Linguistics.

Francis, W. N. and Kucera, H. (1979). Brown corpus manual. Brown university, Rhodes island, third edition.

Ide, N. and Suderman, K. (2007). The open american national corpus (oanc). Corpus available at `http://www.AmericanNationalCorpus.org/OANC/` (Retrieved on October 24, 2012).

Kajino, H., Tsuboi, Y., and Kashima, H. (2012). A convex formulation for learning from crowds. In *Proceedings of the 26th Conference on Artificial Intelligence (AAAI-2012)*, pages 73–79, Tronto, Ontario, Canada.

Kate, R., Luo, X., Patwardhan, S., Franz, M., Florian, R., Mooney, R., Roukos, S., and Welty, C. (2010). Learning to predict readability using diverse linguistic features. In *Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010)*, pages 546–554, Beijing, China. Coling 2010 Organizing Committee.

Kireyev, K. and Landauer, T. K. (2011). Word maturity: Computational modeling of word knowledge. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL-HLT 2011)*, pages 299–308, Portland, Oregon, USA. Association for Computational Linguistics.

Laufer, B. and Nation, P. (1999). A vocabulary-size test of controlled productive ability. *Language testing*, 16(1):33–51.

Liu, D. and Nocedal, J. (1989). On the limited memory BFGS method for large scale optimization. *Mathematical Programming*, 45(1):503–528.

Meara, P. and Buxton, B. (1987). An alternative to multiple choice vocabulary tests. *Language Testing*, 4(2):142–154.

Nation, I. S. P. (1990). *Teaching and Learning Vocabulary*. Heinle and Heinle, Boston, MA.

Nation, I. S. P. (2006). How large a vocabulary is needed for reading and listening? *Canadian Modern Language Review*, 63(1):59–82.

Okazaki, N. (2007). *libLBFGS: L-BFGS library written in C*. Software available at `http://www.chokkan.org/software/liblbfgs/` (Retrieved on October 24, 2012).

Rasch, G. (1960). *Probabilistic Models for Some Intelligence and Attainment Tests*. Danish Institute for Educational Research, Copenhagen.

Schmitt, N., Schmitt, D., and Clapham, C. (2001). Developing and exploring the behaviour of two new versions of the vocabulary levels test. *Language Testing*, 18(1):55–88.

The BNC Consortium (2007). The british national corpus, version 3 (bnc xml edition). Distributed by Oxford University Computing Services on behalf of the BNC Consortium. URL: `http://www.natcorp.ox.ac.uk/` (Retrieved on October 26, 2012).

Jointly Disambiguating and Clustering Concepts and Entities with Markov Logic

Angela Fahrni[1] *Michael Strube*[1]
(1) Heidelberg Institute for Theoretical Studies, Schloss-Wolfsbrunnenweg 35, 69118 Heidelberg, Germany
Angela.Fahrni@h-its.org, Michael.Strube@h-its.org

ABSTRACT
We present a novel approach for jointly disambiguating and clustering known and unknown concepts and entities with Markov Logic. Concept and entity disambiguation is the task of identifying the correct concept or entity in a knowledge base for a single- or multi-word noun (mention) given its context. Concept and entity clustering is the task of clustering mentions so that all mentions in one cluster refer to the same concept or entity. The proposed model (1) is *global*, i.e. a group of mentions in a text is disambiguated in one single step combining various global and local features, and (2) performs disambiguation, unknown concept and entity detection and clustering *jointly*. The disambiguation is performed with respect to Wikipedia. The model is trained once on Wikipedia articles and then applied to and evaluated on different data sets originating from news papers, audio transcripts and internet sources.

KEYWORDS: Word Sense Disambiguation.

Proceedings of COLING 2012: Technical Papers, pages 815–832,
COLING 2012, Mumbai, December 2012.

1 Introduction

Recent advances in knowledge extraction from resources such as Wikipedia have allowed to create various large-scale knowledge bases and concept networks such as Yago (Suchanek et al., 2008), DBpedia (Bizer et al., 2009) or WikiNet (Nastase and Strube, 2012). To exploit the wealth of world knowledge in these resources for natural language processing tasks such as information extraction, text segmentation or summarization, words and phrases in a document first need to be linked to the relevant entries in the respective knowledge base, i.e. to be disambiguated. This problem has been tackled quite successfully by systems such as *WikipediaMiner* (Milne and Witten, 2008), which are mainly based on word sense disambiguation techniques (Agirre and Edmonds, 2006; Navigli, 2009) thus giving research on word sense disambiguation a new spin.

Concept and entity disambiguation is the task of identifying the correct concept or entity in a knowledge base for a single- or multi-word noun (*mention*) given its context.[1] In this paper we disambiguate with respect to the English Wikipedia and consider each article as a concept. One advantage of linking to Wikipedia is that the internal hyperlinks can be used as training data.

Concept disambiguation models the relation between mentions and concepts (Figure 1a). For instance, the system needs to identify if the mention *crocodile* in the first text points to AMERICAN CROCODILES (ANIMAL), to CROCODILE (LOCOMOTIVE) or to the person RENÉ LACOSTE (TENNIS PLAYER) whose nick name is *Crocodile*. We define concept disambiguation as the task of disambiguating both common nouns such as *crocodile* or *biologist* and proper nouns such as FLORIDA or STATES. While the disambiguation of common nouns is usually called *word sense disambiguation* (WSD), the disambiguation of proper nouns is also known as *entity linking*.

Although most of the work in concept disambiguation and WSD assumes that the knowledge base is complete, several studies show that many mentions have no corresponding entry in the English Wikipedia: While Zhou et al. (2010) report that between 10% and 23.5% of the mentions can not be linked to Wikipedia, Lin and Etzioni (2012) report that one third of their mentions have no corresponding entry in Wikipedia. The task of identifying mentions with no corresponding concept in the respective knowledge base is also known as recognition of NILs. In the example in Figure 1 *Aldecoa* does not refer to any entity listed in the knowledge base.

Concept clustering solves the problem of missing concepts in knowledge bases by clustering mentions within and across documents so that mentions in one cluster refer to the same concept. These clustering approaches, also known as *cross-document coreference resolution*, *sense induction* or *unsupervised word sense disambiguation* (Pedersen, 2006), do not link mentions to entries in an existing knowledge base, but cluster mentions as illustrated in Figure 1b.

We integrate the two research lines of disambiguating and clustering concepts and present a novel approach for joint disambiguation and clustering using Markov Logic (ML). Given an already existing knowledge base, mentions are linked to their corresponding entry in this knowledge base, if one exists (Figure 1a). At the same time, mentions are clustered together with other mentions that refer to the same concept, regardless of whether the referred concept exists in the knowledge base or not (Figure 1b). Figure 1c shows the joint view. In contrast most previous approaches (including systems participating at TAC (Ji et al., 2011)) use three cascaded steps: (1) Disambiguation, (2) identification of NILs, (3) clustering of NILs.

The concept selections for the different mentions (e.g. *American crocodile* and *crocodile*) are interrelated. Joint disambiguation and clustering enables us to exploit such connections:

[1] Although we use in the following the term *concepts* instead of *concepts* and *entities*, we always mean both.

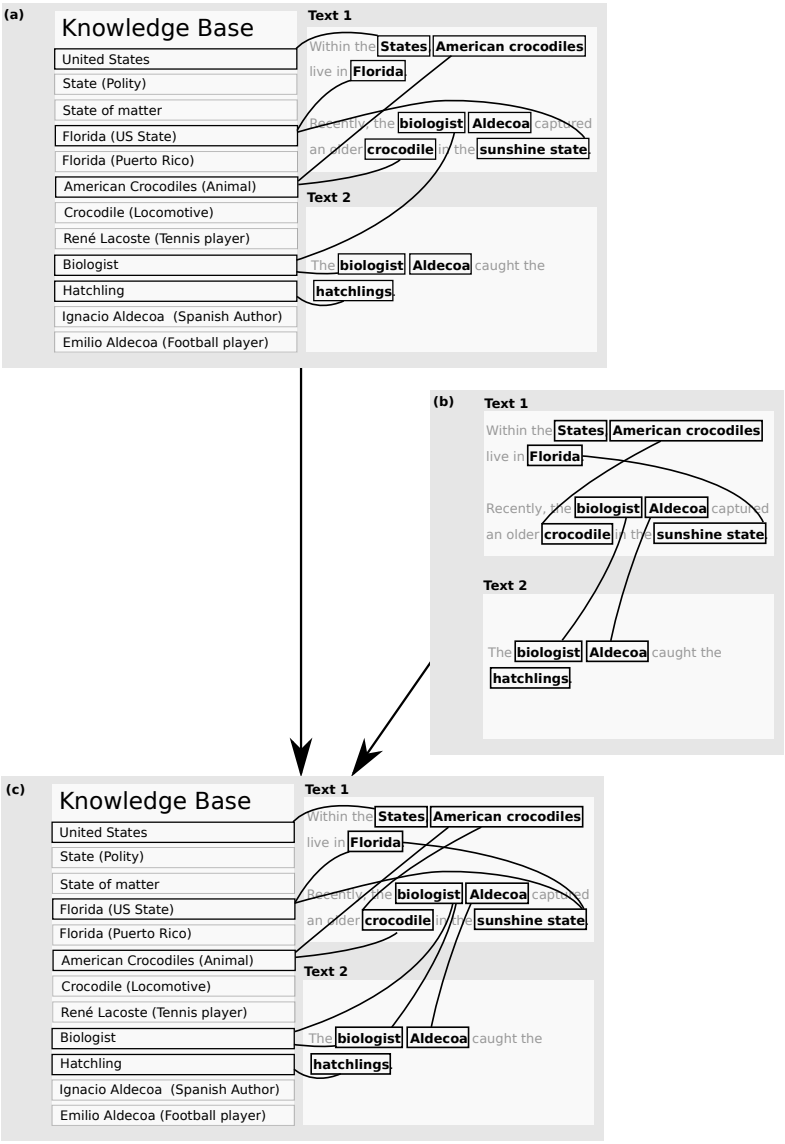


Figure 1: Joint concept disambiguation and clustering

knowledge about which mentions refer to the same concept can support disambiguation decisions. On the other hand, disambiguation influences clustering decisions. In contrast, local approaches which disambiguate mentions independently of each other (Milne and Witten, 2008; Csomai and Mihalcea, 2008) can not take advantage of such relations. Our joint approach disambiguates and clusters groups of mentions at the same time. By using Markov Logic we combine local and global features. Compared to other global models for WSD, e.g. Kulkarni et al. (2009), we do not just consider one single global feature, but combine different global features with local features and learn the weights for their combination.

Our model is trained on Wikipedia only and evaluated on ACE 2005 (annotated by Bentivogli et al. (2010)). Though we are mainly interested in the ACE data, because they provide us with annotations for proper and common nouns, we also evaluate on the TAC 2011 data which are only annotated for named entities. Nevertheless our system performs well compared to the systems participating at the TAC 2011 competition with a much smaller feature set – e.g. McNamee (2010) use 200 features – and without being trained on TAC data specifically.

The paper is organized as follows. Section 2 discusses related work, Section 3 presents our novel approach for joint disambiguation and clustering, and Section 4 presents and analyzes experiments based on ACE 2005 (Bentivogli et al., 2010) and the TAC 2011 data sets.

2 Related Work

In recent years research in monolingual and cross-lingual concept and entity disambiguation has been boosted by shared tasks such as the *Link the Wiki Track* at INEX[2], the *Cross-lingual Link Discovery Task* at NTCIR-9 (Tang et al., 2011) and the *Entity Linking Task* at TAC (McNamee and Dang, 2009; Ji et al., 2010; Ji and Grishman, 2011; Ji et al., 2011).

Dai et al. (2011) is the work that is closest to ours. In order to link gene mentions they perform entity disambiguation and recognition of the NILs at the same time using Markov Logic. In contrast to us they do not cluster mentions, but focus only on one specific type of mentions in a particular domain, namely mentions that refer to genes in a biomedical corpus.

The task of entity disambiguation, recognition of NILs and clustering has been approached in a cascaded way (Ji et al., 2011). Bunescu and Paşca (2006) first decide, if a mention refers to an entity in a knowledge base. Dredze et al. (2010) first disambiguate and then recognize the NILs. NIL recognition is often done by setting a threshold (Han and Sun, 2012). Monahan et al. (2011) interleave entity linking and clustering, but they do not approach the two tasks jointly. After disambiguation they cluster mentions. Then each cluster is assigned an entity in the knowledge base if there exists a corresponding one. Sil et al. (2012) circumvent the NIL problem by an open-database approach instead of disambiguating with respect to only one knowledge base.

Another strand of work that is similar to ours are global disambiguation approaches. While early work often uses local classifiers or rankers that select a concept for each mention independently (Csomai and Mihalcea, 2008; Milne and Witten, 2008; Dredze et al., 2010), recently, various global approaches have been proposed. Kulkarni et al. (2009) propose a method that maximizes local context-concept compatibility and global concept coherence. Fahrni et al. (2011) use a graph-based approach and select the best combination of concepts given the graph structure. Han and Sun (2012) use a generative model integrating topic coherence (one topic per document) and local context compatiblity. Ratinov et al. (2011) describe a two pass method and use

[2]http://www.inex.otago.ac.nz

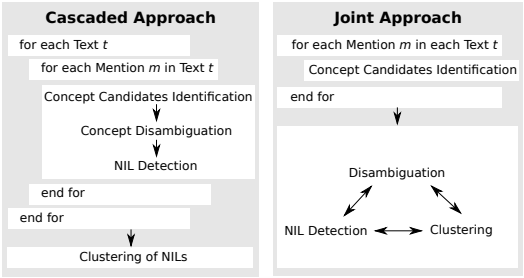


Figure 2: Cascaded approach vs. joint global approach.

the input of the first pass as input for the second one. While all these approaches use a limited number of global features, we integrate and learn the weights for various global features.

While we aim for a general domain disambiguation and clustering system that disambiguates and clusters common and proper nouns, the *Wikify!* (Csomai and Mihalcea, 2008) and *WikipediaMiner* (Milne and Witten, 2008) systems focus on the disambiguation of a few relevant keywords. Chen et al. (2012) only disambiguate person names, while Nothman et al. (2012) perform event linking.

The most prominent research line for sense induction are distributional approaches (Schütze, 1998). Pedersen (2006) gives an overview over state-of-the-art techniques. Recently, the efficency problem caused by the number of necessary comparisons has been addressed (Singh et al., 2011). While Rao et al. (2010) apply streaming clustering, Wick et al. (2012) propose a discriminative hierarchical model and partition entities into trees of latent sub-entities. None of these approaches for clustering also does concept disambiguation at the same time.

3 Approach

Markov Logic enables us to approach the task of disambiguation, recognition of unknown concepts and clustering jointly and to make use of global features. Instead of selecting for each mention – independently from earlier and later decisions – a concept, the concepts for a group of mentions are chosen at the same time.

Figure 2 contrasts the cascaded approach of disambiguation, recognition of unknown concepts and clustering with our joint global approach. As Figure 2 illustrates, first all candidate concepts for all mentions in a document are identified. Then disambiguation, recognition of NILs and clustering is performed using Markov Logic.

3.1 Markov Logic Networks

Markov Logic (ML) combines first-order logic with probabilities (Domingos and Lowd, 2009). A Markov Logic Network (MLN) consists of a set of pairs (F_i, w_i), where F_i is a first-order formula and $w_i \in \mathbb{R}$ is a weight associated with the formula F_i. It builds a template for constructing a Markov Network given a set of constants C. This Markov Network contains a binary node for each possible grounding for each predicate of the Markov Logic Network. If a ground predicate

is true the value of this binary node is 1, otherwise 0. In addition it contains one feature[3] for each ground formula. If a ground formula is true, the feature for this ground formula has the value 1, otherwise 0. The weight of the feature is given by w_i.

The probability distribution in the ground Markov Network is represented by

$$P(X = x) = \frac{1}{Z} \exp\left(\sum_i w_i n_i(x)\right)$$

where $n_i(x)$ is the number of true groundings of F_i in x. The normalization factor Z is the partition function.

To learn the weights for the formulas and to perform MAP inference we use *thebeast*.[4] *thebeast* employs cutting plane inference (Riedel, 2008) and enables us to perform discriminative training using a perceptron.

3.2 Disambiguation and Clustering with Markov Logic

The backbone of our model is the definition of how disambiguation, recognition of NILs and clustering interact. To model these relations we use hard constraints. In the following we will first describe these constraints, before we explain the features in the next section.

Table 1 shows all predicates and formulas used. Each formula is associated with a positive or negative weight. While the weight – except for hard constraints – is learned from training data, the polarity of the weights is set manually. In the following we indicate the direction by the + or − in front of each formula. Formulas with negative weights provide evidence for recognizing NILs. For some formulas the final weight consists of a learned weight w multiplied by a score s (e.g. prior probability). In these cases the final weight for a formula does not just depend on the respective formula, but also on the instantiation, e.g. a specific mention and candidate concept. We indicate such combined weights by the term $w \cdot s$, while w refers to cases where the formula is exclusively weighed by the learned weight. M denotes all mentions and C_m refers to all candidate concepts of a mention m.

Disambiguation and clustering are two different perspectives on the problem of lexical ambiguities. While in concept disambiguation the focus lies on the relation between mentions and concepts, clustering deals with relations between mentions. The tasks of disambiguation and recognition of unknown concepts are interrelated, as both tasks look at mention–concept relations. However, while in concept disambiguation the question is to which concept a mention refers to given its context, the task of recognizing unknown concepts is to determine, if such a concept relation exists for a given mention at all.

To approach disambiguation, recognition and clustering of NILs with ML we define a hidden predicate for each relation we are interested in. The predicate *hasConcept(Mention, Concept)* models the relation between mentions and concepts in the knowledge base (Table 1, p1). To ensure that each mention refers to at most one concept a hard cardinality constraint is defined: for each mention the predicate *hasConcept* is true at most once. This constraint allows us to jointly disambiguate and recognize NILs (Table 1, f1).

[3] Note that *feature* is used differently in this section than in the rest of the paper.
[4] `http://code.google.com/p/thebeast`.

To model if two mentions refer to the same concept, the predicate *hasSameConcept(Mention, Mention)* is used (Table 1, p2). It is true for all mention pairs that refer to the same concept, regardless whether the referred concept exists in the knowledge base or not. This clustering relation is *transitive* and *symmetric* (Table 1, f2, f3).

In order to perform joint disambiguation and clustering it needs to be defined how the mention-concept relation (disambiguation, recognition of NILs) and the clustering relation are interrelated (Table 1, f4, f5). Given that two mentions refer to the same concept in the knowledge base they belong to the same cluster (f5). On the other hand, if two mentions are part of the same cluster and one of them refers to a concept in the knowledge base, the other mention in the cluster has to refer to the same concept (f4). Note, two mentions can also be in the same cluster without referring to a concept in the knowledge base.

3.3 Features

All features have a corresponding predicate which is part of at least one formula. In the following we focus on the features.

3.3.1 Local Features

Local features involve one single mention and its candidate concepts.

Prior probability (p3, f7) The prior probability is defined as the probability that a mention m refers to a concept c. To estimate this probability all internal hyperlinks are extracted from the English Wikipedia dump. For each linked mention (m) it is counted, how many times it links to a particular Wikipedia page ($count_{m,c}$), i.e. concept. This count is normalized by the number of times mention m is linked to Wikipedia pages ($count_m$):

$$p(c|m) = \frac{count_{m,c}}{count_m}$$

Relatedness (p4, f8, f11) This feature reflects the average pairwise relatedness of a candidate concept for a mention to the context and is calculated in the same way as proposed by Milne and Witten (2008). The pairwise relatedness is calculated via

$$rel(c_1, c_2) = \frac{log(max(|C_1|, |C_2|)) - log(|C_1 \cap C_2|)}{log(|W|) - log(min(|C_1|, |C_2|)}$$

where c_1 and c_2 are two concepts, C_1 and C_2 denotes the articles in Wikipedia that link to the articles c_1 and c_2 respectively, and W is the total number of articles in Wikipedia. The more a candidate is related to the context, the more likely it is that a mention refers to it (f8). If a candidate concept for a mention is not at all related to the context, i.e. the average relatedness is zero, this is a negative indicator for a candidate (f11).

Local context similarity (p5, f9) The local context similarity measures how similar the current local context (K_m) – consisting of seven words before and after the mention – is to the local contexts for that concept in Wikipedia. For each mention in the English Wikipedia that is linked to a certain Wikipedia page c we extract the surrounding words (T_c) using the same context definition as above. We then calculate the local context similarity ($sim(c, m)$) for a candidate concept c of a mention m via

$$sim(c,m) = \frac{1}{|K_m|} \sum_{k \in K_m} s(k, T_c)$$

where the first term is used for normalization and $s(k, T_c)$ denotes the frequency of k in T_c divided by the number of times k appears in the context of all concepts in Wikipedia.[5]

String edit distance (p6, f10) This feature accounts for the difference between the mention string used in the text (m) and the preferred name (p) for a candidate concept of m. We assume that the Wikipedia article title and the titles of its redirects are preferred names for a concept (P). To measure the distance between preferred names and the mention in the text we calculate the edit distance[6] ($dist_{m,p}$) and normalize it by the length of the longer string:

$$sim_{m,p} = \frac{dist_{m,p}}{max(|m|, |p|)}$$

If there exists more than one preferred term for a concept, we take the maximum. This feature indicates a negative relation between a candidate concept and a mention. The more distant a preferred name is from a mention, the less likely it is that the mention refers to this concept.

3.3.2 Global Features

In contrast to local features, global features involve more than one mention. From a disambiguation perspective these features define, which mentions are disambiguated jointly.

Shared lemma (p7, f12) The one sense per discourse assumption states that one mention string is used to refer to one sense, i.e. in our case to one concept, in one discourse (Gale et al., 1992). For each document we extract all mentions with the same lemma and the inverse distance in sentences between the two. The bigger the inverse distance is, the closer the two mentions are to each other and the more likely it is that they refer to the same concept.

Head match (p8,f6) The one concept per discourse assumption often applies to mentions that are in a substring relation and share the same syntactic head lemma. We extract all these pairs including the inverse distance between the respective mentions.

Acronyms (p8,f6) In texts, especially in news paper texts, acronyms are often introduced by the following pattern: full name (acronym). We extract all these mention pairs, whereas one mention is the full name and the other one the acronym.[7]

Cross-document n-gram feature (p9,f13) In contrast to the previous features this one is a cross-document feature. The assumption is that we work with a document collection. We extract all mention pairs with the same lemma but coming from two different documents. For each of these mentions we extract all n-grams that include the respective mention and that consist of nouns and adjectives. If the two mentions share at least one of these n-grams, we consider them as referring to the same concept and add as score the number of shared n-grams.

[5] We take its logarithm.

[6] We use the Lingpipe implementation (`http://alias-i.com/lingpipe/`).

[7] In our Wikipedia training data, acronyms are relatively rare. Hence it is difficult to learn a weight for the acronym feature. As it is similar to the head match feature, we use the same predicate and weight for the two features.

Predicates

Hidden predicates	
p1	$hasConcept(m, c)$
p2	$hasSameConcept(m, n)$
Predicates realizing Wikipedia Miner features	
p3	$hasPriorProbability(m, c, s)$
p4	$hasRelatedness(m, c, s)$
Additional predicates involving one mention and one entity	
p5	$hasContextSimilarity(m, c, s)$
p6	$hasStringDistance(m, c, s)$
Predicates involving two mentions (intradocument)	
p7	$isSubStringHeadMatch(m, n, s)$
p8	$haveSameLemma(m, n, s)$
Predicates involving two mentions (cross-document)	
p9	$shareNgram(m, n, s)$

Formulas

Hard constraints			
f1			$\forall m \in M : \lvert\{c \in C : hasConcept(m, c)\}\rvert \leq 1$
f2			$\forall m, n \in M : m \neq n \wedge hasSameConcept(m, n) \rightarrow hasSameConcept(n, m)$
f3			$\forall m, n, l \in M : m \neq n \wedge m \neq l \wedge n \neq l$ $\wedge hasSameConcept(m, n) \wedge hasSameConcept(n, l) \rightarrow hasSameConcept(m, l)$
f4			$\forall m, n \in M : m \neq n \wedge hasSameConcept(m, n) \wedge hasConcept(m, c)$ $\rightarrow hasConcept(n, c)$
f5			$\forall m, n \in M : m \neq n \wedge m \neq n \wedge hasConcept(m, c) \wedge hasConcept(n, c)$ $\rightarrow hasSameConcept(m, n)$
Formulas with learned weights			
f6	$+$	$(w \cdot s)$	$\forall m, n \in M\ \forall c \in C_m : m \neq n \wedge isSubStringHeadMatch(m, n, s)$ $\rightarrow hasConcept(m, c) \wedge hasConcept(n, c)$
f7	$+$	$(w \cdot s)$	$\forall m \in M\ \forall c \in C_m : hasPriorProbability(m, c, s) \rightarrow hasConcept(m, c)$
f8	$+$	$(w \cdot s)$	$\forall m \in M\ \forall c \in C_m : hasRelatedness(m, c, s) \rightarrow hasConcept(m, c)$
f9	$+$	$(w \cdot s)$	$\forall m \in M\ \forall c \in C_m : hasContextSimilarity(m, c, s) \rightarrow hasConcept(m, c)$
f10	$-$	$(w \cdot s)$	$\forall m \in M\ \forall c \in C_m : hasStringDistance(m, c, s) \rightarrow hasConcept(m, c)$
f11	$-$	(w)	$\forall m \in M\ \forall c \in C_m : hasRelatedness(m, c, s) \wedge s = 0 \rightarrow hasConcept(m, c)$
f12	$+$	$(w \cdot s)$	$\forall m, n \in M : m \neq n \wedge hasSameString(m, n, s) \rightarrow hasSameConcept(m, n)$
f13	$+$	$(w \cdot s)$	$\forall m, n \in M : m \neq n \wedge shareNgram(m, n, s) \rightarrow hasSameConcept(m, n)$

Table 1: Predicates and formulas used for disambiguation and clustering (m, n, l represent mentions, M sets of mentions, c concepts and entities, C sets of concepts and entities, and s scores)

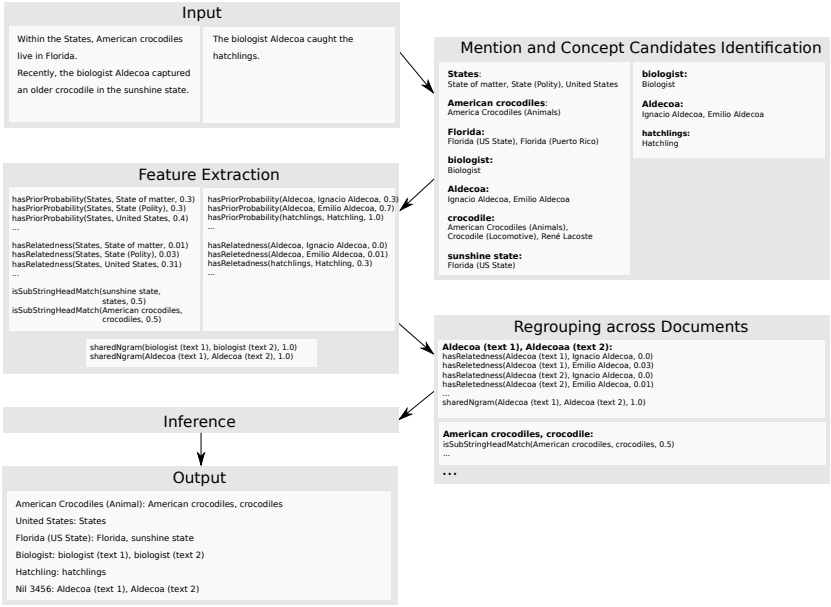


Figure 3: Example for the whole process.

3.4 Illustration of Our Approach

Given several documents – in Figure 3 just two – we first detect mentions in these documents by identifying noun phrases. If a mention is in our lexicon (see Section 4), we obtain all candidate concepts from there. Otherwise we just keep the mention, if it is at most of a length of four tokens and starts and ends with a noun. We keep mentions with no candidate concepts for two reasons: (1) It can happen that during disambiguation and clustering a mention is linked to a concept even if it does not have a candidate concept. (2) As we also want to cluster mentions with no candidate concept, they need to be kept in the network. In the next step we extract all features, we use for disambiguation and clustering (see Section 3.2). As features can cross document boundaries, we then regroup the mentions into pseudo documents, which are given to the inference module. One pseudo document contains all mentions that are linked by global features. The output is shown at the bottom of Figure 3.

4 Experiments

This section describes experiments on two different data sets. All experiments, including the ones for the baseline systems such as *WikipediaMiner*, are based on the same English Wikipedia dump[8], the same lexicon, which includes all anchor texts that occur more than two times with a certain concept (Milne and Witten, 2008),[9] and the same preprocessing. This way we ensure

[8]We use the dump from January 4th, 2012.

[9]We use a query expansion technique and also consider redirects and article titles and retrieve the candidate for the closest anchor.

Dataset	No. of Documents	No. of Mentions	in KB	NILs	Ave. Amb.
WP Training	500	46,810	43,547	3,263	2.18
WP Dev	100	7,197	6,610	587	2.11
ACE 2005	597	29,300	27,184	2,116	5.72
TAC 2011	2162	2250	1124	1126	4.51

Table 2: Datasets: Statistics

that differences in the results are caused exclusively by algorithm and features.

4.1 Data Sets

Disambiguating with respect to Wikipedia has the advantage that training data can be derived from the internal hyperlinks in Wikipedia automatically without manual annotation.

While training and development is done exclusively on Wikipedia (WP Training, WP Dev), we evaluate our approach and compare it to previous work using two data sets containing texts from different sources such as news papers, audio transcription records and the internet (ACE 2005, TAC 2011). Table 2 summarizes some statistics for each data set, namely the number of documents and mentions to disambiguate, the number of mentions with a corresponding concept in the knowledge base (in KB), the number of NILs and the average ambiguity.

4.1.1 Training and Development Data

For training and development we use featured articles from the English Wikipedia data (featured articles are supposed to be of high quality). We randomly select articles among those articles and consider all internal hyperlinks that point to an existing article as an concept-annotated mention. In Wikipedia only the first occurrence of a concept in an article is linked to the respective page. Since our aim is to disambiguate all occurrences we re-wikify the articles: all mentions, that – according to our lexicon – can refer to a linked article, are linked automatically to the respective concept. For training we collect for each annotated mention all candidate concepts from our lexicon. For obtaining NILs we randomly remove some of the concepts from the annotations and the lexicon. The development data set is processed in the same way.

4.1.2 Testing Data

The English part of the *ACE 2005* data set has been manually annotated with links to Wikipedia by Bentivogli et al. (2010). ACE 2005 consists of 597 texts from newswire reports, broadcast news, internet sources and transcribed audio data. Both common and proper nouns that are part of a coreference chain are annotated with one or more links to the English Wikipedia or as NILs. Some of the mentions are annotated with more than one link. We consider a mention as correctly disambiguated, if one of the annotated links is identified.

The English TAC dataset from 2011 consists of 2,250 queries and focuses on named entities such as persons, organizations and locations. A query consists of a query term, i.e. a name for a named entity, and a document, in which the query term appears. The documents are newspaper and web texts. Given our approach we disambiguate the whole document and not just the query terms. In contrast to ACE 2005 the NILs are not just annotated as NILs but also clustered, which allows us to evaluate the entity clustering performance in a direct way and not just its influence on the disambiguation performance as on the ACE data.

4.2 Experiments

4.2.1 Baselines, Other Systems, Upper Bounds

To evaluate our approach we compare it to different baseline systems. For all systems and baselines we cluster the remaining unclustered NILs in a postprocessing step using a string match heuristic, which is a hard-to-beat baseline for this task (Ji et al., 2011). For the upper bound we assume that the clustering is perfect given the disambiguation results.

Upper bound I (UB): The first upper bound shows the performance which can maximally be reached using our lexicon. Each mention is considered as correctly disambiguated, if the correct concept is among the candidates for that mention given our lexicon. If a mention is a NIL according to our gold standard, we also consider it as correct.

Upper bound II (UBD): The second upper bound considers all candidate concepts for all mentions in a document as candidate concepts for a mention. If the correct concept is among these candidates, it is considered as correct. This is the upper bound for our final ML system.

First concept baseline (First concept): In WSD the first sense baseline is known as hard to beat as the distribution of the concepts given a mention obeys Zipf's law. The first concept baseline only makes use of the prior probability of a concept given a mention and always selects the one with the highest prior probability.

WikipediaMiner: WikipediaMiner (Milne and Witten, 2008) is a state-of-the-art system which is freely available. We use version 1.1, extract all the necessary information from the 2012 Wikipedia dump and train it on the same training data that we use to learn the weights for our ML systems.

SVM Rank I (SR I): A common approach for named entity disambiguation is to use a ranker to rank the candidate concepts for each mention and then select the highest ranked concept for each mention. We use SVM^{Rank} (Joachims, 2002) trained on our Wikipedia training data using the same features as for *MLN Dis.* (see below).

SVM Rank II (SR II): This system also uses SVM^{Rank} (Joachims, 2002) and the same features as for *ML Dis.+NILs* (see below).

SVM Rank NIL Classifier II (SRC II): This system uses the *SVM Rank II* system to obtain for each mention the highest ranked concept. Then we apply a classifier to decide for each mention-concept pair, if it is a valid mapping or if the mention is a NIL. We use decision trees as a classifier (Witten and Frank, 2005) and the same features as for *ML Dis.+NILs*.

Other systems: For the TAC 2011 data we also add the best (**Best system**) (Monahan et al., 2011) and median (**Median system**) performance of all participating systems in the English entity linking task at TAC 2011.

4.2.2 Our Systems

ML Dis.: ML system using predicates p1, p3, p4, the local formulas f7, f8 and the global constraint f1. This system uses only information that is also used by WikipediaMiner and do not recognize NILs as no negative information is integrated.

MLN Dis.+NILs: ML system using predicates p1, p3-p6, the local formulas f7-f11 and the global constraint f1. While *ML Dis.* assigns each mention a concept, this system performs concept disambiguation and recognition of NILs jointly.

ACE 2005							
	Non NILs			NILs			
	P	R	F	P	R	F	Acc
UB	90.1	87.3	88.7	71.4	100.0	83.3	88.2
UBD	95.8	93.3	94.5	75.0	100.0	85.7	93.8
First Concept	68.3	69.4	68.8	49.9	39.5	44.1	67.2
WikipediaMiner	86.5	59.1	70.2	16.9	85.8	28.3	61.0
SR I	69.1	70.2	69.7	49.9	39.5	44.1	68.0
SR II	69.7	70.8	70.2	49.9	39.5	44.1	68.5
SRC II	79.7	57.8	67.0	19.0	86.0	31.1	59.8
ML Dis.	71.2	72.4	71.8	49.9	39.5	44.1	70.0
ML Dis.+NILs	79.0	74.2	76.5	36.0	63.9	46.1	73.5
ML Dis.+NILs+Clust.	78.0	75.6	76.7	41.1	57.4	47.9	74.3

Table 3: Evaluation on ACE 2005 data

ML Dis.+NILs+Clust.: Joint ML system that performs disambiguation, recognition of NILs and clustering jointly. While we use all predicates and formulas for the TAC data, we do not consider predicate p9, formula f13 on the ACE data.

4.2.3 Results

Table 3 and Table 4 show the results on the *ACE 2005* and *English TAC 2011* data set respectively. We report precision (P), recall (R) and F-measure (F) for the non NILs and the NILs as well as the overall accuracy (Acc), also known as micro-average. In addition we also present the B^3 precision, recall and F-measure for the *TAC data sets* in order to evaluate the entity clustering performance.[10] Significance is calculated for the overall accuracy using a paired t-test.

4.2.4 Discussion of the Results

The full system, which does joint disambiguation, recognition of NILs and clustering (*ML Dis.+NILs+Clust.*), significantly outperforms the other systems in the two tables – except the best performing system at TAC 2011 (*Best system*) – with $p < 0.01$. Bryl et al. (2010) report on the ACE data an F-Measure of 71.5 for non-NILs, but these results are not comparable as they use gold mentions instead of system mentions and consider a mention as correctly disambiguated only if it links to the first mentioned Wikipedia article in the gold standard. Ratinov et al. (2011) – another state-of-the-art system – report an accuracy of 78.8 and a B^3 F-Measure of 76.2 on the TAC 2011 data set (Ratinov and Roth, 2011).

The system *ML Dis.* is close to *WikipediaMiner*. It uses only positive evidence, i.e. the two described *WikipediaMiner* features, and links each mention to a concept in Wikipedia. No NILs are identified. Hence the results for the NILs on the ACE data set are the same as for the first concept baseline (*First Concept*), the ranking systems (*SR I* and *SR II*), which also assign each mention a concept, and *ML Dis.*. On the TAC 2011 data set the results for the NILs differ between the *First Concept* baseline, the ranking systems (*SR I, SR II*) and *ML Dis.*. The difference comes from the fact that the TAC knowledge base is not identifical with the knowledge base we use for disambiguation. If the system links a mention to an entry in our knowledge base, which is not part of the TAC knowledge base, it is considered as NIL. The system *ML Dis.+NILs* performs disambiguation and recognition of NILs jointly. Compared to

[10] We use the offical TAC scoring scripts.

TAC EN Test 2011										
	Non NILs			NILs						
	P	**R**	**F**	**P**	**R**	**F**	**Acc**	B^3 **P**	B^3 **R**	B^3 **F**
UB	100.0	75.0	85.7	80.0	100.0	88.9	87.5	87.5	87.2	87.4
UBD	100.0	95.2	97.5	95.4	100.0	97.7	97.6	97.6	97.4	97.5
First Concept	61.8	54.2	57.7	76.5	85.9	80.9	70.0	65.4	69.6	67.4
WikipediaMiner	86.1	55.1	67.2	70.0	95.2	80.7	75.2	70.7	73.7	72.2
SR I	72.8	66.5	69.5	81.5	88.5	84.8	77.5	73.7	76.4	75.0
SR II	73.2	66.9	69.9	81.2	88.2	84.5	77.6	73.7	76.5	75.1
SRC II	87.7	59.1	70.6	72.3	95.8	82.4	77.5	73.3	74.6	73.9
Best System										84.6
Median System										71.6
ML Dis.	71.4	65.5	68.3	81.4	88.1	84.6	76.8	72.9	75.7	74.3
ML Dis.+NILs	79.5	64.1	70.9	77.5	92.5	84.3	78.3	74.2	76.6	75.4
ML Dis.+NILs+Clust.	80.3	74.5	77.3	85.1	91.3	88.1	82.9	79.2	81.1	80.1

Table 4: Evaluation on TAC 2011

the two step process (*SRC II*), which uses the same features, but performs ranking and the classification of NILs in a cascaded way, the performance of the joint approach (*ML Dis.+NILs*) is higher. As the differences between the systems *ML Dis.+NILs* and *ML Dis.+NILs+Clust.* show, addressing clustering and disambiguation jointly improves the results even further. The improvement mainly comes from two different cases: (1) Mentions with no candidate concepts, which are recognized as NILs in *ML Dis.+NILs* are correctly disambiguated and clustered by *ML Dis.+NILs+Clust*. While for example *ML Dis.+NILs* recognized *Marinello* in "We pretty much know that Marinello, while on the board, has arranged to get future money" as a NIL, *ML Dis.+NILs+Clust.* links it to the correct entry in the knowledge base by also taking into account other occurrences of *Marinello* such as "because the fact that Randy Bauer is already talking about *Beatriz Marinello*". (2) Wrongly disambiguated mentions are – thanks to discourse knowledge – correctly disambiguated. This especially applies to occurrences of common nouns such as *region* or *friends*. Whereas the system *ML Dis.+NILs* wrongly disambiguated *friends* as the TV series, *ML Dis.+NILs+Clust.* correctly links it to the entry on friendship by taking into account other occurrences of *friends* in the text.

5 Conclusions

This paper presents a new approach for joint disambiguation, NIL recognition and clustering using Markov Logic. Our approach significantly outperforms all baseline systems and shows state-of-the-art performance. To our knowledge this is the first approach for joint disambiguation and clustering of concepts and entities. At the moment, we tested on a relatively small data set. For future work, we will work on scalability and on more linguistically informed features.

Acknowledgments. We would like to thank Mathias Niepert for his advice on Markov Logic and *thebeast*. This work has been partially funded by the European Commission through the CoSyne project FP7-ICT-4-248531 and the Klaus Tschira Foundation.

References

Agirre, E. and Edmonds, P. G., editors (2006). *Word Sense Disambiguation: Algorithms and Applications*. Springer, Heidelberg, Germany.

Bentivogli, L., Forner, P., Giuliano, C., Marchetti, A., Pianta, E., and Tymoshenko, K. (2010). Extending English ACE 2005 corpus annotation with ground-truth links to Wikipedia. In *Proceedings of the 2nd Workshop on The People's Web: Colloboratively Constructed Semantic Resources,* Beijing, China, 28 August 2010, pages 19–27.

Bizer, C., Lehmann, J., Kobilarov, G., Auer, S., Becker, C., Cyganiak, R., and Hellmann, S. (2009). DBPedia – A crystallization point for the Web of data. *Journal of Web Semantics*, 7:154–165.

Bryl, V., Giuliano, C., Serafini, L., and Tymoshenko, K. (2010). Supporting natural language processing with background knowledge: Coreference resolution case. In *Proceedings of the 9th International Semantic Web Conference, Revised Selected Papers, Part I,* Shanghai, China, 7-11 November 2010, pages 80–95.

Bunescu, R. and Paşca, M. (2006). Using encyclopedic knowledge for named entity disambiguation. In *Proceedings of the 11th Conference of the European Chapter of the Association for Computational Linguistics,* Trento, Italy, 3–7 April 2006, pages 9–16.

Chen, L., Feng, Y., Zou, L., and Zhao, D. (2012). Explore person specific evidence in web person name disambiguation. In *Proceedings of the 2012 Conference on Empirical Methods in Natural Language Processing and Natural Language Learning,* Jeju Island, Korea, 12–14 July 2012, pages 832–842.

Csomai, A. and Mihalcea, R. (2008). Linking documents to encyclopedic knowledge. *IEEE Intelligent Systems*, 23(5):34–41.

Dai, H.-J., Tsai, R. T.-H., and Hsu, W.-L. (2011). Entity disambiguation using a Markov-Logic network. In *Proceedings of the 5th International Joint Conference on Natural Language Processing,* Chiang Mai, Thailand, 8–13 November 2011, pages 846–855.

Domingos, P. and Lowd, D. (2009). *Markov Logic: An Interface Layer for Artificial Intelligence*. Morgan Claypool Publishers.

Dredze, M., McNamee, P., Rao, D., Gerber, A., and Finin, T. (2010). Entity disambiguation for knowledge base population. In *Proceedings of the 23rd International Conference on Computational Linguistics,* Beijing, China, 23–27 August 2010, pages 277–285.

Fahrni, A., Nastase, V., and Strube, M. (2011). HITS' graph-based system at the NTCIR-9 cross-lingual link discovery task. In *Proceedings of the 9th NTCIR Workshop Meeting,* Tokyo, Japan, 6-9 December 2011, pages 473–480.

Gale, W. A., Church, K. W., and Yarowsky, D. (1992). One sense per discourse. In *Proceedings of the DARPA Speech and Natural Language Workshop*.

Han, X. and Sun, L. (2012). An entity-topic model for entity linking. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics,* Jeju Island, Korea, 8–14 July 2012, pages 105–115.

Ji, H. and Grishman, R. (2011). Knowledge base population: Successful approaches and challenges. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics,* Portland, Oreg., USA, 19–24 June 2011, pages 1148–1158.

Ji, H., Grishman, R., and Dang, H. (2011). Overview of the TAC 2011 knowledge base population track. In *Proceedings of the Text Analysis Conference,* National Institute of Standards and Technology, Gaithersburg, Maryland, USA, 14–15 November 2011.

Ji, H., Grishman, R., Dang, H., Griffitt, K., and Ellis, J. (2010). Overview of the TAC 2010 knowledge base population track. In *Proceedings of the Text Analysis Conference,* National Institute of Standards and Technology, Gaithersburg, Maryland, USA, 15–16 November 2010.

Joachims, T. (2002). Optimizing search engines using clickthrough data. In *Proceedings of the 8th International Conference on Knowledge Discovery and Data Mining,* Edmonton, Canada, 23–26 July 2002, pages 133–142.

Kulkarni, S., Singh, A., Ramakrishnan, G., and Chakrabarti, S. (2009). Collective annotation of Wikipedia entities in web text. In *Proceedings of the 15th ACM SIGKDD Conference on Knowledge Discovery and Data Mining,* Paris, France, 28 June – 1 July 2009, pages 457–466.

Lin, T. and Etzioni, O. (2012). Entity linking at web scale. In *Proceedings of the Joint Workshop on Automatic Knowledge Base Construction and Web-scale Knowledge Extraction (AKBC-WEKEX),* Montréal, Québec, Canada, 7–8 June 2012, pages 84–88.

McNamee, P. (2010). HLTCOE efforts in entity linking at TAC 2010. In *Proceedings of the Text Analysis Conference,* National Institute of Standards and Technology, Gaithersburg, Maryland, USA, 15–16 November 2010.

McNamee, P. and Dang, H. T. (2009). Overview of the TAC 2009 knowledge base population track. In *Proceedings of the Text Analysis Conference,* National Institute of Standards and Technology, Gaithersburg, Maryland, USA, 16–18 November 2009.

Milne, D. and Witten, I. H. (2008). Learning to link with Wikipedia. In *Proceedings of the ACM 17th Conference on Information and Knowledge Management (CIKM 2008),* Napa Valley, Cal., USA, 26–30 October 2008, pages 1046–1055.

Monahan, S., Lehmann, J., Nyberg, T., Plymale, J., and Jung, A. (2011). Cross-lingual cross-document coreference with entity linking. In *Proceedings of the Text Analysis Conference,* National Institute of Standards and Technology, Gaithersburg, Maryland, USA, 14–15 November 2011.

Nastase, V. and Strube, M. (2012). Transforming Wikipedia into a large scale multilingual concept network. *Artificial Intelligence*.

Navigli, R. (2009). Word sense disambiguation: A survey. *ACM Computing Surveys*, 41(2):1–69.

Nothman, J., Honnibal, M., Hachey, B., and Curran, J. R. (2012). Event linking: Grounding event reference in a news archive. In *Proceedings of the ACL 2012 Conference Short Papers,* Jeju Island, Korea, USA, 8–14 July 2012, pages 228–232.

Pedersen, T. (2006). Unsupervised corpus-based methods for WSD. In Agirre, E. and Edmonds, P., editors, *Word Sense Disambiguation: Algorithms and Applications*, pages 133–166. Springer, Heidelberg, Germany.

Rao, D., McNamee, P., and Dredze, M. (2010). Streaming cross document entity coreference resolution. In *Proceedings of Coling 2010: Poster Volume,* Beijing, China, 23–27 August 2010, pages 1050–1058.

Ratinov, L. and Roth, D. (2011). GLOW TAC-KBP2011 entity linking system. In *Proceedings of the Text Analysis Conference,* National Institute of Standards and Technology, Gaithersburg, Maryland, USA, 14–15 November 2011.

Ratinov, L., Roth, D., Downey, D., and Anderson, M. (2011). Local and global algorithms for disambiguation to Wikipedia. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics,* Portland, Oreg., USA, 19–24 June 2011, pages 1375–1384.

Riedel, S. (2008). Improving the accuracy and efficiency of MAP inference for Markov logic. In *Proceedings of the 24th Conference in Uncertainty in Artificial Intelligence,* Helsinki, Finland, 9–12 July 2008, pages 468–475.

Schütze, H. (1998). Automatic word sense discrimination. *Computational Linguistics*, 24(1):97–123.

Sil, A., Cronin, E., Nie, P., Yang, Y., Popescu, A.-M., and Yates, A. (2012). Linking named entities to any database. In *Proceedings of the 2012 Conference on Empirical Methods in Natural Language Processing and Natural Language Learning,* Jeju Island, Korea, 12–14 July 2012, pages 116–127.

Singh, S., Subramanya, A., Pereira, F., and McCallum, A. (2011). Large-scale cross-document coreference using distributed inference and hierarchical models. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics,* Portland, Oreg., USA, 19–24 June 2011, pages 793–803.

Suchanek, F., Kasneci, G., and Weikum, G. (2008). YAGO: A large ontology from Wikipedia and WordNet. *Elsevier Journal of Web Semantics*, 6(3):203–217.

Tang, L.-X., Geva, S., Trotman, A., Xu, Y., and Itakura, K. (2011). Overview of the NTCIR-9 crosslink task: Cross-lingual link discovery. In *Proceedings of the 9th NTCIR Workshop Meeting,* Tokyo, Japan, 6-9 December 2011, pages 437–463.

Wick, M., Singh, S., and McCallum, A. (2012). A discriminative hierarchical model for fast coreference at large scale. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics,* Jeju Island, Korea, 8–14 July 2012, pages 379–388.

Witten, I. H. and Frank, E. (2005). *Data Mining: Practical Machine Learning Tools and Techniques*. Morgan Kaufmann, San Francisco, Cal., 2nd edition.

Zhou, Y., Nie, L., Rouhani-Kalleh, O., Vasile, F., and Gaffney, S. (2010). Resolving surface forms to Wikipedia topics. In *Proceedings of the 23rd International Conference on Computational Linguistics,* Beijing, China, 23–27 August 2010, pages 1335–1343.

Flexible Structural Analysis of Near-Meet-Semilattices for Typed Unification-based Grammar Design

Rouzbeh FARAHMAND *Gerald PENN*
University of Toronto
{rouzbeh,gpenn}@cs.toronto.edu

Abstract

We present a new method for directly working with typed unification grammars in which type unification is not well-defined. This is often the case, as large-scale HPSG grammars now usually have type systems for which many pairs do not have least upper bounds. Our method yields a unification algorithm that compiles quickly and yet is nearly as fast during parsing as one that requires least upper bounds. The method also provides a natural naming convention for unification results in cases where no user-defined type exists.

Keywords: HPSG, typed feature structures, unification-based grammar.

Proceedings of COLING 2012: Technical Papers, pages 833–848,
COLING 2012, Mumbai, December 2012.

1 Introduction

In Head-driven Phrase Structure Grammars (HPSG, Pollard and Sag, 1994), most of the on-line computation time is spent performing unifications. HPSG type systems are typically large and involve subtyping, so computing a unification involves computing a *join*, the least informative type that is subsumed by both of the argument feature structures' types.[1] Early TFS-based parsing systems like ALE (Carpenter and Penn, 1996) required that every pair of unifiable types have a least common supertype, i.e., joins are uniquely defined wherever they need to be defined. Type hierarchies that have this requirement are called *meet semi-lattices* (MSLs) (Davey and Priestley, 2002), because the existence of a meet for every pair of types is sufficient to guarantee the existence of a join for every unifiable pair of types when the type hierarchy is finite. Later HPSG parsing systems such as the LKB (Copestake and Flickinger, 2000) and PET (Callmeier, 2000) eliminated this restriction, and it is now extremely rare to find a decent-sized HPSG with an MSL type hierarchy. The LKB, at least in some early versions, used an on-line caching algorithm to add joins to the user-defined type system when necessary. PET adds all of the necessary joins in advance, during a grammar compilation stage. In either case, the time it costs to perform these repairs is proportional to the number of new joins that must be added. It therefore makes some sense to strive to add the least number of types necessary.

While modern HPSG type systems are never MSLs in practice, they are always *almost* MSLs, in a sense that can be made mathematically precise, which we do here for the first time. Converting an HPSG type system to an MSL would be horrendously slow if it were not the case that HPSGs were almost MSLs already, because an exponential number of types need to be added in the worst case. There is another cost hidden in this method, furthermore, at least when we strive to add the least number of extra types necessary. In this case, *naming* these types is also a challenge. Figure 1(a), for example, shows an input type hierarchy that is not an MSL. In Figure 1(b), it has been converted into an MSL by adding one extra type. Ideally, we would like to name this type in a manner that reflects its usage in the grammar. Two possibilities that are immediately evident are $e \vee f$ or $a \wedge b \wedge c \wedge d$. Neither of these are ideal because they do not attest to a precise pair of types that the system needed to unify during parsing, and which therefore gave rise to the need for this extra type. This is extremely useful for debugging understanding feature structure outputs with large grammars. Grammar writers instantly know what user-defined types had been combined.

The key to the alternative introduced here is that it actually costs *less* time in practice to compute with a *larger* type system provided that joins are added on-line when necessary. This is achieved by doing a small amount of extra compilation off-line — small in practice again because of a specific mathematical property that practical HPSGs possess - and using the data structure that it creates to add many more types during parsing than the absolute minimum necessary for MSL-hood. In the limit, our method creates a type hierarchy more like a *conjunctive lattice,* but it only adds types as it needs them. Figure 1(c) is what it creates from Figure 1(a). From the standpoint of naming conventions, this type system is ideal, because every added type is named for the set (pair or larger) of types that required it.

[1] Some of the HPSG community draw these subtypes below their supertypes in graphical representations of type hierarchies, but still use the name *join*, even though the correct term is *meet* in this orientation. We will also use the term, *join*, but will depict type hierarchies in the opposite orientation, in which more specific subtypes appear above their more general supertypes.

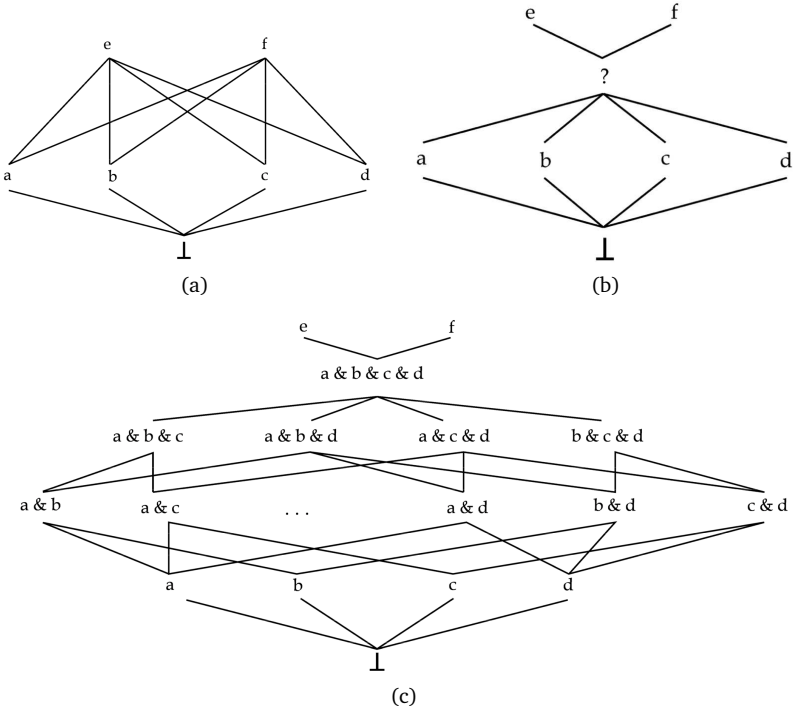


Figure 1: (a) A non-MSL type hierarchy, (b) its MSL completion based on the Dedekind-MacNeille completion, and (c) its conjunctive lattice.

Section 2 discusses the mathematical background necessary to understand the conversion that yields Figure 1(b). Section 3 presents a precise way of appreciating what we mean by a "near-meet-semilattice," through quantifying the size and number of a special kind of subset of types called *prime sets*. Section 4 then introduces another special kind of subset that can be used to efficiently compute the prime sets of a type hierarchy. Section 4.1 describes how joins are computed with the new method, and what the compiler computes to achieve this. Section 6 then presents an evaluation which shows that the new method has nearly the same performance at run-time (5.4% slower) as parsing with precompiled MSLs directly, but without the compile-time cost of the latter. We assert that in the context of grammar design, 5% is worth less compilation time along with the better naming conventions that naturally ensue from this method. Irrespective of compiling and parsing speed, the formal treatment of this subject also provides a more refined theory for analyzing the structure of type hierarchies in large-scale grammars than a binary MSL/non-MSL distinction.

2 Dedekind-MacNeille Completions

Let P be a partially ordered set, and $S \subseteq P$. The set of upper bounds of S, written S^u, is the set of all $x \in P$ such that for all $s \in S, s \leq_P x$. The lower bounds, S^l, are all of the $x \in P$ such that for all $s \in S, x \leq_P s$. The *Dedekind-MacNeille completion* of P ($DM(P)$) is the set of all $A \subseteq P$ such that $A^{ul} = A$. All of the singleton subsets of P are included in $DM(P)$, so informally we can say that P is included in $DM(P)$. $DM(P)$ is the smallest set that both includes P and is an MSL (Davey and Priestley, 2002).

The computer science literature on lattice theory has tended to emphasize $DM(P)$ as either a theoretical device for understanding (but not computing) joins even when they are not present (e.g., Aït-Kaci et al., 1989), or a goal to aim for when adding joins incrementally, each as it becomes necessary (e.g., Bertet et al., 1997). Using $DM(P)$ is supposed to provide us with some sort of reassurance about the maximum amount of extra work that must be performed to embed P into an MSL, at least if the results are pre-compiled or cached. Because $|DM(P)|$ is exponential in $|P|$ in the worst case, however, this does not provide much solace by itself. What matters in the end is the actual amount of overhead accrued for working with a non-MSL type hierarchy. This is true even if $DM(P)$ is not used.

That overhead can be paid either all at once during an initial compilation stage, or in small amounts over time, with the hope that some completion types will never be needed by the user's queries. Obviously, we do not want to compute any completion larger than $DM(P)$ at compile-time if we can help it. The prospect of incrementally computing completion types in a larger lattice is still available, however, as long as the overhead is still acceptable in practice. It is also worth noting that the incremental algorithm of Bertet et al. (1997) includes a line which stops execution for every type that is added and prompts the user to name it. The problem of naming in the Dedekind-MacNeille completion extends well beyond applications to computational linguistics.

3 Prime Sets

What remains then is to find a new source of reassurance about on-line cost that depends upon structural properties of the input type hierarchy rather than upon the size of the worst-case lattice using a given completion strategy. The best structural property of all is for P to be an MSL already. In this case, $DM(P)$ consists only of singleton sets plus possibly a new top-most element (which is usually discarded anyway), and so $|DM(P)| \leq |P| + 1$.

Paradoxically, even if P were not an MSL, finding the upper bound of a set of arguments $S \subseteq P$ in many cases is as easy as combining every type of S in succession, according to some arbitrary linear ordering of its elements. At each step, we compute the join of the current element and a running accumulator type, or fail if they have no upper bounds in common. That is still linear time in $|S|$, even if non-linear in other variables, as is the case when P is an MSL. It is difficult to imagine doing any better in this particular dimension.

The trouble is that this linear-time incremental method does not always work, because of subsets of S (or P) that we will call *prime sets*.[2].

Definition 1. *Let P be a partially ordered set, and $S \subseteq P$. S is an anti-chain iff for all $x, y \in S$, neither $x \leq_P y$ nor $y \leq_P x$.*

[2]These should not be confused with prime ideals or filters nor the prime elements that generate them (see section I-3 of Gierz et al., 2003)

Definition 2. *Let P be a partially ordered set, and $S \subseteq P$. S is a prime set of P iff $|S| > 1$, S is an anti-chain, and, for all non-empty $T \subseteq S$, T has a join iff $|T| = |S|$ or $|T| = 1$.*
Proposition 3. *A partial order is a meet-semi-lattice iff its maximal prime sets are of size 2.*

No non-trivial proper subset of a prime set has a join. When $|S| > 2$ and S is prime, considering pairs of elements in succession will not work for any linear ordering of S. In Figure 2(a), for example, $S = \{a, b, c\}$ is a prime set because it has a join and none of its 2-subsets does. One needs to consider all three at once to see the join.

Candidate sets S of size larger than 2 are important because they naturally result from delaying the computation of joins. A very natural strategy for performing unification in a non-MSL is to use joins where they exist, and the union of the sets of types to be unified otherwise, hoping that the latter will eventually be resolved to a join by the addition of some other element later in the computation. This strategy implicitly uses the conjunctive lattice of P to support its computations, but does not compute it. It is also the strategy that we will adopt. The only thing we need to remember is that we should never refer to a set or subset of types when that set has a join in P that we could refer to instead. That amounts to using the user's names for conjunctions vis-a-vis the subtyping relation where they exist, and explicit conjunctions elsewhere. Operationally, it reduces unification to searching for prime sets and prime subsets.

In a nutshell, the reason that $DM(P)$ can be exponentially larger than P is that the prime sets of P can grow in size linearly with $|P|$, as will be proven in the next section. Better still, tabulating the number of prime sets of each size indicates how "MSL-like" a type hierarchy is. It gives us a spectral view of the basic subsets that every large set decomposes into during unification.

While most large HPSGs are not MSLs in practice, they do not avail themselves of the possibility of having large prime sets either. This is evident in their spectra, and it means that it is relatively inexpensive to enumerate all of the prime sets.

4 Pseudo-prime Sets

A close variant of prime sets turns out to be even more useful:
Definition 4. *$S \subseteq P$ is a pseudo-prime set of P iff $|S| > 1$, S is consistent, and, for all non-empty $T \subseteq S$, T has a join iff $|T| = 1$.*

A pseudo-prime set is as close as a set with no least upper bound can get to being a prime set. The number of pseudo-prime sets is also an upper bound on the potential number of new types added by a completion, because every completion type in $DM(P)$ corresponds to the set of upper bounds of some pseudo-prime set (Penn, 2000, though stated in different terms).

In Figure 2(a), $\{a, b\}, \{a, c\}$ and $\{b, c\}$ are all pseudo-prime sets, for example.

Pseudo-prime sets stand in a very special relationship to prime sets and to each other:
Proposition 5. *Every k-subset of a pseudo-prime set with $k > 1$ is a pseudo-prime set.*
Proposition 6. *Every proper k-subset of a prime set with $k > 1$ is a pseudo-prime set.*

This means that we do not have to search through all $\binom{|P|}{k}$ possible k-subsets of P to find the size-k prime sets of P. Instead, we can merely focus on the size $k-1$ pseudo-primes,

```
Data: A finite poset of types P = ⟨T,⊑⟩
Result: All the pseudo-prime PsPrime(P) and prime sets Prime(P) of P
1  begin
2      init: Prime(P) ← {}, PsPrime(P) ← {} ;
3      forall the consistent types t1,t2 ∈ T do
4          Mins ← FindMinimal({t1,t2}^u);
5          if Mins has only one element then                    // There is a least upper bound
6              Add {t1,t2} to Prime(P) ;
7          else// There is no join
8              Add {t1,t2} to PsPrime(P) ;
9          end
10     end
11     if if there is no set in PsPrime(P) with size 2 then
12         return PsPrime(P) and Prime(P) ;
13     else// P is not an MSL: search for primes and pseudo-primes of size greater than 2
14         size := 2 ;
15         repeat
16             forall the pp1,pp2 ∈ PsPrime(P) do
17                 Cand ← pp1 ∪ pp2;
18                 if |Cand| is size + 1 then
19                     Mins ← FindMinimal(Cand);
20                     if Mins has only one element then
21                         Prime(P) += Cand;
22                     else
23                         PsPrime(P) += Cand;
24                     end
25                 end
26             end
27             size = size + 1;
28         until (there is no pseudo-prime set of size);
29         return PsPrime(P) and Prime(P) ;
30     end
31 end
```

Algorithm 1: Algorithm for finding prime and pseudo-prime sets of a poset.

one of which must live inside every prime set. The size $k-1$ pseudo-primes in turn can be constructed from $k-2$-pseudo-primes, etc., with the 2-pseudo-primes being the pairs of types that attest to the non-MSLhood of P. This recursive search procedure is given in Algorithm 1. Any procedure that enumerates prime sets or pseudo-prime sets is exponential in the worst case, but this algorithm makes efficient use of the relationship between primes and pseudo-primes to achieve efficiency when the largest prime sets are of low cardinality.

Pseudo-primes are also the key to bounding the size of prime sets:

Proposition 7. *The maximum size attainable by a prime subset of a poset P is $\lfloor \frac{|P|-1}{2} \rfloor$.*

Proof. Let Ψ be any prime subset of P. Let $n = |P|$ and $p = |\Psi|$. Consider the p distinct subsets of Ψ of size $p-1$ (i.e. $K_1, ..., K_p \subseteq \Psi$ such that $|K_1| = ... = |K_p| = p-1$). Let $m = |\bigcup_{1 \le i \le p} \mu(K_i)|$ where $\mu(S) = \{x \in S^u \mid \nexists\, u \in S^u \text{ s.t } u \le x\}$, the set of all minimal upper-bounds of S. For all i, $|\mu(K_i)| \ge 2$ because $K_1, ..., K_p$ are all pseudo-prime sets. For the same reason, $|\mu(K_i) \cap \mu(K_j)| \le 1$ when $i \ne j$, because then $K_i \cup K_j = \Psi$. In fact, $\mu(K_1), ..., \mu(K_p)$ are either all pairwise disjoint or all agree on the same one element, so there are only two cases:

Case A: $\bigcap_i \mu(K_i) = \emptyset$ Then P must have at least $p+m$ elements and one extra element for the join of Ψ that must exist above all the m elements in $\bigcup_{1 \le i \le p} \mu(K_i)$. So, (i) $n \ge p+m+1$. Conservatively, we must choose p disjoint antichains of size at least 2 from the m elements, thus we have (ii) $p \le \lfloor \frac{m}{2} \rfloor$. Therefore, from (i) and (ii), if m is even then $n \ge 2p+1$, and if m is odd then $n \ge 3p+2$. Since $2p+1 \le 3p+2 \le n$, then $p \le \frac{n-1}{2}$.

Case B: $\bigcap_i \mu(K_i) \neq \emptyset$. In this case, $n \geq p + m$ and $p \leq m - 1$, subtracting for the one upper bound that all of the K_i have in common. Therefore, $n \geq p + m \geq 2p + 1$ and consequently $p \leq \frac{n-1}{2}$.

□

There are also simple examples in which this bound can be attained, so this is tight.

A spectral decomposition of pseudo-primes is also somewhat interesting because of its relationship to the size of $DM(P)$, and to prime sets — the largest prime set cannot be more than one element larger than the largest pseudo-prime, but in practice it is much less. Figure 3 shows the decompositions for both primes and pseudo-primes for both a 1999 pre-release of the English Resource Grammar (ERG Flickinger, 1999) and Berligram (Müller, 2007). The first thing that we can observe is the very small size of all of the pseudo-primes and primes in both grammars — at most 9 in the ERG, and 6 in Berligram. It took 27 seconds [3] to find all the primes and pseudo-primes in the ERG, and less than 1 second to find them in Berligram. By contrast, it takes the LKB 4 seconds to compute $DM(P)$ for the ERG. We can also see a greater difference between the largest prime and the largest pseudo-prime in the ERG (9-4=5) than we can in Berligram (6-4=2). Pseudo-primes lay the groundwork for primes, so when a prime set nevertheless does not occur, this is significant. Finally, we can observe that, although the ERG is nearly 10 times the size of Berligram in its total number of types, it has about 25 times as many primes of cardinality 3 or greater, and of the same maximum size (4).

A large number of primes, a skewed distribution of primes towards small cardinalities, a large number of pseudo-primes, and a large difference between the size and/or number of pseudo-primes and primes are all strong indicators that a type hierarchy is more MSL-like when interpreted relative to the total number of types. The total number of types is not a good indicator, nor is "join density," which is common to formal concept analysis (Besson et al., 2005).[4] The (relative to P) size of $DM(P)$ is not bad, but it fails to indicate just how far the tails of these distributions extend. A type hierarchy with 10^6 pseudo-primes of size 2 is more MSL-like than one with 10^5 pseudo-primes of size 9, and simple counts of $|DM(P)|$ often cannot tell the difference, especially with high join densities.

One conclusion to draw from this particular comparison is obviously that the ERG is more MSL-like, which implies that it is more conservative in its structure, and would be easier to compute with than many other potential 3414-type hierarchies. On the other hand, the sheer number of types may make finding the right part of the type hierarchy to modify more difficult than it needed to have been. Another way to look at it is that Berligram makes more efficient use of the amount of information that a 434-type hierarchy can carry. There is more information embedded in its structure relative to its size, but that information may make it more difficult to predict the consequences of type unification than in some other hierarchies of its size. The ERG and Berligram do not have the same number of types, nor

[3] All the timing results reported in this paper were obtained on an Intel®-based system with a 3.6 GHz Xeon™ processor and 3 GB of RAM running the Ubuntu 6.06.2 Linux operating system.

[4] Join density, in terms of our partial orders, is calculated as the ratio of the size of the extension of the (transitively and reflexively closed) subtype relation to the square of $|P|$. The join density of the ERG is .001; that of Berligram is .006.

do they attempt to account for the same constructions, nor even the same language, so we can only speculate here. But prime sets and pseudo-primes can illuminate these issues.

4.1 Building an Automaton-based Index

The final application of prime and pseudo-prime sets considered here is to unification itself. As mentioned above, it is possible to maintain argument sets of types from a non-MSL as argument sets even after unification, provided that all of the prime subsets are replaced with their joins. For the unifier not to perform this replacement is tantamount to it simply handing back a candidate pair of types from an MSL ununified.

If we use topologically sorted lists (as induced by the subtype relation) as our representations of prime sets, pseudo-prime sets and argument sets, then Propositions 5 and 6 will allow us to use a simple automaton-based index to perform this replacement. The index has a linear number of states relative to the number and size of the pseudo-primes, because each state corresponds to a prefix of one or more pseudo-primes in topological order. It has two kinds of edges: *suffix edges* and *redex edges*, and both kinds of edges are labelled with elements of pseudo-prime or prime sets.

All of the pseudo-prime sets can be arranged into a tree such that every path through the tree from the root to any node corresponds to a pseudo-prime or a singleton set. Paths from the root to a leaf correspond to maximal pseudo-primes — no pseudo-prime contains them. These trees are connected with suffix edges, each of which is labelled with the k^{th} element that extends a $k-1$-length prefix of one or more pseudo-primes to a k-length prefix. On top of this skeleton, we add the redex edges, which map a pseudo-prime prefix, X, that contains a $k-1$-length prefix of a size-k prime set to the result of replacing that prime set within X by its join, transitively closed under all possible further replacements. The redex edge is labelled with the k^{th} element that completes the prime set.

The index for Figure 2(a), for example, is shown in Figure 2(b). Its pseudo-prime sets are: $\{\{a,b\}_{12}, \{a,c\}_{13}, \{a,z\}_{14}, \{b,z\}_{15}, \{b,c\}_{16}, \{d,z\}_{17}, \{c,z\}_{18}, \{g,z\}_{19}, \{f,z\}_{20}, \{e,z\}_{21}, \{h,z\}_{22}, \{a,b,z\}_{23}, \{a,c,z\}_{24}, \{b,c,z\}_{25}$; and its prime sets are: $\{a,f\}$, $\{b,g\}$, $\{d,c\}$, $\{d,g\}$, $\{d,f\}$, $\{d,e\}$, $\{g,f\}$, $\{g,e\}$, $\{f,e\}$, $\{a,b,c\}$.[5] Suppose $\{a,b,c,z\}$ is of interest. The automaton consumes a and b, which is a pseudo-prime (12) and the prefix of the pseudo-prime, $\{a,b,z\}_{23}$, but upon seeing c traverses a redex edge that reduces the prime set $\{a,b,c\}$ to its join, e, which is a singleton and therefore not subject to further replacement. Consuming z next leads to the pseudo-prime, $\{e,z\}_{21}$. So if the union of the argument sets provided to the unifier is $\{a,b,c,z\}$, what the unifier should return is $\{e,z\}$, which should be interpreted as the conjunctive type, $e\&z$. This is as close to the grammar writer's idioms as we can come in describing this result, given that the type hierarchy is not an MSL.

Having constructed the automaton directly from the pseudo-prime and prime sets, it can then be pruned by eliminating states that correspond to singletons with no outgoing suffix edges and no outgoing or incoming redex edges. Singleton sets clearly do not need to be fed to the automaton. In the example in Figure 2(b), states 9, 10 and 11 would be pruned.

Some statistics for the automata constructed for the ERG and Berligram are presented in

[5] These sets are sorted according to the following topological order (subscripts are topological ordinals): $\bot_0 < a_1 < b_2 < d_3 < c_4 < g_5 < f_6 < e_7 < h_8 < z_9 < y_{10} < x_{11}$.

Table 1. Our implementation of the automaton compiler has been written in Prolog and

	Before pruning		After pruning	
	ERG	Berligram	ERG	Berligram
# of suffix edges	9508	1425	7058	1205
# of redex edges	2296	554	2296	554
Total # of edges	11804	1979	9354	1759
Total # of states	8967	1314	6517	1094

Table 1: Statistics related to the constructed automaton-based index for type hierarchies of the ERG (Flickinger, 1999) and Berligram (Müller, 2007).

generates the automaton as Prolog clauses, too, which are then also compiled. The two stages of compilation together consumed 30 milliseconds for Berligram and 2.2 seconds for the ERG, which is well within the range of compilers that add joins on-line. Further speed improvements are doubtlessly possible, since the automaton can in principle be constructed incrementally as Algorithm 1 is being executed.

5 Unification

Having constructed the automaton directly from pseudo-prime and prime sets, we are implicitly assuming that every input set to the automaton is an anti-chain. The algorithm for unification, given as Algorithm 2, shows how the result should look.

Data: $P = \langle T, \sqsubseteq_P \rangle$;
two sets of types $A, B \subseteq T$;
the automaton-based index $\mathscr{M}_P$;
ψ: the topology of P used for construction of $\mathscr{M}_P$;
Result: minimal upper bound of A and B or failure

```
1 begin
2     Feed ← AntiChainReduce(P, ψ ,A∪B);
3     if Feed has only one element then
4         return Feed;
5     else
6         AutResult ← Aut(start state of M_P , Feed);
7         return AutResult;
8     end
9 end
```

Algorithm 2: The algorithm for unification of two sets of types.

`AntiChainReduce` reduces an input argument set to the topologically sorted, maximally specific anti-chain that contains it. This can be accomplished in quadratic time as a function of the size of the input set, simply by considering every pair of types and replacing the pair with the higher of the two if they are ordered.

6 Parsing Evaluation

To evaluate how this unification algorithm performs in practice, we evaluated three systems on a common grammar and corpus of sentences. The grammar was the ERG, and the corpus, called the FUSE corpus, has 2354 sentences, ranging from 1 to 50 words in length, 192 of which are ungrammatical according to the ERG. All three of the systems that we

tested had memory allocation problems while parsing this corpus, so every sentence that resulted in at least one system running out of memory was excluded, leaving a remainder of 1727 sentences, 159 of which were ungrammatical.[6] We also imposed a maximum of 8000 chart edges after which parsing was terminated. The excluded sentences were those that exceeded the memory available to one of the processes with that cap in place. The three systems were:

(1) ALE version 4.0 beta, running on the Dedekind-MacNeille completion of an ALE port of a 1999 pre-release version of the ERG, with 45 rules, 155 features, 1314 lexical entries, no lexical rules and 3412 types, plus a most general type, $\bot$, plus a built-in type for strings. $DM(P)$ added another 893 types to the system. We compiled ALE with SICStus Prolog 3.12.10 (compact code). [7]

(2) the LKB, version as at March, 2009: This system allows non-MSL grammars as input, but it automatically computes the Dedekind-MacNeille completion on their type systems at compile-time, naming newly added types with a number. The 1999 pre-release of the ERG no longer runs on the LKB because of non-backwards-compatible changes in the system over the last 10 years. Porting an LKB grammar to ALE is very tricky because the LKB's input syntax is very heavily overloaded. So instead of porting the current ERG to ALE, we reported our existing port of the 1999 pre-release of the ERG back to this version of the LKB. We have verified that these two ports generate exactly the same edges on the non-excluded sentences of this corpus. We compiled the LKB with Allegro Common Lisp Enterprise 8.0.

(3) an experimental system, obtained by replacing ALE 4.0's type unifier with the one described in this paper. This system generates the same object code as ALE 4.0 when the input grammar's type system is an MSL. We ran this on the 1999 pre-release of the ERG without computing the Dedekind-MacNeille completion, which thus results in different object code.

The results are shown in Figure 6. These are log-linear graphs, so the roughly even separations attest to a nearly constant factor of speed-up, not a constant difference in milliseconds. The LKB is approximately 1.82 times slower than ALE 4.0, and the experimental version implemented for this study is approximately 1.054 times slower than ALE 4.0. Thus the experimental system produces a run-time parser that is 5.4% slower than ALE after having compiled out the Dedekind-MacNeille completion in advance, but is still more than 40% faster than the LKB, a commonly used system that caches them on-line.

The figure of 5.4% obtains with an experimental system that does not cache the results of prime-set reduction on previously seen argument sets. Caching is best implemented by caching not only previously seen argument sets that are consistent, but also previously seen argument sets that are inconsistent. As shown in Figure 6(d), however, the difference is not large. In the latter case, one obtains a system that is an average of 5.7% slower than parsing with precompiled MSLs on a first pass of parsing, but only 2.7% slower on a second pass through the same corpus of sentences (for which all of the cache entries are in place). We used the built-in SICStus `term_hash/2` predicate to hash argument sets, which gave us a perfect hash (one set per hash). Here again, a comparison with the LKB is informative in that argument set caching behaves surprisingly like the LKB's lexicon caching: while we

[6] The test sentences and results are available at `http://www.cs.toronto.edu/~rouzbeh/resources.html`.
[7] The MSL-ERG in ALE syntax is available at `http://www.cs.toronto.edu/~rouzbeh/resources.html`.

might naturally expect to see a steady stream of new words throughout the parsing of a corpus of this size, we also see a steady stream of new argument sets, even though the number of types and their relative ordering are held constant throughout the experimental parsing runs. As a result, it would take a much larger sample to witness a convergence of the second pass with the first. This suggests that argument set caching is only worthwhile over much longer timespans of use with the same type system.

7 Conclusion

We introduced two new mathematical constructions, prime sets and pseudo-prime sets, and showed that they provide a reasonable alternative to the Dedekind-MacNeille completion, by providing a means for manipulating conjunctive sets of types at very low overhead. These sets provide better naming conventions for newly added types than any implementation based on Dedekind-MacNeille could hope to do because they effectively allow for a trace of partial unifications of a set of arguments. Prime and pseudo-prime sets also form the basis of a more refined method for analyzing the upper bounds of a type system than simply calling it an MSL or non-MSL. This is of independent value to grammar designers.

We have not yet looked at the frequency distributions of primes and pseudo-primes encountered during parsing with the ERG over the FUSE corpus, for example. Both our compilation strategy and the spectral decomposition method would benefit from taking this information into account, because some portions of a completed lattice are clearly more important than others, simply as a result of certain types being used more often in parsing representative corpus input than others.

Some additional work on the combinatorial properties of prime and pseudo-prime sets also remains. Although we have identified a tight upper bound on the possible sizes of prime sets, the only known bounds on the numeracy of prime and pseudo-prime sets are based on the classical problem (the so-called *Dedekind Problem*), of finding the number of anti-chains of a given poset. The field had settled on a fairly stable bound until recently (Korshunov and Shmulevich, 2000).

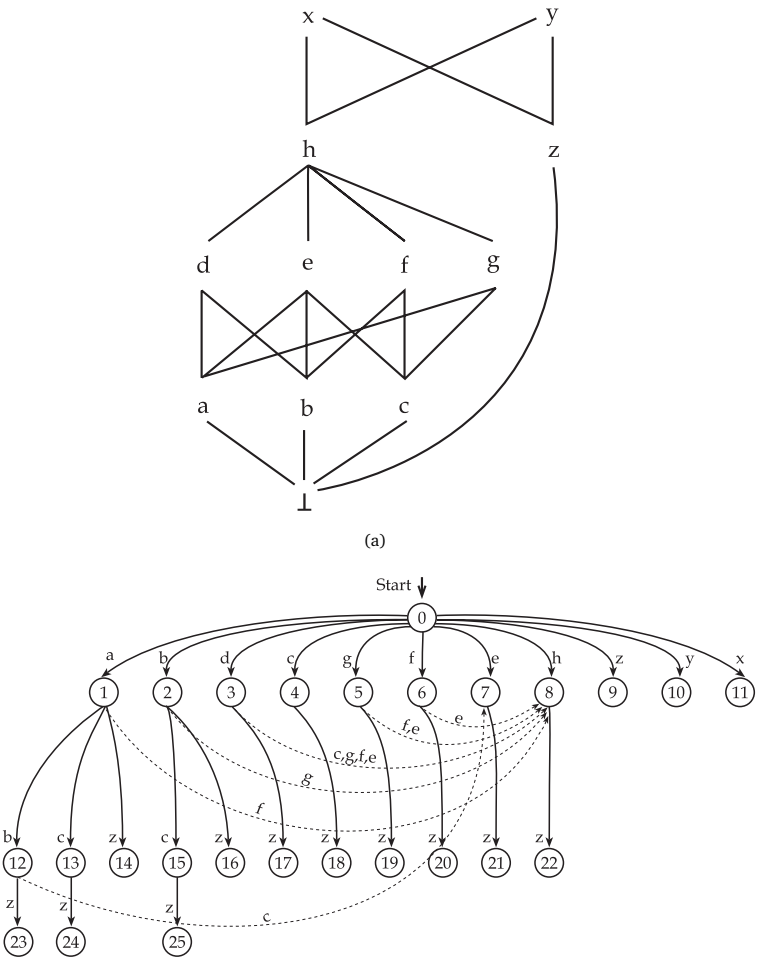


Figure 2: (a) A non-MSL type hierarchy and (b) its automaton-based index). The redex edges are depicted as dotted arcs; if those eliminated from the automaton, a suffix tree is obtained.

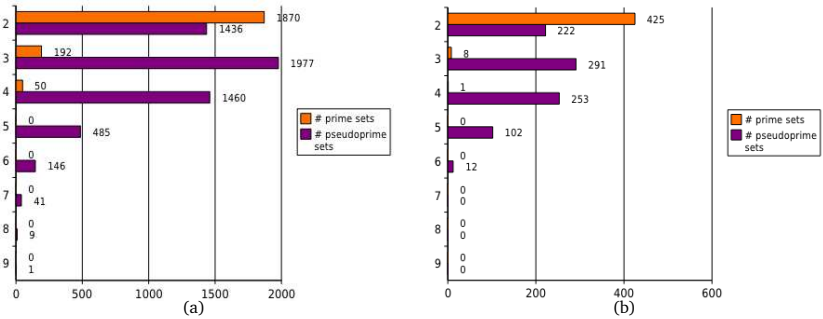


Figure 3: Number and size of the prime and pseudo-prime sets of (a) the English Resource Grammar (3412 types in total), and (b) the German Berligram (434 types). The horizontal and vertical axes show the number and size of the sets, respectively.

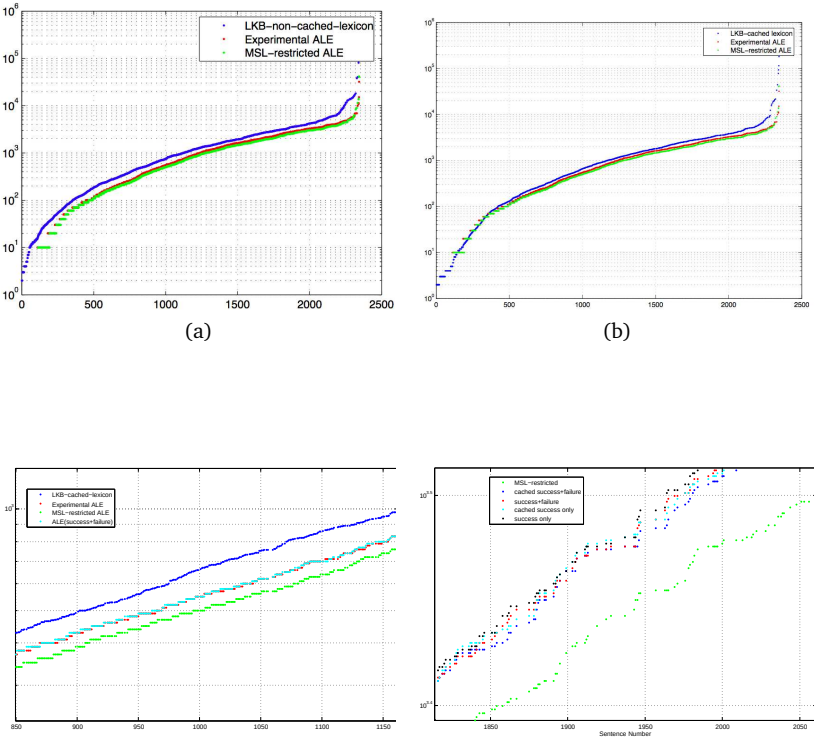


Figure 4: Evaluation of the `MSL-restricted ALE`, `Experimental ALE` and the LKB on FUSE. The LKB caches the lexicon as it parses, so each sentence was parsed twice in succession. The parsing time of the first parse is given in (a) and that of the second in (b). Figure (c) is a close-up of a portion of Figure (b), along with Experimental ALE plus caching of consistent and inconsistent argument sets (first pass). Figure (d) shows the effect of caching in close-up detail.

References

Aït-Kaci, H., Boyer, R., Lincoln, P., and Nasr, R. (1989). Efficient implementation of lattice operations. *ACM Trans. Program. Lang. Syst.*, 11(1):115–146.

Bertet, K., Morvan, M., and Nourine, L. (1997). Lazy completion of a partial order to the smallest lattice.

Besson, J., Robardet, C., Boulicaut, J.-F., and Rome, S. (2005). Constraint-based concept mining and its application to microarray data analysis. *Intell. Data Anal.*, 9(1):59–82.

Callmeier, U. (2000). Pet – a platform for experimentation with efficient HPSG processing techniques. *Nat. Lang. Eng.*, 6(1):99–107.

Carpenter, B. and Penn, G. (1996). Efficient parsing of compiled typed attribute value logic grammars. In Bunt, H. and Tomita, M., editors, *Recent Advances in Parsing Technology*, pages 145–168. Kluwer Academic Publishers, Dordrecht, Boston, London.

Copestake, A. and Flickinger, D. (2000). An open-source grammar development environment and broad-coverage English grammar using HPSG. In *In Proceedings of LREC 2000*.

Davey, B. and Priestley, H. (2002). *Introduction to Lattices and Order*. Cambridge University Press.

Flickinger, D. (1999). LinGo, the English Resource Grammar.

Gierz, G., Hofmann, K. H., Keimel, K., Lawson, J. D., Mislove, M., and Scott, D. S. (2003). *Continuous Lattices and Domains*, volume 93 of *Encyclopedia of Mathematics and its Applications*. Cambridge.

Korshunov, A. D. and Shmulevich, I. (2000). On the distribution of the number of monotone boolean functions relative to the number of lower units. *Discrete Math.*, 257(2-3):463–479.

Müller, S. (2007). Berligram: German grammar based on Head-driven Phrase Structure Grammar: Eine Einführung.

Penn, G. (2000). *The Algebraic Structure of Attributed Type Signatures*. PhD thesis, School of Computer Science, Carnegie Mellon University.

Pollard, C. and Sag, I. (1994). *Head-Driven Phrase Structure Grammar*. Chicago University Press, Chicago, Illinois.

Stacking of Dependency and Phrase Structure Parsers

Richárd Farkas and Bernd Bohnet
Institute for Natural Language Processing
University of Stuttgart
{farkas,bohnetbd}@ims.uni-stuttgart.de

ABSTRACT
We investigate the stacking of dependency and phrase structure parsers, i.e. we define features from the output of a phrase structure parser for a dependency parser and vice versa. Our features are based on the original form of the external parses and we also compare this approach to converting phrase structures to dependencies then applying standard stacking on the converted output. The proposed method provides high accuracy gains for both phrase structure and dependency parsing. With the features derived from the phrase structures, we achieved a gain of 0.89 percentage points over a state-of-the-art parser and reach 93.95 UAS, which is the highest reported accuracy score on dependency parsing of the Penn Treebank. The phrase structure parser obtains 91.72 F-score with the features derived from the dependency trees, and this is also competitive with the best reported PARSEVAL scores for the Penn Treebank.

KEYWORDS: parsing, stacking, dependency parsing, phrase structure parsing.

Proceedings of COLING 2012: Technical Papers, pages 849–866,
COLING 2012, Mumbai, December 2012.

1 Introduction

Both phrase structure and dependency parsers have developed considerably in the last decade, cf. (Nivre et al., 2004; McDonald et al., 2005b; Charniak and Johnson, 2005; Huang, 2008). The development has taken rather different directions as phrase structure parsers and dependency parsers employ different techniques to parse sentences. Phrase structure parsers usually apply probabilistic context free grammars and focus on the relationships among phrases. Dependency parsers use edge factored models for parsing that primarily model the interaction between the a head word and a dependent word. Second order graph-based parsers consider in addition for the decision on a dependency edge the interaction with siblings and grandchildren. Phrase structure and dependency parsers have both shown to be efficient and to provide accurate parsing results. Different parsing approaches have different strengths on distinct linguistic constructions, cf. (Nivre and McDonald, 2008).

In this paper, we exploit the difference of the representations of dependency and phrase structure parses and the divergence in the parsing techniques. We use features derived from the 1-best automatic phrase structure parse to augment the feature set of the dependency parser and vice versa. This way of combining parsers proved to be effective and provides a substantial accuracy gain for both parsing approaches.

Our ultimate objective is to advance one parser by a second one. This motivation is different from previous approaches for combining phrase-structure and dependency parsers (cf. (Klein and Manning, 2003; Carreras et al., 2008; Rush et al., 2010)) which aimed to achieve a joint optimum of the two approaches. Our proposal has three advantages over previous work. (i) Our approach is simple to implement by defining new feature templates, yet very effective. (ii) It does not require a joint representation of different parse structures. (iii) The participating dependency and phrase-structure parsers can be easily replaced, and each can be developed and optimized independently.

In our experiments both parsers utilize the same (training) data set – having two different representations – without having access to external information. Thus, our results indicate that state-of-the-art parsers can be still further improved without additional data and that there may be approaches that reach even higher accuracy levels in a single pass. At the dependency parser framework, we also investigated whether our approach retains its added value over other extensions and we answer the research questions: Is there a significant improvement still possible with two stacked dependency parsers? What is the impact on top of the results which utilize external information?

We also describe the findings of a manual contribution analysis of the second parser. This discussion identifies linguistic constructions which are responsible for the improvements and it brings up ideas how the parser approaches themselves can be improved.

The contributions of this paper are as follows:

- We propose the stacking of dependency and phrase-structure parsers.
- We report outstanding scores on the Penn Treebank and on the German Tiger Treebank for stacking a phrase structure parser on and a dependency parser and vice versa. The added value of the approach in the dependency parsing environment is still considerable in the presence of other extensions.
- We give explanations why the proposed stacking approach works.

2 Related Work

A number of studies have addressed feature-rich dependency and phrase structure parsing, cf. (Nivre et al., 2004; McDonald et al., 2005b; Charniak and Johnson, 2005; Huang, 2008).

The two main approaches to **dependency parsing** are transition-based dependency parsing (Yamada and Matsumoto, 2003; Nivre, 2003; Titov and Henderson, 2007), and graph-based dependency parsing (Eisner, 1996; McDonald et al., 2005a; Carreras, 2007; Rush et al., 2010).

The most successful supervised **phrase structure parsers** are feature-rich discriminative parsers which heavily depend on an underlying PCFG (Charniak and Johnson, 2005; Huang, 2008). These approaches consist of two stages. At the first stage they apply a PCFG to extract possible parses, then at the second stage select the best parse from the set of possible parses (i.e. rerank this set) employing a large feature set (Collins, 2000; Charniak and Johnson, 2005).

There is little related work on **combining the two approaches**. Some generative parsing approaches have exploited the differences between phrase structure and dependency parsers. For instance, Klein and Manning (2003) introduced an approach where the objective function is the product of the probabilities of a generative phrase structure and a dependency parser. Model 1 of Collins (2003) is based on the dependencies between pairs of head words. On the other hand, the related work on this topic for discriminative parsing is sparse, and we are only aware of the following works. Carreras et al. (2008) and Rush et al. (2010) introduced frameworks for joint learning of phrase and dependency structures, and showed improvements on both tasks for English. These frameworks require special formulation of – one or both – parsing approaches while our approach allows the usage of arbitrary dependency parsers and any feature-based phrase structure parser.

Our motivation differs from these solutions as we focus on advancing one approach rather than achieving a joint optimum. Our approach can be regarded as a special stacking procedure (Nivre and McDonald, 2008), specifically, the stacking of a phrase structure parser with a dependency parser. In our previous work (Farkas et al., 2011), we reported results with a stacking approach for phrase structure parsing evaluated on a German corpus. We focus herein on the reverse direction as well, i.e. we define features for dependency parsers, we report performance outstanding scores on English for both directions, compare to the state-of-the-art results and carried out a detailed analysis of the stacking's contributions.

Wang and Zong (2010) introduced a procedure that exploits dependency parses to improve a phrase structure parser. They used automatic dependency parses for pruning the chart of a phrase structure parser and reported a significant improvement. One of our feature templates for the phrase structure parser can be regarded as a generalization of this approach.

3 Dependency Parser Exploiting Phrase Structure Parses

In this study we focus on graph-based parsers, however, our features from the phrase structures can also be adapted to a transition-based parser. **Graph-based dependency parsers** decompose the dependency structure into *factors*. Each factor of the first order graph-based parser corresponds to a dependency edge. McDonald et al. (2005a) first used second order factors which incorporates siblings of the head and dependent.

The second order algorithm of Carreras (2007) uses three second order factors: the sibling, the leftmost and rightmost grandchildren. The edge labeling is an integral part of the algorithm, which requires an additional loop over the labels. This algorithm has a complexity of $O(n^4L)$.

Koo and Collins (2010) presented a parser that can evaluate factors of three edges.

In this paper, we utilize the state-of-the-art parser of Bohnet (2010), a graph-based parser which employs online training with a perceptron which employs the MIRA update (Crammer and Singer, 2003). The parser contains a feature function for the first order factor, one for the sibling factor, and one for the grandchildren. We integrated in each of the functions features representing information on the phrase structure.

3.1 Features Defined on Phrase Structure Parses

We explore new feature templates to include features from the phrase structures. These templates are defined on the phrase structures themselves which is an important difference compared to previous work on joint parsing. Joint parsing requires a joint representation of phrase and dependency structures and is traditionally carried out by – explicitly or implicitly – converting the phrase structures into bilexical dependencies. For a comparative analysis of these two approaches please refer Section 6.3. Here we introduce our feature template utilizing the original phrase structures.

The feature templates provide information for the dependency parser about the embedding of the nodes in the phrase structure. A first order factor consists of the head and the dependent. A second order factor comprises of two connected edges, which additionally consist of a sibling or grandchild. The feature templates represent the location of the nodes in the phrase structure and the label of the phrase involved. We use the following abbreviations to define the feature templates: `L` represents the edge label; `hP`, `dP`, and `xP` the part-of-speech tag of the head, that of the dependent and the sibling or grandchild. The `hPoPS`,`dPoPS`, `xPoPS` denote the part-of-speech tag provided by the phrase structure parser for the head, dependent, and sibling or grandchild, respectively.

`hPS`, `dPS`, `xPS` denote the label of the phrase where the corresponding terminal node is embedded, respectively. `hPPS`, `dPPS`, `xPPS` denote the phrase that contains the phrase of the corresponding terminal of the head, dependent and sibling/grandchildren, respectively. The **r**elative **l**ocation, `rl`, of the head and dependent in the phrase structure: `rl=Phd`, if the head and dependent are terminals of the same phrase; `rl=Ph->Pd`, if the dependent is in a subphrase of the phrase of the head; `rl=Pd->Ph` denotes the inverse relation; otherwise `rl=Pd?Ph`. `rlx` stands for the relative location of the head and dependent, grandchild or sibling in the sentence, e.g. if the head is before or after the dependent in the sentence `rlx=h<d` or `rlx=d>h`. We count all pairwise combinations of the above nodes.

1. first order features:
   `L+hP+dP+hPS+rl`, `L+hP+dP+dPS+rl`, `L+hP+dP+hPS+dPS+rl`,
   `L+dP+hPS+hPPS+dPS+rl`, `L+dP+hP+dPPS+hPS+rl`, `L+hPoPS+dPoPS`.
2. sibling features:
   `L+hP+dP+xP+hPS+rlx`, `L+hP+dP+xP+dPS+rlx`, `L+hP+dP+xP+xPS+rlx`,
   `L+hPS+dPS+xPS+rlx`, `L+xP+hPS+dPS+xPS+rlx`, `L+xP+hPS+dPS+xPS+xPPS+rlx`,
   `L+xP+hPPS+dPS+xPS+rlx`, `L+xP+hPPS+dPS+xPS+xPS+rlx`,
   `L+hPoPS+dPoPS+xPoPS`.
3. We use as grandchildren features the same feature set as for the sibling features and replace x by the grandchild.

4 Phrase Structure Parser Exploiting Dependency Parses

Phrase structure parsers use a PCFG to extract possible parses. The full space of possible parses cannot be iterated through in practice, and they are usually pruned as a consequence. The *n-best list parsers* keep just the 50-100 best parses according to the PCFG (Charniak and Johnson, 2005). Other methods remove nodes and hyperedges whose posterior probability is under a pre-defined threshold from the forest (chart) (Huang, 2008). The task of the second stage is to select the best parse from the set of possible parses (i.e. rerank this set). These methods employ a large feature set (usually a few millions features) (Collins, 2000; Charniak and Johnson, 2005). The n-best list approaches can straightforwardly employ local and non-local features as well because they decide at the sentence-level (Charniak and Johnson, 2005) while involving non-local features is more complicated in the forest-based approaches and studies show only a minor empirical advantage of the latter approach (Huang, 2008; Wang and Zong, 2011). In this study, we experiment with n-best list reranking using a Maximum Entropy machine learning model and our goal is to investigate the extension of the standard feature set of these models by features extracted from the automatic dependency parse of the sentence in question.

4.1 Features Defined on Dependency Parses

The objective of the features defined for phrase structure parsing is to characterize the divergence/similarity of constituents with the corresponding part of the automatic (1-best) dependency parse of the sentence in question. We defined three feature templates for representing hyperedges (i.e. CFG rules applied over a certain span of words). We illustrate them on two possible subparses of Figure 1.

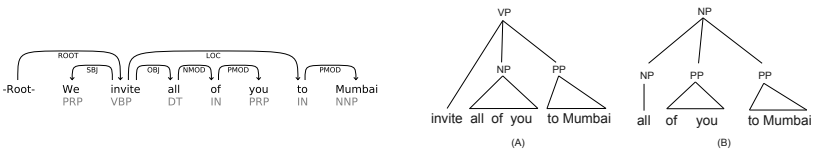


Figure 1: A dependency parse and two hyperedges (*A* and *B*) of the phrase structure chart for the introduction of feature templates.

outArc features are counting the dependency arcs which "go out" from the constituent in question. More precisely we count the words within the span whose parent in the dependency tree lays outside the span of words in question. We use the absolute count and the ratio of outArcs among the words of the span. The more arcs go out, the further away is the dependency subtree over the words of the constituent from a dominating subtree. Hence, these features try to capture the "phraseness" of the span of words in question based on the dependency tree. The example *A* in Figure 1 gets `outArc=1` and `outArcRatio=1/6` as only the parent of *invite* lays outside the constituent. For *B* we have `outArc=2` and `outArcRatio=2/5` because of *all* and *to*.

POSRel features indicate the dependencies which occur in the dependency parse (explicitly) as well as in the phrase-structure parse (implicitly). For this we gather the daughter constituents whose lexical head is linked in the (undirected) dependency tree to the head of the parent constituent. We define features from them using the pair of the two heads' POS tag and a

triplet using the POS tags and the corresponding dependency label. For *A* we have the following binary valued features: `VBP-DT`, `VBP-DT-OBJ` (from *invite–all*), `VBP-IN`, `VBP-IN-LOC` (from *invite–to*) as both daughter attachments have the corresponding arcs in the dependency tree. For *B* we do not extract features for the second PP daughter attachment as the lexical head of the parent (*all*) and the lexical head of the daughter (*to*) are not linked in the dependency parse.

ConstRel features are similar to `POSRel` but use the constituent labels rather than the POS tags of the heads. Thus, for *A* we have `VP-NP`, `VP-NP-OBJ` (from *invite–all*), `VP-PP`, `VP-PP-LOC` (from *invite–to*).

We also investigated the role of case and grammatical functions of the phrase structure parser in German and extended the `POSRel` and `ConstRel` feature sets by adding this information to the labels.

Note that the value of `outArc` is 1 iff the word span in question has a dominating dependency subtree in the automatic parse. Wang and Zong (2010) prune hyperedges with `outArc`$\neq 1$ thus this feature can be regarded as a generalization of their approach.

These features are similar in nature to the lexical head-based features of Charniak and Johnson (2005) and Versley and Rehbein (2009) but their role is different. Those dependency-like features try to describe the local hyperedges using the internal state of the parsing approach while we exploit here a globally (i.e. sentence-level) optimized dependency parse tree which represents external knowledge to the parser.

5 Experimental Setting

5.1 Data Sets

We used the Penn Treebank (section 23 served as test set and section 22 and 24 were used as development set in the phrase-structure and dependency experiments, respectively) with the Penn2Malt converter, and the head-finding rules of Yamada and Matsumoto (2003) for our experiments on English because most of the published state-of-the-art results were obtained on these datasets and conversion. For German, we used the Tiger Treebank and followed the split of the Parsing German (PaGe) shared task (Kübler, 2008) for phrase structure parsing. For dependency parsing, we used the dependency corpora of the CoNLL-2009 shared task, cf. (Hajič et al., 2009) (which consists of automatically converted trees from a part of the Tiger Treebank).

The phrase structure parsers traditionally conduct POS tagging during parsing (the different POS alternatives are on the chart). For obtaining POS tags for the dependency parsing experiments, we jackknifed the training data 10 fold, i.e. we trained the POS tagger on nine folds and tagged the tenth fold. On the English training set, the POS tagger of Toutanova et al. (2003) achieves an accuracy of 97.1, on the development set, 97.2 and the test set, 97.3. For German, we used a SVM-based tagger and we obtained on the training set an accuracy of 97.2, on the development set, 97.5 and on the test set, 97.4.

We followed the same jackknifing procedure in the stacking experiments in order to obtain the parses from which the features are extracted. We generated the dependency and constituent parses of the training corpus by training the respective parser on nine folds and parsing the tenth fold. During parsing the test set and development set we utilized the parses from the second parser which was trained on its entire training corpus.

5.2 Implementation Details

For our experiments, we utilize the second-order graph-based parser of the **dependency parsers** of Bohnet (2011)[1]. The second-order parser employs online training with a perceptron (Crammer and Singer, 2003) and it is based on a Hash Kernel. We employed this parser since it is quick to train, provides useful labeled dependency trees and achieves state-of-the-art accuracies. We used the transition-based parser that is included in the same package for our experiments as well.

For the **phrase structure parsing** experiments, we also employed state-of-the-art parsers. We used the first-stage parser of Charniak and Johnson (2005) for English and Bitpar (Schmid, 2004) for German. The latter employs a grammar engineered for German (Farkas et al., 2011). At the second stage, we used the 50 and 100-best parses to rerank and filtered out rare features (which occurred in less than 5 sentences). We employed the ranking MaxEnt implementation of the MALLET package (McCallum, 2002) and optimized the L2 regularizer coefficient on the development set.

5.3 Baseline Feature Sets

For conducting baseline **dependency parsing** experiments, we employed the feature set of Bohnet (2010). For the stacking of graph-based and transition-based dependency parsers, we used the feature template definitions from Nivre and McDonald (2008) which was extended by grandchildren factors (similarly to Torres Martins et al. (2008)).

For **phrase structure reranking** we utilized the features of Charniak and Johnson (2005) and for German we reimplemented the feature templates of Versley and Rehbein (2009) as baseline feature sets. The latter is the state-of-the-art feature set for German, which consists of features constructed from the lexicalized parse tree and its typed dependencies along with features based on external statistical information (like the clustering of unknown words according to their context of occurrences and PP attachment statistics gathered from the automatic POS tagged DE-WaC corpus, a 1.7G words sample of the German-language WWW).

5.4 Evaluation Metrics

For the evaluation of the dependency parses, we use the standard labeled (LAS) and unlabeled attachment scores (UAS) excluding punctuation for English and including punctuation for German. We use the predicted POS tags everywhere thereafter.

We use the standard `evalb` implementation of PARSEVAL on every sentence without length limitation as evaluation metric for phrase structure parsers. As the grammatical functions of constituents are important for downstream applications – especially in German –, we also report PARSEVAL scores on the conflation of constituent labels and grammatical functions (the scores in brackets in Table 3). We have to mention that our F-values are not directly comparable to the official results of PaGe Shared Task – which was our original goal – because the evaluation metric had a special implementation for calculating the F-measure (which differs from `evalb` for example in handling punctuation marks) and it used gold-standard POS tags in the input (which we thought to be unrealistic).

[1] Downloadable from http://code.google.com/p/mate-tools/

6 Results

6.1 Oracle Experiments

In order to validate the value of our new feature templates, we carried out the following oracle experiment. We used the gold standard parse trees from the second parser for training and parsing, i.e. we extracted features from the gold standard phrase structure parses for dependency parsing and vice versa. Our dependency parser achieved an UAS of 98.52 and LAS of 98.17 on the English test set. The re-ranking phrase structure parser achieved 95.78 where the oracle score – i.e. selecting the best candidate from the candidate list – is 96.83. Hence using the gold standard trees of the second parser yield scores close to the theoretical upper bound, which empirically proves that our new feature templates can effectively capture the information present in the other syntactic representation.

6.2 Dependency Parsing Results

Table 1 shows the results achieved on the test and development sets of English and German.

Our baseline parsers here are the transition-based parser (baseline T) and graph-based parser (baseline G) of (Bohnet, 2011). The row "G+T features" presents stacking results for the transition-based and graph-based parser. In this setting, the graph-based parser uses the output of the transition-based parser. This setting shows already a high accuracy improvement of 0.44 and obtained an unlabeled accuracy score of 93.5 in English. This result shows that the stacking idea of Nivre and McDonald (2008) works well on this higher accuracy level obtained with the offspring of their parsers.

The row "G+P features" shows the results when the graph-based parser exploits the features extracted from the phrase structure parser. The unlabeled accuracy score achieved the current best results for this data set with 93.95 UAS. When we use the output of the transition-based parser and the phrase structure parser as input to the graph-based parser (G+P+T features), we are able to obtain an even higher accuracy score of 94.04. Exploiting phrase structure parses achieves a gain in UAS of 0.89 and 0.54 employing them as an extension of the graph-based and the stacked dependency parser G+T. The added value of the features from the transition-based dependency parses on top of the extended feature set (i.e. the difference between G+P and G+P+T) is minor – and there is a decrease at 2 out of the 8 settings. These results indicate that the features gathered from the phrase structure parse contain almost all of the information which can be gathered from the transition-based dependency parse.

The two last rows show scores when we use in addition cluster-based features similarly to Koo et al. (2008). The cluster-based feature provided an improvement of 0.17 on top of the results of the graph-based parser (G+C features). The last row (G+P+T+C features) shows the scores for the features from all sources where we finally obtain 94.14 UAS.

Similarly to the results for English, we can also report large improvements for the German Tiger Treebank. With the G+T features, we gain 0.24 UAS and 0.71 with the "G+P features" compared with the "baseline G". When we utilized each feature set, the parser achieved 91.88 UAS and 89.90 LAS (0.81 improvements in UAS), which is the highest reported accuracy score for this data set.

	English		German	
	Devel.	Test	Devel.	Test
baseline T	91.46/90.10	92.71/91.56	90.27/87.76	90.39/88.05
baseline G	92.09/90.81	93.06/91.95	90.37/88.01	91.02/88.81
G+T features	92.34/90.96	93.50/92.53	90.45/88.29	91.26/89.13
G+P features	92.81/91.71	93.95/92.99	90.64/88.47	91.73/89.50
G+P+T features	92.83/91.70	94.04/93.10	90.70/88.54	91.69/89.69
G+C features	92.14/90.87	93.23/92.15	90.53/88.31	91.40/89.19
G+P+T+C features	92.96/91.80	94.14/93.17	90.73/88.50	91.88/89.90

Table 1: Dependency parsing results achieved by the enriched feature sets. The accuracy scores are statistically significant between the baseline G and G+T, G+P, and G+P+T+C with $p < 0.001$ and between G+P and G+P+T+C with $p<0.05$. We used two sample t-test.

6.3 Stacking of Converted Parses

An alternative approach for stacking phrase structures into a dependency parser is to convert the phrase structures into dependency trees and extract features from this representation. We investigated this approach on the English dataset[2]. We used the Penn2Malt converter on the automatic phrase structure parses and then employed the same dependency stacking features which were used for "G+T features" and were described in Section 5.3. Table 2 repeats the scores from Table 1 achieved by the baseline parser and by the feature templates defined directly on phrase structures and compares these scores with the results achieved by the conversion-based stacking.

	English	
	Devel.	Test
baseline G	92.09/90.81	93.06/91.95
G+P converted	92.48/91.23	93.74/92.73
G+P direct	92.81/91.71	93.95/92.99

Table 2: Results achieved by utilizing features defined directly on phrase structures vs. defined on converted trees. The accuracy scores are statistically significant between the two approaches with $p<0.05$. We used two sample t-test.

6.4 Phrase Structure Parsing Results

Table 3 summarize the results achieved by phrase structure parsers[3]. In the reranking approach we used 50-best lists for English and 100-best lists for German as it is standard in previous work.

The "generative" row stands for the results achieved by the first-stage parsers. The "oracle" score is the highest available score in the 50-best and 100-best lists, thus serve as an upper

[2]Unfortunately, we do not have access to the conversion scripts for German. As the conversion in the reverse direction (i.e. converting dependency parses into phrase structures) is not straightforward we restricted ourself to phrase structure to dependency conversion and the English dataset.

[3]Note that the difference in German scores between this table and the scores reported in our previous study (Farkas et al., 2011) is because we report score on each sentences – and compare our results here against re-trained baseline parsers – while we limited the evaluation to sentences shorter than 40 tokens in the previous study – in order to be able to compare those results against previously published scores on the same corpus.

bound for rerankers. The last three rows contain the results of reranking on top of first-stage parsers employing the baseline[4], only the new features defined from dependency parses (dep) and their union (bl+dep).

	English		German	
	Devel.	Test	Devel.	Test
generative	88.86	89.71	77.83 (66.88)	76.81 (65.76)
oracle	95.90	96.83	86.18 (76.10)	83.37 (73.07)
baseline features	90.59	91.38	79.76 (67.99)	78.41 (67.05)
dep features	89.90	90.66	79.82 (69.28)	78.01 (67.46)
bl+dep features	90.92	91.72	80.90 (70.39)	79.45 (68.73)

Table 3: Phrase structure parsing PARSEVAL scores achieved by utilizing various feature sets. The figures in brackets for German are PARSEVAL scores including grammatical functions. The accuracy scores are statistically significant between "baseline features" and "bl+dep features" with $p<0.05$. We used two sample t-test.

As Table 3 shows, the reranking parser using only our features constructed from the automatic dependency parse of the sentence achieved a 1.0 improvement on the English development and test sets, respectively and they added a value of 0.3 over the baseline feature set as well.

The results for German are even more convincing. The simple dependency parse-based features are competitive with the German-specific feature set of Versley and Rehbein (2009). Moreover these two feature sets have a certain level of diversity as their union could achieve significantly better results (over 1.0 point of improvements), than any of them alone. The added value of the dependency parse-based features is higher if we consider the evaluation metric which also takes into account the grammatical function labels as well.

6.5 Comparison to previous work

Table 4 shows labeled and unlabeled accuracy scores of previous work reported for the Penn2Malt conversion with the head finding rules of Yamada and Matsumoto (2003). Carreras et al. (2008) (CCK) combine a dependency parsing decoder with parsing based on TAG. Koo and Collins (2010) (Koo'10) used a parser with third order factors and obtained an unlabeled accuracy of 93.04. Martins et al. (2010) introduced a parser (Turbo) with even higher accuracy (93.26 UAS), which is one of the highest accuracy scores for a parser that does not employ additional sources of information.

The following approaches use additional sources of information. Suzuki et al. (2009) (SICC) use a second order dependency parser that employs siblings and grandchildren factors. They apply the semi-supervised learning approach of Suzuki and Isozaki (2008) to dependency parsing and include additionally the cluster-based features of Koo et al. (2008) (KCC).

The improvement of the accuracy achieved by our stacking approach is surprisingly high. We achieved an error reduction of 16.5% (an absolute gain of 1.1 percentage points) over our baseline and it outperforms the best reported scores with combining dependency and phrase structure parses (Carreras et al., 2008) by 0.6.

[4]The small difference between the original Brown parser and reranking with baseline features is due to the different MaxEnt implementations employed.

	English (UAS)			German (LAS)	
	Devel.	Test		Devel.	Test
CCK'08	92.5	93.5	GHPT'09		87.28
Koo'10		93.04	B'10		88.06
Turbo'10		93.26			
This work (G+P+T)	92.14	94.04		88.54	89.69
KCC'08	91.85	93.16			
SICC'09		93.8			
This work (G+P+T+C)	92.96	94.14		88.50	89.90

Table 4: Comparison to previous work. We report UAS only for the English Penn Treebank since usually unlabeled scores were published and labeled attachments scores only for the German Tiger Treebank since in the CoNLL shared task 2009 only labeled attachment scores were reported.

For German the two top scoring parsers of the CoNLL 2009 shared task on Syntactic and Semantic Dependency parsing were the parser of Gesmundo et al. (2009), who had the best overall scores and the parser of Bohnet (2010), which performed best on English and German. However we note that directly comparing these results to our ones is not fully fair since the POS tagging accuracy of the data sets were lower than 95.5 at the CoNLL 2009 shared task.

Table 5 sketches the **phrase-structure parsing** results in the context of previously published results. We retrained the current version of the Charniak and Johnson (2005) parser[5] (C&J) for English. We also cite the scores reported by Rush et al. (2010) who also exploited phrase structure and dependency parsers together. Our final scores are competitive with the best reported supervised PARSEVAL score in the Penn Treebank by (Huang, 2008) who also extended the feature set of Charniak and Johnson (2005) and applied forest-based reranking.

	English			German	
	Devel.	Test		Devel.	Test
C&J	90.60	91.36	Berkeley	76.60 (65.25)	76.05 (64.00)
Rush'10	–	90.7	PyNLP	76.76 (66.29)	76.27 (65.21)
Huang'08	–	91.69			
This work (bl+dep)	90.92	91.72		80.90 (70.39)	79.45 (68.73)

Table 5: Comparison to previous work. Our method significantly outperforms state-of-the-art systems.

For German, we trained the current version of the Berkeley parser[6] (Petrov et al., 2006) and PyNLP[7] parsers (Versley and Rehbein, 2009). Note that the results of the generative Berkeley parser is competitive with the PyNLP parser which utilizes reranking, while the first-stage parser, which we employ here – using a fine-tuned German-specific tree annotation schema – outperforms both of them. The discriminative approach with the baseline feature set we employed gave an improvement of 2 percentage points over the first-stage parser and the features gathered from dependency parsers could add an additional 1 point to this.

[5] http://www.cog.brown.edu/ mj/Software.htm
[6] http://code.google.com/p/berkeleyparser/
[7] https://bitbucket.org/yannick/pytree

7 Discussion

We manually analyzed the outputs of the parsers in order to identify linguistic constructions where one or another parser performed better. Our findings help understand the reasons why the approach works and they might suggest ways of improving data-driven parsers as well.

7.1 Why did the phrase structures help dependency parsing?

In order to find the explanation why phrase structure parse-based features help the dependency parser we manually investigated the sentences of the English development set where the parsers with and without these features gave different output.

We found that the most frequent error fixing categories are related to longer phrases, typically noun phrases and named entities. For example, the phrase structure parser could recognize the phrase boundaries of *US Today 's total paid ad pages* and *St. Therese , Quebec* and with the phrase structure-based features the dependency parser was able to fix its errors here. Note that in the latter example if the parser analyses the entity as two separated entities *St. Therese* and *Quebec*, it introduces 3 attachment errors (for *Therese*, for the comma and for *Quebec*). Besides noun phrases the parses of institutionalized phrases like *with or without* and *rather than to* were also considerable improved. These improvements can be probably attributed to the phrase-based thinking of the second parser.

Phrase structure-based features improved considerably the accuracy of coordinations. It is a known issue that the representation of coordination in the dependency structure has an impact on the accuracy. The Penn2Malt converter builds dependency structures where the conjunction is the head. Nilsson et al. (2007) showed that this representation of coordinations yields a lower parsing accuracy – across four languages – than a representation where the conjunction and the members of the coordination form one chain. Our features defined on the phrase structure trees can be regarded as a third type of representation of coordination. Although these features help in the current setting we expect a smaller contribution if a different dependency representation (like the chain type) is targeted.

The counting of the errors – or error fixes – related to coordination is not straightforward because besides the wrong attachment of a `CC` it introduces several attachment errors in the construction. For instance, we observed several article attachment errors related to this as well. For example in *the agreement and consent decree* the article can be attached to the *agreement* or to the *decree* according to the analysis of the coordination.

The rest of the improvements is heterogenous but we observed two other quite frequent error types. The first one is the attachment to a verb if multiple verbs are present in the sentence, most typically the choice between the auxiliaries and the main verbs. For instance in *having just passed* the dependency parser attached *just* to the main verb while was able to correctly attach it to the auxiliary with the help of phrase structure-based features. We counted down the attachment errors of the dependency parser where it chose a wrong verb. The phrase structure-based features could eliminate 245 such errors while introduced 117 new attachment errors.

A related interesting issue is that the incorporation of the phrase structure-based features into the graph-based parser improves the selection of the sentence's root by almost 1 percentage point (the F-score of the `ROOT` label increased from 96.24 to 97.1). This might be an artifact of the two-stage approach of the phrase structure parser, i.e. the reranking second stage could

choose from candidates having different roots. We intend to more deeply investigate this issue in the future as it seems to be promising to design a reranking phase for dependency parsers which can select from parses with different roots.

7.2 Why did the dependency structures help phrase structure parsing?

We followed the methodology of contribution analysis for phrase structure parses and we manually investigated the sentences of the English development set where the discriminative parsers with and without these features gave different output. We identified two categories of improvement. The first one is the "attachment" of adverbs and adjectives, especially in the case of collocational modifiers like *more like* and *many more*. The second phenomenon where the dependency parse-based features could eliminate a considerable amount of errors – much more than was introduced – is PP attachment. The explanation for both of these contributions might be the "lossless" lexicalization of the dependency parsers. Although the PCFG parser of Charniak (2000) is lexicalized[8] and the re-ranking stage is deeply lexicalized, using only 50 candidate parses and a frequency-based feature pruning implies some extent of information loss while dependency parsers can exploit very rare lexical cues.

7.3 What is the difference between stacking directly and stacking after conversion?

We also manually compared the parses of the two stacking approaches, i.e. the one where the features are defined directly on phrase structures versus the other where the features are defined on a dependency tree which were automatically converted from the phrase structures (see Section 6.3). Although the difference between the two approaches is significantly different we cannot conclude that the first one is clearly better than the second one. There are 214 sentences in the development set where the direct appraoch achieves higher attachment scores than the other while there are also 197 sentences where its scores were worse.

Interestingly, the most frequent linguistic constructions which are responsible for these differences are related to conjunctions at both the directions. We found that in the case of noun phrase coordinations, the features defined directly on the phrase structures are more useful than after conversion. This further supports our earlier discussion that our features as a representation type of coordination is more useful than the "conjunct is the head" representation used in the Penn2Malt conversion.

On the other hand, in the case of compound sentences (i.e. the conjunction of independent clauses) the converted parses contributed frequently more than features directly defined on the phrase structures (this is expressed also by the F-score of the `ROOT` label which is 0.3 higher in the first case). The cause for this effect is that the phrase structure puts the two (or more) independent clauses to same level while the dependency tree – at least in the Penn2Malt conversion – has to label a single token as the root of the sentence. The root selection among the heads of the independent clauses follows a deterministic rule during the conversion process, thus the converted trees do not face the problem of root ambiguity in these compound sentences.

8 Conclusions

We proposed here the stacking of dependency and phrase structure parsers. We introduced a feature set for describing phrase structures for data-driven dependency parsers and dependency

[8] However, it utilizes just the head of the parent instead of bi- (or higher order) lexicalization.

parses for discriminative phrase structure parsers. We investigated the performance of our stacking approach on English and German and we reported a significant improvement over the state-of-the-art parsers in both directions. We believe that the key factor of this success is that we designed our framework to advance one parser exploiting the other, rather than optimizing a joint objective function.

In our experiments both parsers utilize the same (training) data set without having access to external information. We argue that our results – especially the gain at dependency parsers – indicate that state-of-the-art parsers can be still further improved without additional data. To facilitate these further developments, we manually analyzed several parses of different approaches and identified linguistic phenomena where the gain was considerable. For instance, for the dependency parsers these phenomena are named entities, longer noun phrases, coordination and root selection. Investigating them more deeply could probably reveal parsing techniques which can yield considerable added value to current procedures.

Acknowledgement

This work was supported by the Deutsche Forschungsgemeinschaft (DFG) via the SFB 732 “Incremental Specification in Context”, projects D4 (PI Helmut Schmid) and D8 (PI Jonas Kuhn).

References

Bohnet, B. (2010). Top accuracy and fast dependency parsing is not a contradiction. In *Proceedings of COLING*, pages 89–97.

Bohnet, B. (2011). Comparing advanced graph-based and transition-based dependency parsers. In *International Conference on Dependency Linguistics*, pages 282–289.

Carreras, X. (2007). Experiments with a Higher-order Projective Dependency Parser. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 957–961.

Carreras, X., Collins, M., and Koo, T. (2008). Tag, dynamic programming, and the perceptron for efficient, feature-rich parsing. In *CoNLL 2008: Proceedings of the Twelfth Conference on Computational Natural Language Learning*, pages 9–16.

Charniak, E. (2000). A maximum-entropy-inspired parser. In *Proceedings of the first conference on North American chapter of the Association for Computational Linguistics*, pages 132–139.

Charniak, E. and Johnson, M. (2005). Coarse-to-fine n-best parsing and maxent discriminative reranking. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, ACL '05, pages 173–180.

Collins, M. (2000). Discriminative reranking for natural language parsing. In *Proceedings of the Seventeenth International Conference on Machine Learning*, ICML '00, pages 175–182.

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

Crammer, K. and Singer, Y. (2003). Ultraconservative online algorithms for multiclass problems. *Journal of Machine Learning Research*, 3:951–991.

Eisner, J. M. (1996). Three new probabilistic models for dependency parsing: An exploration. In *Proceedings of the 16th International Conference on Computational Linguistics (COLING)*, pages 340–345.

Farkas, R., Bohnet, B., and Schmid, H. (2011). Features for phrase-structure reranking from dependency parses. In *Proceedings of the 12th International Conference on Parsing Technologies*, pages 209–214.

Gesmundo, A., Henderson, J., Merlo, P., and Titov, I. (2009). A Latent Variable Model of Synchronous Syntactic-Semantic Parsing for Multiple Languages. In *Proceedings of the 13th Conference on Computational Natural Language Learning (CoNLL-2009)*.

Hajič, J., Ciaramita, M., Johansson, R., Kawahara, D., Martí, M., Màrquez, L., Meyers, A., Nivre, J., Padó, S., Štěpánek, J., Straňák, P., Surdeanu, M., Xue, N., and Zhang, Y. (2009). The conll-2009 shared task: Syntactic and semantic dependencies in multiple languages. In *Proceedings of the 2009 CoNLL Shared Task*, pages 1–18.

Huang, L. (2008). Forest reranking: Discriminative parsing with non-local features. In *Proceedings of ACL-08*, pages 586–594.

Klein, D. and Manning, C. D. (2003). Fast exact inference with a factored model for natural language parsing. In *Proceedings of Advances in Neural Information Processing Systems*, volume 15.

Koo, T., Carreras, X., and Collins, M. (2008). Simple semi-supervised dependency parsing. In *Proceedings of ACL*, pages 595–603.

Koo, T. and Collins, M. (2010). Efficient third-order dependency parsers. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 1–11.

Kübler, S. (2008). The PaGe 2008 shared task on parsing german. In *Proceedings of the Workshop on Parsing German*, pages 55–63.

Martins, A. F. T., Smith, N. A., Xing, E. P., Aguiar, P. M. Q., and Figueiredo, M. A. T. (2010). Turbo parsers: dependency parsing by approximate variational inference. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, EMNLP '10, pages 34–44.

McCallum, A. K. (2002). Mallet: A machine learning for language toolkit. http://mallet.cs.umass.edu.

McDonald, R., Crammer, K., and Pereira, F. (2005a). Online large-margin training of dependency parsers. In *Proceedings of ACL*, pages 91–98.

McDonald, R., Pereira, F., Ribarov, K., and Hajic, J. (2005b). Non-projective dependency parsing using spanning tree algorithms. In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*, pages 523–530.

Nilsson, J., Nivre, J., and Hall, J. (2007). Generalizing tree transformations for inductive dependency parsing. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*.

Nivre, J. (2003). An efficient algorithm for projective dependency parsing. In *Proceedings of IWPT*, pages 149–160.

Nivre, J., Hall, J., and Nilsson, J. (2004). Memory-based dependency parsing. In Ng, H. T. and Riloff, E., editors, *HLT-NAACL 2004 Workshop: Eighth Conference on Computational Natural Language Learning (CoNLL-2004)*, pages 49–56.

Nivre, J. and McDonald, R. (2008). Integrating graph-based and transition-based dependency parsers. In *Proceedings of ACL-08: HLT*, pages 950–958.

Petrov, S., Barrett, L., Thibaux, R., and Klein, D. (2006). Learning accurate, compact, and interpretable tree annotation. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 433–440.

Rush, A. M., Sontag, D., Collins, M., and Jaakkola, T. (2010). On dual decomposition and linear programming relaxations for natural language processing. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 1–11.

Schmid, H. (2004). Efficient parsing of highly ambiguous context-free grammars with bit vectors. In *Proceedings of Coling 2004*, pages 162–168.

Suzuki, J. and Isozaki, H. (2008). Semi-supervised sequential labeling and segmentation using giga-word scale unlabeled data. In *Proceedings of ACL-08*, pages 665–673, Columbus, Ohio. Association for Computational Linguistics.

Suzuki, J., Isozaki, H., Carreras, X., and Collins, M. (2009). An empirical study of semi-supervised structured conditional models for dependency parsing. In *Proceedings of Conference on Empirical Methods in Natural Language Processing*, pages 551–560.

Titov, I. and Henderson, J. (2007). A latent variable model for generative dependency parsing. In *Proceedings of IWPT*, pages 144–155.

Torres Martins, A. F., Das, D., Smith, N. A., and Xing, E. P. (2008). Stacking dependency parsers. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 157–166, Honolulu, Hawaii. Association for Computational Linguistics.

Toutanova, K., Klein, D., Manning, C. D., and Singer, Y. (2003). Feature-rich part-of-speech tagging with a cyclic dependency network. In *Proceedings of the Conference on North American chapter of the Association for Computational Linguistics*, pages 252–259.

Versley, Y. and Rehbein, I. (2009). Scalable discriminative parsing for german. In *Proceedings of the 11th International Conference on Parsing Technologies (IWPT'09)*, pages 134–137.

Wang, Z. and Zong, C. (2010). Phrase structure parsing with dependency structure. In *Proceedings of COLING*, pages 1292–1300.

Wang, Z. and Zong, C. (2011). Parse reranking based on higher-order lexical dependencies. In *Proceedings of 5th International Joint Conference on Natural Language Processing*, pages 1251–1259.

Yamada, H. and Matsumoto, Y. (2003). Statistical Dependency Analysis with Support Vector Machines. In *Proceedings of International Conference on Parsing Technologies*, pages 195–206.

Semantic Cohesion Model for Phrase-based SMT

Minwei FENG[1] *Weiwei SUN*[2] *Hermann NEY*[1]
(1) Human Language Technology and Pattern Recognition Group,
Computer Science Department,
RWTH Aachen University,
Aachen, Germany
(2) The MOE Key Laboratory of Computational Linguistics,
Institute of Computer Science and Technology,
Peking University,
Beijing, China
feng@cs.rwth-aachen.de, ws@pku.edu.cn, ney@cs.rwth-aachen.de

ABSTRACT
In this paper, we propose a novel semantic cohesion model. Our model utilizes the predicate-argument structures as soft constraints and plays the role as a reordering model in the phrase-based statistical machine translation system. We build a translation system with GALE data. Experimental results on the NIST02, NIST03, NIST04, NIST05 and NIST08 Chinese-English tasks show that our model improves the baseline system by 0.93 BLEU 0.98 TER on average. We also compare our method with a syntax-augmented model (Cherry, 2008), and demonstrate the importance of predicate-argument semantics in machine translation.

KEYWORDS: statistical machine translation, semantic role labeling.

Proceedings of COLING 2012: Technical Papers, pages 867–878,
COLING 2012, Mumbai, December 2012.

1 Introduction

In recent years, there are growing interests in incorporating semantics into statistical machine translation (SMT) (Wu and Fung, 2009; Liu and Gildea, 2010; Gao and Vogel, 2011; Baker et al., 2012). Among all existing semantic representations, semantic role labeling (SRL) aims at automatically analyzing predicate-argument structures and can capture essential meaning of sentences. Since the seminal work of (Gildea and Jurafsky, 2002), quite a few researchers concentrate on resolving SRL with different machine learning methods. Nowadays, good SRL systems can be built based on accurate syntactic parsers for English, as well as many other languages.

In this paper, we explore predicate-argument analysis of source sentences to improve phrase-based SMT systems. On one hand, the predicate-argument event layer of SRL captures global dependencies which is crucial for the MT output quality. On the other hand, the semantic role information contained in SRL also provide a good clue to the appropriateness of a phrase segment chosen by a translation system. Compared to the syntactic representation in both constituency and dependency formalisms, SRL focuses more on modeling the skeleton of a sentence.

To exploit the two attributes of SRL, we propose two types of constraints which are implemented as two models in the decoder. The first model restrains the translation process so that it is consistent with the global dependency in the source sentence. The second model inspects the source sentence segmentation so that each source phrase is consistent with the semantic roles.

We conduct experiments on the NIST02, NIST03, NIST04, NIST05 and NIST08 Chinese-English tasks. Experimental results show that our model improves the baseline system by 0.93 BLEU 0.98 TER on average. We also compare our method with a syntax-augmented model (Cherry, 2008), and demonstrate the importance of predicate-argument semantics in machine translation.

The remainder of this paper is organized as follows: Section 2 describes the semantic soft constraints proposed for translation system. Section 3 introduces a related work (Cherry, 2008) as we compare our method with it. Section 4 provides the experimental configuration and results. Section 5 briefly summarizes recent work on employing different semantics related techniques. Conclusions will be given in Section 6.

2 Semantic Cohesion Model

In this section, we describe the proposed idea. Firstly, we will briefly describe the basement of this research: the principle of statistical machine translation. Secondly, the SRL framework will be given. Thirdly, we demonstrate how the semantic role information can be used for translation.

2.1 Principle

In statistical machine translation, we are given a source language sentence $f_1^J = f_1 \dots f_j \dots f_J$. The objective is to translate the source into a target language sentence $e_1^I = e_1 \dots e_i \dots e_I$. The strategy is among all possible target language sentences, we will choose the one with the highest probability:

$$\hat{e}_{\hat{i}}^{\hat{I}} = \arg\max_{I, e_1^I} \{Pr(e_1^I | f_1^J)\} \quad (1)$$

We model $Pr(e_1^I|f_1^J)$ directly using a log-linear combination of several models (Och and Ney, 2002):

$$Pr(e_1^I|f_1^J) = \frac{\exp\Big(\sum_{m=1}^{M} \lambda_m h_m(e_1^I, f_1^J)\Big)}{\sum_{I', e'^{I'}_1} \exp\Big(\sum_{m=1}^{M} \lambda_m h_m(e'^{I'}_1, f_1^J)\Big)} \quad (2)$$

The denominator is to make the $Pr(e_1^I|f_1^J)$ to be a probability distribution and it depends only on the source sentence f_1^J. For search, the decision rule is simply:

$$\hat{e}_{\hat{i}}^{\hat{I}} = \arg\max \Big\{ \sum_{m=1}^{M} \lambda_m h_m(e_1^I, f_1^J) \Big\} \quad (3)$$

The model scaling factors λ_1^M are trained with Minimum Error Rate Training (MERT).

In this paper, the phrase-based machine translation system is utilized (Och et al., 1999; Zens et al., 2002; Koehn et al., 2003). The translation process consists in segmentation of the source sentence according to the phrase table which is built from the word alignment. The translation of each of these segments is then just extracting the target side from the phrase pair. With the corresponding target side, the final translation is the composition of these translated segments. In this last step, reordering is allowed.

2.2 Semantic Role Labeling

In the last decade, there has been an increasing interest in SRL on several languages, which consists of recognizing arguments involved by predicates in a given sentence and labeling their semantic types. Typical semantic classes include *Agent*, *Patient*, *Source*, *Goal*, and so forth, which are core arguments to a predicate, as well as *Location*, *Time*, *Manner*, *Cause*, and so on, which are adjuncts. In order to indicate exactly what semantic relations hold among a given predicate and its associated participants and properties, the role-bearing constituents must be identified and their correct semantic role labels assigned.

Such sentence-level semantic analysis of text is concerned with the characterization of events and is therefore important to understand the essential meaning of the original input language sentences – *who* did *what* to *whom*, for *whom* or *what*, *how*, *where*, *when* and *why*? Different from many other semantic representation formalisms, this shallow semantic interpretation abstracts important predicate-argument structural information away from syntactic structure and may potentially benefit machine translation, as well as many other NLP applications.

2.3 Semantic Cohesion for Translation

We now explain how the SRL is utilized in the decoder. Suppose the source language is temporarily English, we detect and classify semantic roles of all predicates for the source sentence before translation. The Figure 1 is an example with Propbank-style annotations. Figure 1 consists of four rows. The first row is the orginal source sentence. The next three rows demonstrate the event layers of the sentence. Each event layer has one predicate and corresponding arguments. The source sentence in Figure 1 has three predicates. The first predicate *make* and second predicate *distribute* governs two arguments A0 (namely proto-Agent) and A1 (namely proto-Patient) respectively; third predicate *base* possesses two semantic

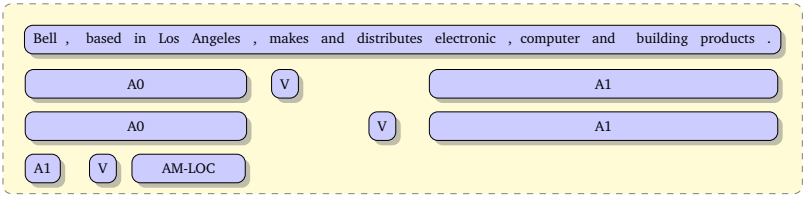


Figure 1: An example of SRL. The source sentence is given at top.

roles A1 and AM-LOC (namely location). Then a model will be added into the decoder as a new feature of the log-linear framework (Equation 2). The algorithm is as follows:

Given the source sentence and its predicate-argument structural information, during the translation process, every time one hypothesis is extended, the added model checks if the source semantic analysis contains one structure S such that:

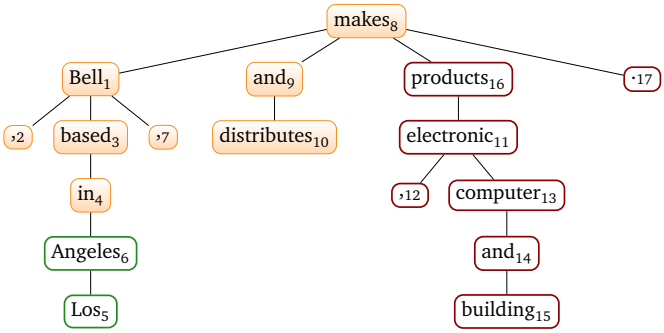
- Its translation is already started (at least one word is covered)
- It is interrupted by the new added phrase (at least one word in the new source phrase is not in S)
- It is not finished (after the new phrase is added, there is still at least one uncovered source word in S)

If so, we say this hypothesis violates the structure S, and the model returns the number of structures that this hypothesis violates.

We use two kinds of structures. In other words, we add two models/features into the log-linear framework. The structure of the first model (we name it SRL1) is the whole predicate-argument structure, i.e. event layer. The SRL1 feature will report how many event layers that one search state violates. For the second model (we name it SRL2), the structure is semantic role. The SRL2 feature will calculate the amount of semantic roles that one search state violates. In Figure 1, there are three event layers and six semantic roles. Suppose currently only the first source word *Bell* is already covered/translated, and then the decoder decides to translate the source word *computer*. Then the feature SRL1 will give penalty 1. Because this decision violates the third event layer in Figure 1. The third event layer was starting to be translated (*Bell* is covered), however, the decoder jumps to *computer* before it finishes current event layer(A1 and AM-LOC have not been translated yet). The feature SRL2 will give penalty 2 because the two semantic role A0 have been violated. During search, we have the access to the coverage vector which stores those source words that have been translated. So based on the source sentence semantic analysis and current coverage vector, the two feature values can be easily computed.

3 Syntactic cohesion model

We now introduce a related work as comparison will be presented later in this paper. (Cherry, 2008) proposed a syntactic cohesion model. The core idea is that the syntactic structure of the source sentence should be preserved during translation. This structure is represented by a source sentence dependency tree. To keep syntactic cohesion, the decoding process should not break



[Bell , based in Los Angeles , makes and distributes electronic , computer and building products .]

Figure 2: A dependency tree example. The source sentence is given at bottom.

this dependency structure. (Cherry, 2008) used his model as a new feature of the log-linear decoding framework and showed improvement on English-French direction. We implement this model in the phrase-based decoder and report results on Chinese-English translation.

To illustrate the method, we use the Figure 2 as an example. Figure 2 is a dependency tree (again, suppose the source language is now English) . Every node represents a word. We also put the position of the word in the node. So $Bell_1$ means $Bell$ is the first word of the sentence. The algorithm is as follows:

Given the source sentence and its dependency tree, during the translation process, once a hypothesis is extended, check if the source dependency tree contains a subtree T such that:

- Its translation is already started (at least one node is covered)
- It is interrupted by the new added phrase (at least one word in the new source phrase is not in T)
- It is not finished (after the new phrase is added, there is still at least one free node in T)

If so, we say this hypothesis violates the subtree T, and the model returns the number of subtrees that this hypothesis violates.

In Figure 2, nodes filled with yellow means the words have been already covered/translated. Now suppose the length of the new added phrase is 1, then according to the above algorithm only position 5 and 6 (green rectangle) are good candidates. Choosing other source words (red rectangle) to translate will violate the subtree $in_4 - Los_5 - Angeles_6$.

4 Experiments

4.1 Experimental Setup

Our baseline is a phrase-based decoder, which includes the following models: an n-gram target-side language model, a phrase translation model and a word-based lexicon model. The

latter two models are used for both directions: $p(f|e)$ and $p(e|f)$. Additionally we use phrase count features, word and phrase penalty. The reordering model for the baseline system is the distance-based jump model which uses linear distance. This model does not have hard limit. We list the important information regarding the experimental setup below. All those conditions have been kept same in this work.

- lowercased training data (Table 1) from GALE task
  alignment trained with GIZA++
- tuning corpus: NIST06 test corpora: NIST02 03 04 05 and 08
- 5-gram LM (1 694 412 027 running words) trained by SRILM toolkit (Stolcke, 2002) with modified Kneser-Ney smoothing
  LM training data: target side of bilingual data.
- BLEU (Papineni et al., 2001) and TER (Snover et al., 2005) reported
  all scores calculated in lowercase way.
- Stanford Parser (Levy and Manning, 2003) used to get the Chinese constituent tree for the SRL and the dependency tree for the syntactic cohesion model

	Chinese	**English**
Sentences	5 384 856	
Running Words	115 172 748	129 820 318
Vocabulary	1 125 437	739 251

Table 1: training data statistics

4.2 A Full Parsing Based Chinese SRL System

4.2.1 Background

SRL methods that are successful on English are adopted to resolve Chinese SRL (Xue, 2008; Sun, 2010). Previous work indicates that syntactic information is very important for SRL and full parsing based approaches are considerably better than shallow parsing based ones. Based on a phrase-structure parsing, SRL is usually formulated as a constituent classification problem. In particular, SRL is divided into three sub-tasks: 1) pruning with a heuristic rule, 2) argument identification (AI) to recognize arguments, and 3) semantic role classification (SRC) to predict semantic types. To efficiently excluded non-arguments, a pruning procedure is executed to filter out constituents that are highly unlikely to be semantic roles of a predicate p. In the AI sub-task, for every candidate constituent c, a binary classifier is employed to determine whether c is an argument of p. In the SRC sub-task, all arguments recognized in the AI step are assigned detailed semantic types by a multi-class classifier. Distinct from the constituent classification method introduced above, we use a constituent chunking based method to acquire predicate-argument structures in this paper.

4.2.2 Semantic Chunking Based SRL

SRL is performed on the output of a syntactic parser, and only phrases in the parse tree are taken as possible candidates. If there is no phrase in the parse tree that shares the same text

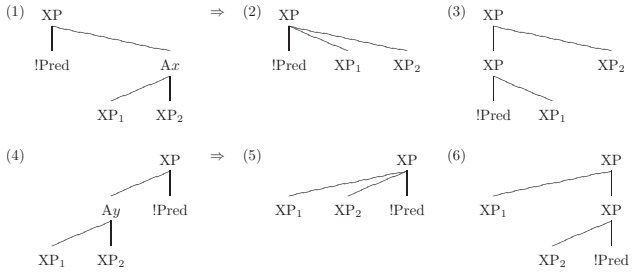


Figure 3: Parsing errors that can be tolerated by full parsing based constituent chunking.

span with an argument in the manual annotation, the system cannot possibly get a correct prediction. In other words, the best the system can do is to correctly label all arguments that have a counterpart node in the parse tree.

In this paper, we implement a semantic chunking method which can tolerate some syntactic parsing errors. First, our system collects all c-commanders[1] and puts them in order. Because c-commanders of a predicate are not overlapped with each other and compose the whole sentence, we can take this step as a sequentialization procedure. Sun et al. (2008) present a theoretical analysis about argument positions and suggest that an argument should c-command a predicate. Therefore, our sequentialization precedure keeps most semantic roles. On basis of sequentialized constituents, we define semantic chunks which do not overlap nor embed using IOB2 representation (Ramshaw and Marcus, 1995) and transfer the SRL problem as a **constituent tagging** problem. Our definition of semantic chunks is described below.

- Constituent outside an argument receive the tag *O*.
- For a sequence of constituents forming a semantic role of *Ax*, the first constituent receives the semantic chunk label *B(egin)-Ax*,
- and the remaining ones receive the label *I(nside)-Ax*.

Developing features has been shown crucial to advancing the state-of-the-art in SRL. To achieve good Chinese SRL results, we utilize rich syntactic features introduced in (Sun, 2010). For sequential tagging, we use a first order linear-chain global linear model and estimate parameters with structured preceptron (Collins, 2002).

Our semantic chunking method can tolerate two types of parsing errors that are shown in Figure 3. Assume tree structures (1 and 4) on the left hand side are the correct syntactic analysis, while tree structures (2, 3, 5 and 6) on the right hand side are some wrong analysis. Though a constituent classification system, the arguments Ax and Ay can not be recovered since there is no node to express them. In our constituent chunking system, however, when these errors occur, the arguments can still be found, if XP$_1$ is assigned a label *B-Ax* or *B-Ay* and XP$_2$ is assigned a label *I-Ax* or *I-Ay*.

[1]C-command is an concept in X-bar theory. Assuming α and β are two nodes in a syntax tree: α C-commands β means every parent of α is ancestor of β.

The full parsing based semantic chunking system was first introduced in (Sun, 2012). To learn more empirical evaluation results of this system as well as a comparative study of full and shallow parsing based SRL, readers can refer to (Sun, 2012).

4.2.3 An Example

An example is illustrated in Figure 4, where the predicate is the verb "调查/investigate". To find all arguments and adjuncts, our system first employs Stanford parser to obtain a phrase-structure analysis. Based on the syntactic tree, our SRL system then collects all c-commanders of the predicate, including all square nodes. Finally, a sequence classifier is applied to assign these candidate nodes chunk labels *B-A0*, *B-TMP*, *B-MNR* and *B-A1*. Based on these labels, we can know the event structure, e.g. the proto-Patient is the NP "事故原因/accident cause".

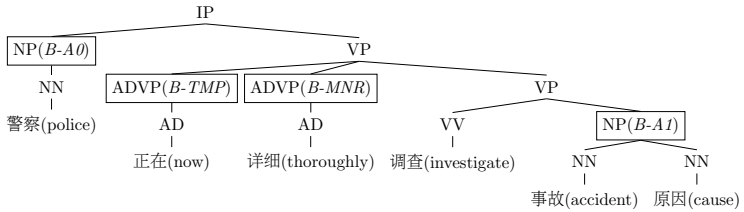


Figure 4: An example sentence: *The police are thoroughly investigating the cause of the accident*.

4.3 Results

Experimental results are presented in Table 2. Besides the five test corpora, we add a column **avg.** to show the average improvements. We also add a column **Index** for score reference convenience. **SRL1** and **SRL2** are the two features described in Section 2.2. **SC** is the syntactic cohesion model described in Section 3.

From Table 2 we see that our proposed feature **SRL1** is able to improve the baseline by 0.57 BLEU and 0.71 TER. **SRL2** improves the baseline by 0.75 BLEU and 0.77 TER. When **SRL1** and **SRL2** are used together, further improvements have been observed. In general, the semantic analysis information improves the baseline system by 0.93 BLEU and 0.98 TER (line 4 and 9). For the comparison with **SC** model, we see their performance is very close. Both **SRL1+SRL2** and **SC** refine the baseline. **SRL1+SRL2** is slightly better at BLEU (compare line 4 and 5) while **SC** is slightly better with TER (compare line 9 and 10).

SRL abstracts important event structures away from syntactic parses. Compared to full parsing model **SC**, **SRL1+SRL2** only concentrates on modeling the skeleton of a sentence, and provide much less information. Nevertheless, our experiments suggest that SRL achieves an equivalent contribution (as constraints) to a phrase-based MT system.

5 Previous Work

(Wu and Fung, 2009) propose a Hybrid two-pass model to use semantics for SMT. The first pass is a conventional phrase-based SMT decoder. The second pass generate a set of candidate re-ordered translation hypotheses by iteratively moving constituent phrases whose predicate or semantic role label was mismatched to the source sentence. The output of the second pass

Systems	NIST02	NIST03	NIST04	NIST05	NIST08	avg.	Index
			BLEU scores				
baseline	33.60	34.29	35.73	32.15	26.34	-	1
baseline+SRL1	34.50	34.76	36.21	32.75	26.77	0.57	2
baseline+SRL2	34.73	34.88	36.60	32.83	26.82	0.75	3
baseline+SRL1+SRL2	35.05	34.93	36.71	33.22	26.89	0.93	4
baseline+SC	34.96	34.52	36.37	33.35	26.90	0.79	5
			TER scores				
baseline	61.36	60.48	59.12	60.94	65.17	-	6
baseline+SRL1	60.44	59.58	58.61	60.01	64.88	0.71	7
baseline+SRL2	60.28	59.49	58.14	60.20	65.11	0.77	8
baseline+SRL1+SRL2	60.05	59.55	58.14	59.69	64.74	0.98	9
baseline+SC	59.90	59.37	58.27	59.69	64.44	1.08	10

Table 2: Experimental results

is the re-ordered translation hypothesis with the maximum match of semantic predicates and arguments. In essence, the paper proposes a semantics-based postprocessing or a semantics-based post reordering technique, where the usage of semantic information is limited due to the fact that the lexical choice has been fixed. The second pass can only do some permutation of the output of the phrase-based decoder.

(Liu and Gildea, 2010) implement two semantic role features in their tree-to-string machine translation system. First feature is named semantic role reordering which describes reordering of the source side semantic roles in the target side. Second feature is named deleted role which can penalize the deletion of source side semantic roles. They show improvement over a small FBIS English-to-Chinese system.

(Gao and Vogel, 2011) present an approach of utilizing target side SRL information to improve the hierarchical phrase-based machine translation. They extract SRL-aware Synchronous Context-Free Grammar (SCFG) rules together with conventional Hiero rules. Instead of adding additional features in the decoder, special conversion rules are applied during rule extraction procedure to ensure that when SRL-aware SCFG rules are used in derivation, the decoder only generates hypotheses with complete semantic structures.

(Baker et al., 2012) build a modality/negation (MN) annotation scheme in the target side. They define modality as an extra-propositional component of meaning and negation as an inextricably intertwined component of modality. A MN lexicon is created in a semi-supervised way. Using this lexicon a MN tagger is designed to identify the substrings that are related to MN. After the target side MN annotation, they incorporate the MN into a small Urdu-English translation task. Their baseline is a Syntax Augmented Machine Translation (SAMT) system. By using a tree grafting approach, the original syntactic tags enriched with new MN annotations are assigned to the parse trees of target side training data.

The main characteristics (also the differences to the above work) of this paper are as follows: we utilize the source side SRL information as soft constraints to improve the phrase-based machine translation system; the two models/features are implemented in the log-linear framework so that SRL information can directly act on the translation process; the experiments are carried out with a large scale Chinese-to-English NIST task; a comparison with (Cherry, 2008) which

uses dependency tree as soft constraints has been conducted.

6 Conclusion

In this paper, we proposed a method to utilize the source side semantic analysis for SMT. Two models have been created. The first model SRL1 captures the global semantic dependency in the source sentence; the second model SRL2 makes sure that the local semantic coherence is kept during search. From SMT perspective, SRL1 is a long distance reordering model; SRL2 is a local reordering model and a phrase model (it encourages the decoder to choose those phrases that keep semantic coherence). The SRL is able to improve the baseline 0.93 BLEU and 0.98 TER. We also did comparative study with SC model. Results show that the performance of the two methods is similar. SRL is 0.14 BLEU better than SC while SC is 0.1 TER better than SRL.

Acknowledgments

This work was partly supported by the Defense Advanced Research Projects Agency (DARPA) under Contract No. 4911028154.0. Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the Defense Advanced Research Projects Agency (DARPA). This work was also partly supported by National High-Tech R&D Program (2012AA011101).

The authors would like to give special thanks to anonymous reviewers for their valuable comments and suggestions.

References

Baker, K., Bloodgood, M., Dorr, B. J., Callison-Burch, C., Filardo, N. W., Piatko, C., Levin, L., and Miller, S. (2012). Modality and negation in simt use of modality and negation in semantically-informed syntactic mt. *Comput. Linguist.*, 38(2):411–438.

Cherry, C. (2008). Cohesive phrase-based decoding for statistical machine translation. In *Proceedings of ACL-08: HLT*, pages 72–80, Columbus, Ohio.

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

Gao, Q. and Vogel, S. (2011). Utilizing target-side semantic role labels to assist hierarchical phrase-based machine translation. In *Proceedings of the Fifth Workshop on Syntax, Semantics and Structure in Statistical Translation*, SSST-5, pages 107–115, Stroudsburg, PA, USA. Association for Computational Linguistics.

Gildea, D. and Jurafsky, D. (2002). Automatic labeling of semantic roles. *Computational Linguistics*, 28:245–288.

Koehn, P., Och, F. J., and Marcu, D. (2003). Statistical phrase-based translation. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology - Volume 1*, NAACL '03, pages 48–54, Stroudsburg, PA, USA. Association for Computational Linguistics.

Levy, R. and Manning, C. D. (2003). Is it harder to parse chinese, or the chinese treebank? In *Proceedings of ACL-03*, pages 439–446, Sapporo, Japan.

Liu, D. and Gildea, D. (2010). Semantic role features for machine translation. In *Proceedings of the 23rd International Conference on Computational Linguistics*, COLING '10, pages 716–724, Stroudsburg, PA, USA. Association for Computational Linguistics.

Och, F. J. and Ney, H. (2002). Discriminative training and maximum entropy models for statistical machine translation. In *Proceedings of ACL-02*, pages 295–302, Philadelphia, Pennsylvania, USA.

Och, F. J., Tillmann, C., and Ney, H. (1999). Improved alignment models for statistical machine translation. In *Proc. of the Joint SIGDAT Conf. on Empirical Methods in Natural Language Processing and Very Large Corpora (EMNLP99)*, pages 20–28, University of Maryland, College Park, MD, USA.

Papineni, K., Roukos, S., Ward, T., and Zhu, W.-J. (2001). Bleu: a method for automatic evaluation of machine translation. (RC22176 (W0109-022)).

Ramshaw, L. and Marcus, M. (1995). Text chunking using transformation-based learning. In Yarowsky, D. and Church, K., editors, *Proceedings of the Third Workshop on Very Large Corpora*, pages 82–94, Somerset, New Jersey. Association for Computational Linguistics.

Snover, M., Dorr, B., Schwartz, R., Makhoul, J., Micciulla, L., and Weischedel, R. (2005). A study of translation error rate with targeted human annotation. Technical Report LAMP-TR-126, CS-TR-4755, UMIACS-TR-2005-58, University of Maryland, College Park, MD.

Stolcke, A. (2002). Srilm - an extensible language modeling toolkit. In *Proceedings of ICSLP-02*, pages 901–904, Denver, Colorado,USA.

Sun, W. (2010). Improving Chinese semantic role labeling with rich syntactic features. In *Proceedings of the ACL 2010 Conference Short Papers*, pages 168–172, Uppsala, Sweden. Association for Computational Linguistics.

Sun, W. (2012). *Learning Chinese Language Structures with Multiple Views*. PhD thesis, Saarland University.

Sun, W., Sui, Z., and Wang, H. (2008). Prediction of maximal projection for semantic role labeling. In *Proceedings of the 22nd International Conference on Computational Linguistics (Coling 2008)*, pages 833–840, Manchester, UK. Coling 2008 Organizing Committee.

Wu, D. and Fung, P. (2009). Semantic roles for smt: a hybrid two-pass model. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics, Companion Volume: Short Papers*, NAACL-Short '09, pages 13–16, Stroudsburg, PA, USA. Association for Computational Linguistics.

Xue, N. (2008). Labeling Chinese predicates with semantic roles. *Computational Linguistics*, 34:225–255.

Zens, R., Och, F. J., and Ney, H. (2002). Phrase-based statistical machine translation. In *German Conference on Artificial Intelligence*, pages 18–32. Springer Verlag.

Comparing taxonomies for organising collections of documents

Samuel Fernando[1] *Mark Hall*[1,2] *Eneko Agirre*[3]
Aitor Soroa[3] *Paul Clough*[2] *Mark Stevenson*[1]
(1) Department of Computer Science, Regent Court, 211 Portobello, Sheffield, S1 4DP, UK
(2) Information School, Regent Court, 211 Portobello, Sheffield, S1 4DP, UK
(3) Universidad del País Vasco, Barrio Sarriena s/n, 48940 Leioa, Bizkaia
{s.fernando,m.mhall,p.d.clough,m.stevenson}@sheffield.ac.uk
{e.agirre, a.soroa}@ehu.es

ABSTRACT
There is a demand for taxonomies to organise large collections of documents into categories for browsing and exploration. This paper examines four existing taxonomies that have been manually created, along with two methods for deriving taxonomies automatically from data items. We use these taxonomies to organise items from a large online cultural heritage collection. We then present two human evaluations of the taxonomies. The first measures the cohesion of the taxonomies to determine how well they group together similar items under the same concept node. The second analyses the concept relations in the taxonomies. The results show that the manual taxonomies have high quality well defined relations. However the novel automatic method is found to generate very high cohesion.

TITLE AND ABSTRACT IN BASQUE

Dokumentu bildumak antolatzeko taxonomien arteko alderaketa

Dokumentu bildumak kategorietan sailkatzea oso erabilgarria da, dokumentuak arakatzeko eta aztertzeko aukera berriak eskaintzen duen heinean. Hori horrela izanik, dokumentu bilduma handiak sailkatzeko taxonomien behar handia dago. Artikulu honetan eskuz sortutako lau taxonomia aztertzen dira, taxonomiak automatikoki sortzen dituzten bi metodorekin batera. Taxonomia hauek ondare kulturaleko bilduma handiak antolatzeko erabili ditugu. Taxonomien ebaluazioa egin dugu galdetegietan oinarritutako bi metodo erabiliaz. Lehenbizikoak taxonomiaren kohesioa neurtzen du, hau da, antzeko itemak kontzeptu beraren azpian zein ondo taldekatzen diren. Bigarrenak taxonomiako kontzeptuen arteko erlazioak aztertzen ditu. Emaitzek erakusten dute eskuzko taxonomien erlazioen kalitatea, baina metodo automatiko berri batek lortzen du kohesio handiena.

KEYWORDS: Semantic network, taxonomy, hierarchy, Wikipedia, WordNet, ontology.

KEYWORDS IN BASQUE: Sare semantiko, taxonomia, hierarkia, Wikipedia, WordNet, Ontologia.

Proceedings of COLING 2012: Technical Papers, pages 879–894,
COLING 2012, Mumbai, December 2012.

1 Introduction

With increasingly large sets of diverse collections of documents available online a key challenge is organising and presenting these items effectively for information access. To enable the navigation and exploration of collections, content providers typically provide users with free-text search functionalities, along with some form of browsable subject categories or taxonomy, also useful in organising documents. Providing multiple mechanisms for accessing documents enables users to conduct various modes of information seeking activity, from locating specific documents to more exploratory forms of searching and browsing behaviour (Hearst, 2009; Marchionini, 2006; Wilson et al., 2010).

In this paper we focus on evaluating different taxonomies that could be used to organise and navigate content from Europeana[1], an online cultural heritage collection. This collection comprises many subcollections taken from different providers, and thus contains a very diverse set of cultural heritage *items*[2]. Some of the subcollections are linked to bespoke taxonomies; however, many are not. This therefore represents a very challenging dataset to organise in a consistent and uniform manner. We focus on two main approaches for organising content: the first is to map items from Europeana onto existing manually-created taxonomies; the second is to use data-driven approaches to automatically derive taxonomies from the collection. This requires being able to successfully group items into categories and generate suitable category labels. A note on definitions: the term 'taxonomy' is used in this paper as a general term meaning a conceptual hierarchy. A taxonomy does not necessarily have to be a subsumption hierarchy (where each child concept is subsumed by its parent concept). Some of the taxonomies described here are subsumption hierarchies and some are not.

There are many different ways of evaluating taxonomies (Snow et al., 2004; Malaisé et al., 2006; Yi, 2008; Nikolova et al., 2010; Ponzetto and Strube, 2011). Here we focus on two approaches which capture different qualities of the taxonomies. The first evaluation measures the *cohesion* of the taxonomies; how well they group together similar items into the same concept node. The second analyses the *relationships* between concept nodes in the taxonomies and whether people can understand the concept labels. We believe this is the first time that such evaluations have been applied to such taxonomies over large collections of data items.

This paper provides three main contributions: (1) the systematic comparison of different taxonomies for organising a large cultural heritage collection; (2) a novel data-driven approach based on using Wikipedia article links as concept nodes in the taxonomy; and (3) the evaluation of cohesion and relationship type between concepts using an approach based on crowdsourcing. The rest of the paper comprises the following. Section 2 describes related work in this area. Section 3 gives a brief overview of the key data resources and tools referenced and Section 4 describes the taxonomies and item-to-resource mapping approaches used in the experiments. Section 5 describes the cohesion and relatedness experiments and the results obtained.

2 Related work

There are many category systems or taxonomies available, some of which are domain specific; others aimed at covering more general subjects. For example, one of the most popular and commonly used resources is the Library of Congress Subject Headings (LCSH)[3]. LCSH provides

[1] `http://www.europeana.eu`

[2] An item is defined here as a online record of a cultural heritage artifact (usually an image), together with associated metadata, such as title, subject, description etc.

[3] `http://www.loc.gov/aba/cataloging/subject/`

a controlled vocabulary of keywords (or subject headings), which are widely used in libraries to catalogue materials and facilitate information access. Similarly, in the medical domain MeSH[4], created and maintained by the National Library of Medicine, provides a controlled vocabulary of medical subject headings. In computational linguistics, WordNet (Fellbaum, 1998) is a commonly used lexical knowledge base that links concepts in various ways. WordNet has been expanded with WordNet domain labels which group together words from different syntactic categories and different senses (Bentivogli et al., 2004). These domain labels are organised into a hierarchical structure.

There is a body of previous work on automatically deriving taxonomies and relations from free text. Hearst (1992) was perhaps the earliest significant effort to derive hyponym-hypernym relations from free text using the now eponymous Hearst patterns which code common syntactic forms of the hyponymy pattern (e.g. 'Vehicles such as Cars'). These patterns were hand-crafted. More recently Snow et al. (2004) developed on this work by using an existing knowledge base to automatically derive lexico-syntactic patterns containing the hyponym-hypernym pairs.

An alternative to creating hierarchies of concepts from the pattern-based methods is to use statistical methods. Sanderson and Croft (1999) used an approach to automatically build a hierarchy of terms or concepts (nouns and noun phrases) based on term co-occurrences within a set of documents. To order the concepts a statistical relation called *subsumption* was used to determine which of a co-occurring pair of concepts was most likely to be the parent. Griffiths and Tenenbaum (2004) uses Bayesian methods and LDA (Latent Dirichlet Allocation) to derive topic clusters and create a topic hierarchy. Recent work has exploited the information in Wikipedia to create taxonomies. Ponzetto and Strube (2011) uses the category hierarchy along with lexical matching methods to create a taxonomy which compares well with manually created resources. DBpedia (Auer et al., 2007) also uses the Wikipedia category hierarchy but also additionally links in the articles into the hierarchy.

One key problem is how to evaluate taxonomies effectively. This is a complex problem since there are many aspects to consider. We can consider how users would rate various aspects of their experience using a questionnaire. However this subjective study can be misleading since people can underrate or overrate their experience. A more objective approach is to log user interactions and then infer from these how effective the taxonomy is i.e. time spent on a task, how much of the domain was covered and so on. There have been user studies of taxonomies which have used combinations of both of these approaches (Malaisé et al., 2006; Yi, 2008; Nikolova et al., 2010). There is also an important distinction to be made between the data content of a taxonomy and the methods used for visualisation. There have been user studies which have focussed on evaluation different visualisations while keeping the data constant (Katifori et al., 2007). In this paper we are considering only the data content, the concepts and relations, so visualisations are kept constant.

User studies are certainly important. However these studies often don't answer certain questions about the taxonomy. We can measure aspects of their experience but we might not have a fine-grained understanding of which aspects of the taxonomy were positively or negatively perceived. Also such studies require users to be physically present at a machine specially set up for logging. This can make such studies expensive and time-consuming. A different approach is to use intrinsic evaluations. Yu et al. (2007) present a wide range of ontology evaluation approaches as applied to variations of the Wikipedia category structure such as

[4] http://www.nlm.nih.gov/mesh/

fanout, tangledness, relationship richness, class richness, importance connectivity and cohesion. Cohesion attempts to measure the degree to which similar items are clustered together under a single node in the taxonomy. This was initially proposed by (Boyd-Graber et al., 2009) and has also recently been applied to cultural heritage (Hall et al., 2012).

Another evaluation is to measure the accuracy of the child-parent pairs from the taxonomies by asking evaluators to classify them as either *isa* or *notisa* relations (Snow et al., 2004; Ponzetto and Strube, 2011). A similar method is used here but expanded to give a more detailed understanding of the types of the relations found in the different taxonomies.

3 Resources and tools

This section lists some key resources and tools that are used in this paper.

3.1 Wikipedia Miner

The Wikipedia Miner (Milne and Witten, 2008) tool is used in this paper both as a tool to help map items into existing taxonomies and as a way to generate a novel taxonomy from scratch. Wikipedia Miner is a Wikification tool which adds inline links to Wikipedia articles into free text. The software is trained on Wikipedia articles, and thus learns to disambiguate and detect links in the same way as Wikipedia editors. Disambiguation of terms within the text is performed first. A machine-learning classifier is used with several features. The main features used are *commonness* and *relatedness*, as in Medelyan et al. (2008). The commonness of a target sense is defined by the number of times it is used a destination from some anchor text e.g. the anchor text 'Tree' links to the article about the plant more often than the mathematical concept and is thus more common. Relatedness gives a measure of the similarity of two articles by comparing their incoming and outgoing links. The performance achieved using their approach is currently state of the art for this task. The Wikipedia Miner software is freely available[5].

3.2 Europeana cultural heritage collection

Cultural heritage items from Europeana are used for the evaluation. Europeana is a large online aggregation of cultural heritage collections from across Europe. In this paper a snapshot of the English subset of the data from March 2011 is used. This comprises 547780 items in total. Each item consists of an XML metadata record. This comprises a number of fields the most informative of which are `dc:title`, `dc:subject`, `dc:description` which contain the title, subject keywords and a textual description of the item. About 74% of the items have an associated image which is displayed on the portal website. A difficulty with this collection is that a significant number of the items have very little associated metadata. In the worst case some items have only a one-word title, with no subject or description. This problem is dealt with implicitly in the methods described below, where such sparse records are effectively filtered out in the mapping and taxonomy generation stages and are not included in the evaluations.

4 Taxonomies and mappings

Six taxonomies were tested in these experiments. Four of these were based on existing taxonomies which have been mostly manually created: the Library of Congress Subject Headings, WordNet Domains, Wikipedia Taxonomy and DBpedia. The remaining two taxonomies were automatically derived from the metadata present for the items in the collections: WikiFreq and

[5] `http://wikipedia-miner.cms.waikato.ac.nz/`

LDA topics. This section gives a description of each of these taxonomies and how the items in the collection were mapped into them. Statistics for each taxonomy are presented at the end of this section.

4.1 Manually created taxonomies

4.1.1 Library of Congress Subject Headings (LCSH)

The LCSH comprises a controlled vocabulary maintained by the United States Library of Congress for use in bibliographic records. They are used in many libraries to organise their collections as well as for organising materials online.

The text from the `dc:subject` field in the Europeana item are used for the mapping. The text is lemmatized using Freeling (Padró, 2011). The text is compared to the category labels for the LCSH concepts. If the text contains any of the category labels then the item is matched to these categories. If more than one matching label is found, then the longest matching label is used for the mapping.

4.1.2 WordNet domains

The WordNet hierarchy is a fine-grained classification which is too detailed for browsing, with more than a hundred thousand nodes and concepts like *entity, natural phenomenon* and *body of water*. Instead *WordNet domains* organises the WordNet concepts into a smaller hierarchy, with only 164 domain labels which are easily understood by a general user. The domain labels have been semi-automatically applied to each of the synsets in WordNet. Each synset is annotated with each one label from a set of about two hundred. The information provided by the domain labels is complementary to the data existing already in WordNet. The domain labels group together words from different syntactic categories (e.g. nouns and verbs), and also may group together different senses of the same word and thus reduce polysemy.

WordNet domains lists the domain labels for all open class words in WordNet, but it only contains a few proper nouns. Given the large concentration of proper nouns in Europeana, we extend the list of words using Yago2. Yago2 (Hoffart et al., 2011) is a knowledge base derived from Wikipedia with more than 10 million entities, and each entity in Yago2 is linked to a WordNet 3.0 synset. We also used a mapping from WordNet 3.0 synsets to WordNet Domain labels as provided by the Multilingual Central Repository (MCR) (Atserias et al., 2004). To perform the mapping, the first step is again to use the `dc:subject` field to link Europeana items to Yago2 entities (using lemmatization and finding the longest possible match). These are then mapped to the WordNet Domain labels via the Yago2 entity-to-synset and the MCR synset-to-WordNetDomain mappings.

4.1.3 Wikipedia Taxonomy

Wikipedia Taxonomy (Ponzetto and Strube, 2011) is a taxonomy derived from Wikipedia categories. The authors create the Wikipedia Taxonomy by keeping the isa relations between Wikipedia categories and discarding the rest. We first apply Wikipedia Miner (see Section 3.1) over the Europeana items to find the relevant Wikipedia entities in the the `dc:subject` field. Then, we link the Europeana item to all Wikipedia Taxonomy categories which are related to these entities.

4.1.4 DBpedia ontology

The DBpedia ontology (Auer et al., 2007) is a small, shallow ontology manually created based on information derived from Wikipedia. Contrary to the previous vocabularies described above, the DBpedia ontology is a formalised ontology, including inference capabilities. The authors provide the instances of each ontology class, i.e. the set of Wikipedia entities pertaining to this class. For mapping Europeana items to DBpedia ontology classes, we first apply Wikipedia Miner to find the relevant Wikipedia entities to the item, and then link the item to the classes these entities belong.

4.2 Automatically created data-driven taxonomies

4.2.1 LDA topic modelling

Latent Dirichlet Allocation (LDA) is a state-of-the-art topic modelling algorithm, that creates a mapping between a set of topics T and a set of items I, where each item $i \in I$ is linked to one or more topics $t \in T$. Each item is input into LDA as a bag-of-words and then represented as a probabilistic mixture of topics. The LDA model consists of a multinomial distribution of items over topics where each topic is itself a multinomial distribution over words. The item-topic and topic-word distributions are learned simultaneously using collapsed Gibbs sampling based on the item - word distributions observed in the source collection (Griffiths and Steyvers, 2004). LDA has been used to successfully improve result quality in Information Retrieval (Azzopardi et al., 2004; Wei and Croft, 2006) tasks and is thus well suited to support exploration in digital libraries.

To turn the flat LDA topic model into a navigable hierarchy, Griffiths and Tenenbaum (2004) describe a hierarchical LDA approach. However this was found to be prohibitively time consuming given our large data-set. Instead a recursive divide and conquer approach was used, which was much more efficient. The number of topic groups at each stage was limited to a maximum of 9 to make the hierarchy manageable for users to navigate. The algorithm is outlined below.

1) Run LDA over the corpus to determine the document-topic probabilities. The number of topics $topic\_n$ to generate is automatically determined using this equation:

$$topic\_n = \min(9, \frac{\left|documents\_in\_corpus\right|}{30}) \quad (1)$$

2) For each topic used the document-topic probabilities that LDA outputs to identify the set of documents associated with that topic. Each document is assigned only to its highest-probability topic. While this removes some of the power inherent in the LDA topic model, we believe that from a navigational perspective it is better if each document is located at only one point in the hierarchy and not at multiple points. To give each topic a label, we simply selected the highest-probability word from each topic's topic-word distribution.

3) If a topic set has less than 60 items then stop. Otherwise go back to 1) using the set of items identified in 2) as the corpus.

4.2.2 Wikipedia link frequencies

This is a novel method for taxonomy creation which uses Wikipedia article links as the concept nodes in the taxonomy. The first step is to add inline article links to all the item texts in the collection using Wikipedia Miner (see Section 3.1). A confidence threshold of 0.5 was used to help ensure the links were of high quality - that is they are correctly disambiguated and relevant to the topic of the item.

The first step is to find the frequency counts of all article links that occur in the items. Let L be the set of all links found in the items. Then the frequency function $F : L \rightarrow \mathbb{N}$ gives the global frequency count for occurrences of the link in all items.

The following procedure is then used for each item to create and populate the taxonomy. Let $S \subset L$ be the set of links found in that item. The links are ordered in S by order of frequency according to the F function (most frequently occuring first) to give an ordered list of links $a_1, a_2, a_3 \cdots a_n$. The item is then inserted into the tree under the branch $a_1 \rightarrow a_2 \rightarrow a_3 \cdots \rightarrow a_n$, with a_1 at the top level in the tree and the item appearing under the node a_n. If this branch does not already exist in the tree then it is created.

It was found that using this approach the branching factor was very high at some levels so the number of child nodes at each level was limited to at most 20. Furthermore only items with at least 2 links were used to prevent a large number of single-linked items appearing at the top level. Concepts with less than 20 items were also filtered out.

This method is labelled WikiFreq in the remainder of the paper.

4.3 Taxonomy statistics

Table 1 shows some statistics for each taxonomy:

- The number of items that are mapped into the taxonomy.
- The average number of parents for each item.
- The average depth from the root node to an item.
- The number of top level nodes in the taxonomy.

A problem with some of the manual taxonomies is the very high number of top level nodes, which makes it difficult for users to browse. However there is no obvious way to select suitable top level nodes in these taxonomies. Additionally some of the taxonomies assign items to many parent nodes - this means that the data is repeated across the taxonomy. This is not a problem in itself, but is likely to mean that items may often be assigned to incorrect nodes.

5 Experiments

Two evaluations were performed on the taxonomies. The first measured the cohesion of the item clustering, and the second gathered human judgements of the relations that were found between child-parent concept pairs in the taxonomy. For both evaluations online surveys were created using an in-house crowdsourcing interface. Links to the surveys were sent out to a mailing list comprising thousands of students and staff members at the University of Sheffield.

Type	Taxonomy	Items	Nodes	Avg. parents	Avg. Depth	Top nodes
Manual	LCSH	99259	285238	1.8	1.97	28901
	DBpedia	178312	273	4.2	2	30
	Wiki Taxonomy	275359	121359	11.7	1.13	10417
	WN domains	308687	170	7.1	7.1	6
Automatic	LDA topics	545896	22494	1	7.3	9
	Wiki Freq	66558	502	1	3.39	24

Table 1: Statistics for each taxonomy

5.1 Cohesion

A cohesive cluster is defined as one in which the items are similar while at the same time clearly distinguishable from items in other clusters (Tan et al., 2006). To measure the cohesiveness of the taxonomies we use the *intruder detection* task originally devised in Boyd-Graber et al. (2009) and recently used for cultural heritage items in Hall et al. (2012). The idea of this is to present 5 items to an evaluator. Four of these are taken from one concept node in the taxonomy and the other (the *intruder*) is randomly picked from elsewhere in the taxonomy. The more cohesive the concept in the taxonomy the more obvious it should be which is the intruder item. Each *unit* was displayed as a list of five images along with the titles of the items. An example[6] of a cohesive unit is shown in Figure 1. To generate good quality units for the evaluation the *informativeness* of items was calculated as follows:

$$\begin{aligned} informativeness(item) = \quad & length(item_{title})/avgL_{title} * log(N/count(item_{title})) \\ & + length(item_{desc})/avgL_{desc} * log(N/count(item_{desc})) \\ & + length(item_{subj})/avgL_{subj} * log(N/count(item_{subj})) \end{aligned}$$

where $title, desc, subj$ refer to the title, description and subject fields of the metadata, $avgL_X$ is the average length of field X over the whole collection, $length(item_X)$ is the length of that item field text, and $count(item_X)$ gives the frequency of that item field text over the whole collection. The higher the resulting value the more informative the item is. Note that as well as taking into account the length of the fields, this also weights by the inverse document frequency (idf) value, so very frequently occuring terms will be downweighted.

The most informative items were selected for each category. This helped to ensure that the users were presented with informative items, allowing them to have enough information to decide which was most likely to be the intruder. The same also applies to the taxonomy mappings being evaluated; it is difficult to correctly map items with very little information in the metadata. The procedure for selecting the sample units was as follows:

[6] Images reproduced from Wikipedia and subject to relevant licenses.
http://en.wikipedia.org/wiki/File:York_Minster_close.jpg.
http://en.wikipedia.org/wiki/File:Wells_Cathedral,_Wells,_Somerset.jpg.
http://en.wikipedia.org/wiki/File:West_Side_of_Westminster_Abbey,_London_-_geograph.org.uk_-_1406999.jpg.
http://en.wikipedia.org/wiki/File:Goatfell_from_Brodick_Harbour.jpg
http://en.wikipedia.org/wiki/File:Sfec-durham-cathedral-2007-263.JPG

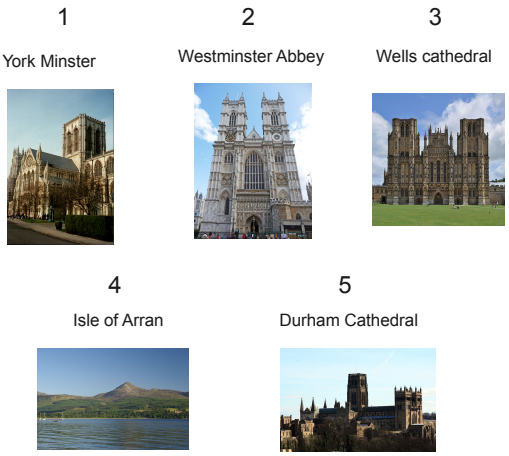


Figure 1: Example of a cohesive unit. Here the intruder item is number 4.

1. Select categories at random that have at least 4 items.

2. For each category:

   (a) Return the 4 items in the category which were most informative.

   (b) To find the intruder item, select 100 items at random from the whole collection and return the most informative item.

Six units were shown on each page, one of which was always a control unit. The control units were manually chosen to be examples where the intruder was very obvious. The purpose of the control units was to ensure the quality of the judgements, since if a participant got the control unit wrong it was an indication that they were not taking the task seriously.

Thirty non-control units were created from each taxonomy. Altogether 134 people attempted the survey. 23 of the users evaluated at least one control unit wrong, or evaluated less than 5 units in total, and so their answers were excluded. The remaining 111 participants contributed a total of 1255 answers. Each unit received a minimum of 5 answers and an average of 6.97 answers. A unit was judged as cohesive if more than 80% of the annotators agreed on the same intruder.

Type	Taxonomy	Coherent units	Percentage
Manual	LCSH	19	63.3
	DBpedia	17	56.7
	Wiki Taxonomy	18	60.0
	WN domains	15	50.0
Automatic	LDA topics	17	56.7
	Wiki Freq	**29**	**96.7**

Table 2: Number of coherent units (out of 30) for each of the taxonomies.

The results (Table 2) show that most of the taxonomies achieved roughly the same level of cohesion for the clusters, roughly between 50 and 63%. However the WikiFreq taxonomy performed far better, with only one unit of the 30 judged as not coherent. This success shows that the Wikipedia links are very effective as a means of grouping together similar items. This might be explained by considering that items grouped together under the same node will share a number of keywords which link to the same Wikipedia articles which would ensure that the items are very similar. In contrast the Wikipedia taxonomy and DBpedia ontology use categories rather than articles in Wikipedia as the concept nodes. These are much more loosely defined; each article in Wikipedia can belong to many categories and each category contains many articles. The results also indicate that Wikipedia articles as entities are much more clearly defined than the LDA topic keywords and thus work much better at grouping together the similar items.

5.2 Relation classification

Previous work has evaluated taxonomies by presenting child-parent pairs of concept nodes to evaluators and asking them a simple boolean question - does the pair represents a valid hypernymic relation, i.e. is it true that "ChildNode isa ParentNode"? (Ponzetto and Strube, 2011; Snow et al., 2004). We would expect the manually created taxonomies to perform well here. The automatic methods also intend to create a hierarchical structure, with more general concepts at the top nodes going to more specific in the lower nodes.

Here we conduct a slightly deeper analysis of what kinds of relations were present in these taxonomies. Instead of a simple boolean question a two-part question was used. Given a child-parent pair A, B the evaluators were asked two questions:

1. Are the two concepts A and B related? (Yes/No/I don't know) The evaluators were asked to judge the relation within the context of the cultural heritage taxonomy. A positive example was presented: Westminster and London, which were related because Westminster is in London. A negative example was Fish and Bicycle which were unrelated and would not be a useful pair to include in a taxonomy.

2. If Yes, then how would you best define the relationship? Is A more specific than B, less specific than B, neither, or don't know? Examples were also given to help with this question. Westminster is *more* specific than London since Westminster is within London. The term Scientist is less specific than Physicist, since while all Physicists are Scientists, not all Scientists are Physicists (they could be biologists or chemists for example). For

Type	Taxonomy	Child (A)	Parent (B)
Manual	LCSH	Work	Human Behaviour
		Braid	Weaving
		Time	Geodetic Astronomy
	DBpedia	Mountain Range	Place
		Fern	Plant
		Congressman	Politician
	Wiki Taxonomy	Mammals of Africa	Wildlife of Africa
		Schools in Wiltshire	Schools in England
		British Culture	European Culture
	WN domains	vehicles	transport
		mechanics	engineering
		home	applied science
Automatic	LDA topics	earthenware	dish
		view	church
		tunnel	chapel
	Wiki Freq	Corrosion	Coin
		Interior Design	Industrial Design
		Towpath	Waterscape

Table 3: Some examples of child-parent pairs from each taxonomy.

the 'neither' option consider Physicist and Biologist. The concepts are related (both are scientists) but neither is more specific than the other.

Forty non-control pairs from each taxonomy were presented to the evaluators giving a total of 240 pairs. Examples of concept pairs from each taxonomy are shown in Table 3. As for the previous experiment control pairs were manually identified where the answer should be obvious. Five pairs were shown on each page of which one was always a control pair.

Altogether 270 people attempted this survey. 97 people evaluated more than half the control pairs wrong or evaluated less than 5 pairs in total, and so their answers were excluded. Of the 173 remaining participants, a total of 3826 evaluations were made for each pair. A minimum of 8 evaluations and an average of 15.94 evaluations were made for each instance.

Type	Taxonomy	Yes	No	Don't know	Agreement
Manual	LCSH	74.2	8.8	17.0	79.1
	DBpedia	86.6	11.2	2.2	88.4
	Wiki Taxonomy	**96.1**	1.7	2.3	**95.9**
	WN domains	77.1	14.5	8.4	83.9
Automatic	LDA topics	30.3	**50.3**	19.3	71.6
	Wiki Freq	47.6	16.5	**35.8**	70.9

Table 4: Are *A* and *B* related?

The results for the relatedness question (Table 4) show a clear pattern. The manually created taxonomies are markedly more likely to contain clearly related pairs of concepts. The Wikipedia

taxonomy hierarchy is the highest performing in this regard which suggests that the user-created category hierarchy is of high quality and has easily understood concepts and relations. DBpedia scores slightly lower. This difference might be explained due to DBpedia containing article entities as well as categories. The lower score suggests either that articles are not always placed in the best categories, or that it is harder for users to identify article-category relationships. WordNet domains scores lower. This may be due to the domain concepts being sometimes quite general and possibly harder for general users to understand (for example one pair of concepts was 'color' and 'factotum'). LCSH scored surprisingly low considering that it is a manually created taxonomy. This suggests that the concepts and relations in this hierarchy are even harder for users to understand or identify. Finally the two data derived taxonomies score lower still. For the WikiFreq taxonomy a high percentage of the relations were classified as 'Don't know'. This may be because a high number of the article links are about quite obscure concepts which most people would not know about. Finally the LDA topics produced the highest number of definite 'No' judgements which shows that the taxonomy may be difficult or confusing for users to navigate.

Type	Taxonomy	$A < B$	$A > B$	Neither	Don't know	Agreement
Manual	LCSH	65.4	8.7	23.4	2.5	68.7
	DBpedia	76.2	4.9	18.1	0.7	78.9
	Wiki Taxonomy	**78.3**	4.7	16.0	0.9	**82.8**
	WN domains	63.6	6.3	28.0	2.0	67.6
Automatic	LDA topics	21.4	14.8	**62.1**	1.6	61.0
	Wiki Freq	30.9	**22.6**	43.6	**2.9**	67.0

Table 5: Specificity of the pairs, with A the child node, and B the parent node. $A < B$ means A is more specific than B.

The results for the specificity question are shown in Table 5. These follow a roughly similar pattern to the relatedness. The $A < B$ case is the most desirable for the taxonomies since we would prefer the most general concepts at the top of the hierarchy narrowing down into more specific concepts. The Wikipedia taxonomy and DBpedia both score relatively highly here, although both contain a surprisingly high number of cases where neither A or B was identified as more specific than the other (16.0 and 18.1% respectively). For the Wikipedia taxonomy this shows that although almost all child-parent pairs are considered to be related concepts, they are not always easily identified with the child as more specific than the parent. Both WordNet domains and LCSH fare worse, again with more relations identified as 'neither'. The WikiFreq taxonomy contains a more mixed set of results with quite a high proportion of relations the 'wrong way round' with A deemed to be less specific than B, although the highest number falls into the 'neither' category. This result is a reflection of the nature of the links within the items. The taxonomy is ordered with the most frequent occuring links at the top going down towards the least. Clearly this is not enough to create the kind of general-to-specific relationships which are desirable. Finally the results for the LDA topics show that the majority are defined as 'neither' - the concepts are topically related but mostly without any specificity ordering.

Conclusion and perspectives

Developing effective taxonomies for the purpose of organising large number of data items is a complex task. Existing manually created taxonomies might be accurate and well structured but may not be adequate for a specified domain or may be hard for users to navigate. Automatic methods for deriving taxonomies have the advantage of closely reflecting the nuances of the data - but organising the derived concepts into meaningful relations remains a problem.

The experiments in this paper shows some surprising results. The LCSH taxonomy has been manually created for the purpose of organising library collections and so might be the obvious choice to organise CH data online. However the results show that the relations within LCSH are defined less clearly than that of the Wikipedia derived taxonomies. WordNet domains performs at a similar level to the LCSH in terms of the quality of the relations. The LDA topic hierarchy gave poor results in terms of the identified topics. The topic pairs were often unrelated, and had no general-to-specific structure as would be desirable for this application.

The WikiFreq hierarchy performed slightly better in this regard. Just over half the concept pairs were judged to be related. Just under a third were labelled as 'Don't know' which may reflect the obscurity of the concept nodes identified. It was hoped that organising the frequency counts of the links would organise the hierarchy into a general-to-specific direction. This was not achieved, although the hierarchy does have the benefit of providing the user with an overview of the collection by immediately seeing which kind of items are most prevalent.

In terms of cohesion all the taxonomies achieved similar results except for the WikiFreq taxonomy which achieved almost perfect cohesion. This shows how effective the Wikipedia links are in grouping together similar items.

It is also worth noting that WikiFreq and LCSH map significantly fewer items into the taxonomy.

Future work will continue with different evaluation approaches, such as domain/task coverage and accuracy of the mappings. We also aim to expand the evaluations to include user studies; a key question is how well these taxonomies assist users when used for browsing large collections, such as Europeana. The aim is to see if there is a correlation between the intrinsic results that were found here with the extrinsic quality judgements when used in real life applications. A promising line of work will be to build on the WikiFreq approach by integrating with the high quality Wikipedia taxonomy knowledge base. The hope is that using this approach will generate highly coherent units along with a well structured conceptual tree.

Acknowledgements

The research leading to these results was supported by the PATHS project (`http://paths-project.eu`) funded by the European Community's Seventh Framework Programme (FP7/2007-2013) under grant agreement no. 270082. This research was also partially funded by the Ministry of Economy under grant TIN2009-14715-C04-01 (KNOW2 project).

References

Atserias, J., Villarejo, L., Rigau, G., Agirre, E., Carroll, J., Magnini, B., and Vossen, P. (2004). The meaning multilingual central repository. In *Proceedings of GWC*, pages 23–30.

Auer, S., Bizer, C., Kobilarov, G., Lehmann, J., Cyganiak, R., and Ives, Z. (2007). Dbpedia: A nucleus for a web of open data. *The Semantic Web*, pages 722–735.

Azzopardi, L., Girolami, M., and Van Rijsbergen, C. (2004). Topic based language models for ad hoc information retrieval. In *Neural Networks, 2004. Proceedings. 2004 IEEE International Joint Conference on*, volume 4, pages 3281–3286. IEEE.

Bentivogli, L., Forner, P., Magnini, B., and Pianta, E. (2004). Revising the wordnet domains hierarchy: semantics, coverage and balancing. In *Proceedings of the Workshop on Multilingual Linguistic Ressources*, pages 101–108. Association for Computational Linguistics.

Boyd-Graber, J., Chang, J., Gerrish, S., Wang, C., and Blei, D. (2009). Reading tea leaves: How humans interpret topic models. In *Proceedings of the 23rd Annual Conference on Neural Information Processing Systems*.

Fellbaum, C., editor (1998). *WordNet: An electronic lexical database*. MIT Press.

Griffiths, D. and Tenenbaum, M. (2004). Hierarchical topic models and the nested chinese restaurant process. In *Advances in neural information processing systems 16: proceedings of the 2003 conference*, volume 16, page 17. The MIT Press.

Griffiths, T. and Steyvers, M. (2004). Finding scientific topics. *Proceedings of the National Academy of Sciences of the United States of America*, 101(Suppl 1):5228.

Hall, M. M., Clough, P. D., and Stevenson, M. (2012). Evaluating the use of clustering for automatically organising digital library collections. In *Theory and Practice of Digital Libraries 2012*.

Hearst, M. (1992). Automatic acquisition of hyponyms from large text corpora. In *Proceedings of the 14th conference on Computational linguistics-Volume 2*, pages 539–545. Association for Computational Linguistics.

Hearst, M. (2009). *Search user interfaces*. Cambridge Univ Pr.

Hoffart, J., Suchanek, F., Berberich, K., Lewis-Kelham, E., De Melo, G., and Weikum, G. (2011). Yago2: exploring and querying world knowledge in time, space, context, and many languages. In *Proceedings of the 20th international conference companion on World wide web*, pages 229–232. ACM.

Katifori, A., Halatsis, C., Lepouras, G., Vassilakis, C., and Giannopoulou, E. (2007). Ontology visualization methods - a survey. *ACM Comput. Surv.*, 39(4).

Malaisé, V., Aroyo, L., Brugman, H., Gazendam, L., de Jong, A., Negru, C., and Schreiber, G. (2006). Evaluating a thesaurus browser for an audio-visual archive. In Staab, S. and Svátek, V., editors, *Managing Knowledge in a World of Networks*, volume 4248 of *Lecture Notes in Computer Science*, pages 272–286. Springer Berlin / Heidelberg. 10.1007/11891451_25.

Marchionini, G. (2006). Exploratory search: from finding to understanding. *Communications of the ACM*, 49(4):41–46.

Medelyan, O., Witten, I. H., and Milne, D. (2008). Topic Indexing with Wikipedia. In *Proceedings of the Association for the Advancement of Artificial Intelligence (AAAI) WikiAI workshop*.

Milne, D. and Witten, I. H. (2008). Learning to Link with Wikipedia. In *Proceeding of the 17th ACM conference on Information and Knowledge Management*, pages 509–518.

Nikolova, S., Ma, X., Tremaine, M., and Cook, P. (2010). Vocabulary navigation made easier. In *Proceedings of the 15th international conference on Intelligent user interfaces*, IUI '10, pages 361–364, New York, NY, USA. ACM.

Padró, L. (2011). Analizadores multilingües en freeling. *Linguamatica*, 3(2):13–20.

Ponzetto, S. and Strube, M. (2011). Taxonomy induction based on a collaboratively built knowledge repository. *Artificial Intelligence*, 175(9-10):1737–1756.

Sanderson, M. and Croft, B. (1999). Deriving concept hierarchies from text. In *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 206–213. ACM.

Snow, R., Jurafsky, D., and Ng, A. (2004). Learning syntactic patterns for automatic hypernym discovery. *Advances in Neural Information Processing Systems 17*.

Tan, P., Steinbach, M., Kumar, V., et al. (2006). *Introduction to data mining*. Pearson Addison Wesley Boston.

Wei, X. and Croft, W. (2006). Lda-based document models for ad-hoc retrieval. In *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 178–185. ACM.

Wilson, M., B, K., MC, S., and B, S. (2010). From keyword search to exploration: Designing future search interfaces for the web. *Foundations and Trends in Web Science*, 2(1):1–97.

Yi, M. (2008). Information organization and retrieval using a topic maps-based ontology: Results of a task-based evaluation. *Journal of the American Society for Information Science and Technology*, 59(12):1898–1911.

Yu, J., Thom, J., and Tam, A. (2007). Ontology evaluation using wikipedia categories for browsing. In *Proceedings of the sixteenth ACM conference on Conference on information and knowledge management*, pages 223–232. ACM.

Modeling the Complexity of Manual Annotation Tasks: a Grid of Analysis

Karën Fort[1,2] *Adeline Nazarenko*[1] *Sophie Rosset*[3]
(1) LIPN, Université Paris 13 & CNRS, 99 av. J.B. Clément, 93430 Villetaneuse
(2) LORIA, Campus Scientifique, 54506 Vandœuvre-lès-Nancy
(3) LIMSI-CNRS Rue John von Neumann, Université Paris-Sud 91403 Orsay
karen.fort@loria.fr, adeline.nazarenko@lipn.univ-paris13.fr,
sophie.rosset@limsi.fr

ABSTRACT
Manual corpus annotation is getting widely used in Natural Language Processing (NLP). While being recognized as a difficult task, no in-depth analysis of its complexity has been performed yet. We provide in this article a grid of analysis of the different complexity dimensions of an annotation task, which helps estimating beforehand the difficulties and cost of annotation campaigns. We observe the applicability of this grid on existing annotation campaigns and detail its application on a real-world example.

KEYWORDS: manual corpus annotation, annotation campaign management, annotation campaign cost estimate.

Proceedings of COLING 2012: Technical Papers, pages 895–910,
COLING 2012, Mumbai, December 2012.

1 Introduction

With the development of NLP applications, the annotation campaigns are becoming more numerous and more varied. Annotated corpora are used for acquiring knowledge as well as for testing theories, models and tools. They can be directly used in end-applications or for specific internal tasks.

Manual annotation is actually a widespread practice in NLP. It consists in adding labels of linguistic nature or reflecting the usage of NLP technologies on some oral or written discourse. This corresponds to a great diversity of phenomena, as these annotations vary in nature (phonetic, morpho-syntactic, semantic or task-oriented labels), in the range they cover (they can concern a couple of characters, a word, a paragraph or a whole text), in their degree of coverage (all the text is annotated or only a part of it) and in their form (atomic value, complex feature structures or relations and even cross-document alignment relations).

It has been a long road since the big pioneer annotation campaigns like the Penn Treebank (Marcus et al., 1993), but one problem remains: manual annotation is expensive. Various strategies have been implemented to reduce or control annotation costs. Tools have been developed to assist and guide the work of annotators. Automatic annotation methods, sometimes based on machine learning, have been introduced to relieve the annotator of the most trivial and repetitive work and to allow him/her to focus on the hardest annotation tasks where human interpretation is critical. For simple tasks, the use of crowdsourcing is developed with the idea of dividing up tedious work and exploiting the number of annotators to compensate for the heterogeneity of their competence and reliability. Significant efforts have also been made to develop evaluation protocols and to measure intra and inter-annotator agreements, which allow for a better control of the quality of the produced annotated data.

Despite all this work and the experience gained in annotation, we lack a global picture or an overall methodology to *a priori* determine the various costs of an annotation campaign (task definition, data preparation, recruitment and management of annotators, annotation itself, quality control, etc.), make the necessary compromises and choose the appropriate solutions to alleviate the annotators' work.

This article relies on the analysis of many annotation campaigns, which are described in the state of the art or were managed by the authors. It does not offer a tool or a ready-made solution to tell you "how to build your next annotation campaign". Instead, we propose an analytical framework, a grid of analysis, to understand the complexity of annotation tasks. It is based on a decomposition of these tasks into elementary ones and a decomposition of complexity into 6 dimensions that are orthogonal to each other except for one. We also provide concrete metrics to measure the complexity of annotation tasks, *a priori* when possible, from already annotated samples or by comparison with similar annotation tasks. Our approach is pragmatic, as our aim is to provide practical tools to analyze the complexity in all its dimensions. Obviously, it gives a simplified view of complex annotation tasks, but it enables to compare different campaigns on a common basis.

This article is as follows. Section 2 shows that the notion of complexity is present in many works in the state of the art but that no comprehensive analysis has been proposed so far. The third section presents our analysis of the complexity dimensions of annotation. Section 4 illustrates the practical benefit of this method of analysis and shows how to apply it on a complex task.

2 An important but implicit issue

If the question of the complexity of annotation tasks has deserved little attention as such, it is implicitly present in most of the issues related to annotation.

2.1 Feedback on large-scale campaigns

An effort has been made to document large-scale projects and encountered issues. For example, concerning the Penn Treebank, (Marcus et al., 1993) explains that the POS tagset has been largely reduced as compared to that of the Brown corpus, in order to eliminate the categories that could be deduced from the lexicon or the syntactic analysis. It is also noted that reducing the size of the tagset allows to reducing inconsistencies. Finally, in order to "avoid arbitrary decisions", the annotators were allowed to associate several categories to a same token. The same principle was used for the syntactic layer, allowing multiple binding in case of ambiguity. For the same reason, no distinction was made between arguments and circumstants. (Abeillé et al., 2003) presents the main issues encountered during the French Treebank annotation and the found solutions. One difficulty concerns multi-word expressions that the campaign managers finally chose to annotate both as multi-word expressions and as distinct elements to avoid "linguistic debates". Another interesting issue concerns the hierarchical structure of the tagset. It allows to simplify the annotation as most of the sub-categories end up being unambiguous once the main category is selected. In the speech domain, the MEDIA annotation campaign gave rise to an in-depth reflexion on the methodology (Bonneau-Maynard et al., 2005), but is addresses the management of annotation campaigns rather than the analysis of annotation tasks.

2.2 Good practices in manual annotation

Good practices have progressively emerged to tackle various aspects of annotation tasks. They have been, in particular, inherited from corpus linguistics. (Leech, 1993, 2005) present a list of what a specification for annotation should contain. It represents an effort toward the identification of several levels of difficulty encountered when making decisions during the manual annotation of a corpus: segmentation, inclusion of units (words or phrases) within other units (clauses or sentences), assigning categories to some textual fragments.

Annotation formats Recommendations were published within the framework of the standardization effort of the annotation formats, in particular in (Ide and Romary, 2006), that finally gave birth to the ISO 24612 standard. In this view, the authors are little concerned by manual annotation difficulties as such but, to represent annotations they identify complex structuring elements which are of interest here : segmentation (identification of continuous or discontinuous sequences of characters), several layers of annotations (for example morpho-syntactic, then syntactic), relations between these layers, overlapping issues.

Organization of annotation campaigns An overall schema of the organization of annotation campaigns has also emerged. It involves several actors, among which the client (who can, for example, be the organizer of an evaluation campaign), the annotation manager, that is the person who is going to write the annotation guide from a (clearly) expressed need, and the annotators themselves, who proceed from the annotation guide. The increasing use of so-called "non-experts", through crowdsourcing for example, introduced a further distinction between expert annotators (editors/curators in GATE Teamware (Bontcheva et al., 2010)) and annotators themselves. The multiplication of actors contributes to the difficulty to organize an annotation campaign but not to that of the annotation task *per se*. Note also that the analysis we propose here applies whatever the level of expertise of the annotators. Besides, the annotation guide is recognized as the keystone of annotation campaigns, as it defines what should be annotated. Here, we consider that the need is clearly defined and known from all the participants.

Annotation evaluation Studies concerning the evaluation of the quality of manual annotation allowed to identify some factors influencing inter- and intra-annotator agreements. (Gut and Bayerl, 2004; Bayerl and Paul, 2011) demonstrated that the inter-annotator agreement and the complexity of the annotation task are correlated (the larger the number of categories, the lower the inter-annotator agreement) and that the categories prone to confusions are in limited number. This brings out two complexity dimensions related to the number of categories and to the ambiguity between some categories. This ambiguity-related complexity dimension also appears in (Popescu-Belis, 2007). In the study of the inter-annotator agreements, (Krippendorff, 2004) identified a step of identification of the elements to annotate that is called "unitizing". Similarly, in the Proposition Bank project (Palmer et al., 2005), the organizers separated role "identification" from role "classification" to compute the inter-annotator agreement, in order to "isolate the role classification decisions" from the (supposedly easier) identification.

2.3 Insights from Cognitive Science

Few publications focus on the difficulties of manual annotation. In (Tomanek et al., 2010), the authors used an eye-tracking device to analyze and model the annotation cognitive complexity. Their experiment was carried out on a simple named-entity annotation task (persons, locations and organizations), with some pre-identification of complex noun phrases containing at least one potential named entity. They measured the influence of the syntactic and semantic complexities[1] as well as the size of context that was used by the annotators. The results show that the annotation performance tends on average to "correlate with the [semantic] complexity of the annotation phrase" and less so with its syntactic complexity, and that the size of the needed context also depends on the semantic complexity of the annotation phrase. However interesting, their conclusions only apply to simple named entity annotation task.

3 Measuring the complexity of an annotation task

Manual annotation requires the annotator to determine which units should be annotated and how. Measuring the complexity of an annotation task requires a detailed analysis of these localization and characterization operations.

We propose to analyze the complexity of elementary annotation tasks according to six dimensions: the first two (discrimination and delimitation) relate to the localization of annotations, while the next three concern their characterization (expressiveness, tagset dimension and ambiguity degree). The context is a sixth factor of complexity that impacts annotation decisions: it is presented here as a separate dimension for the sake of simplicity, even though it simultaneously affects discrimination, boundaries delimitation and disambiguation.

Analyzing a task along those six dimensions is artificial in the sense that annotators do not make separate decisions, but this analytic approach is meant for the management of annotation. It is independent from both the volume of annotations to be added and the number of annotators involved: these values participate in the cost of a task but not in its complexity, which is important to evaluate as early as possible in the annotation process.

3.1 Decomposition of annotation tasks

In an annotation task, one or several human annotators are asked to explicit how they interpret a source signal. The guidelines usually explain what kind of interpretation is expected and for which purpose. Depending on the task, the annotators are provided with a closed tagset or

[1] Respectively measured as the "number of nodes in the parse tree" and the "inverse document frequency of the words in the phrase according to a reference corpus".

are allowed to introduce any tag they find relevant. The annotators have to read the source document and tag some or all of its segments with one or several tags.

Instead of decomposing annotations tasks into *levels* or *layers* (Goecke et al., 2010), in order to analyze the task complexity, we propose to decompose a campaign into *elementary annotation tasks* (EATs). The complexity of the various EATs is computed independently and the complexity of a whole campaign is computed as the combination of the complexity of the elementary tasks. We consider that an annotation task can be decomposed into two or more EATs if the tagset itself is made of smaller independent ones. [2] This decomposition in EATs is formal in the sense that it is independent of the pragmatic organization of the work: the annotators can handle different EATs as separate steps on the source signal or all at once depending on the specific nature of the work and the tools they use. The decomposition in EATs does not result in a simplification of the original task as it is often the case for the Human Intelligence Tasks (HITs) to be performed by Turkers (workers) in Amazon Mechanical Turk (Cook and Stevenson, 2010).

To take a simple example, the annotation of gene renaming relations can be analyzed as a combination of two EATs. The first one identifies the gene names in the source signal. The second one relies on the first level of annotation and indicates which of the genes hold in a renaming relation. Obviously, the annotators can add both types of annotations at the same time, but the tagsets are independent and it is easier, from a formal point of view, to analyze the annotation task as a combination of two EATs than as a unique, but complex one.

3.2 What to annotate?

Localizing the units to be annotated consists in distinguishing what is to be annotated from what is not, and potentially adjusting the boundaries of the identified units.

3.2.1 Discrimination

In some annotation experiments, the question of "what to annotate" is straightforward, for instance when the units to annotate have already been marked in an automatic pre-annotation phase or when all the units are to be annotated, as in a POS-tagging task. However, for the annotators, the corpus is often a haystack within which they must find what to annotate, and discriminating what should be annotated from what should not is a complex task.

Identifying the units on which the annotation work should focus is all the more complex as the units to consider are heterogeneous. (Erk et al., 2003) emphasizes that semantic role annotation and discourse annotation mix several levels of segmentation (from less than a word to more than a sentence). As a simple example, it is easier to identify in a text negatively connoted adverbs, in particular, than all the negative expressions, as the latter can be words, phrases, or even entire parts of a text. In the second case several segmentation levels are actually to be considered. This gives a first scale of difficulty. An annotation task is considered difficult to the extent that the discrimination factor, defined by the following formula, is high:

$$Discrimination_a(F) = 1 - \frac{|A_a(F)|}{\sum_{i=1}^{n} |D_i(F)|}$$

where F is the flow of data to annotate, a is an annotation task, n is the number of segmentation levels that are potentially relevant, $|D_i(F)|$ is the number of units obtained during the

[2] Independence, here, means that the tags of two different tagsets are globally compatible (even if some specific combinations may be forbidden), whereas the tags of a single tagset are mutually exclusive (except for encoding ambiguity).

segmentation of F at level i and $|A_a(F)|$ is the number of units to be annotated in the relevant annotation task.

Intuitively, this measure indicates that the discrimination weight is high when the units to annotate or mark are "submerged" within others, and when the proportion of what is to be annotated ($|A_a(F)|$) as compared to what could be annotated or is "markable" ($\sum_{i=1}^{n} |D_i(F)|$) is low. The identification factor is 0 when all the units of the flow are to be annotated, and approaches 1 when only a few units are actually to be annotated while many could be annotated.

For the classification of pronouns as anaphoric or impersonal, the discrimination factor is 0 if all the occurrences of pronouns have been identified beforehand. By contrast, for gene renaming relations, the discrimination factor is high because only a small proportion of gene name couples participate in a renaming relation and are to be annotated as such.[3]

It is generally easy to estimate the number of units to be annotated – from the definition of the annotation task, on a sample of the relevant corpus or by reference to comparable tasks – but more difficult to estimate what could be annotated. This requires the choice of a reference segmentation that divides the flow to annotate into units, of which some are to be annotated, and others not. This reference segmentation can be chosen in different ways:

1. The simplest solution is to rely on an obvious segmentation which can be automatically computed or which is intuitive for the annotator, even if it implies reviewing the boundaries of the units to be annotated which do not correspond to this segmentation. For example, with named entities, starting from a segmentation into words leads to consider compound named entities as "modified" units. This strategy reduces the discrimination weight, but increases the issue of boundary delimitation (see section 3.2.2 below).

2. When the units to annotate are too heterogeneous, several reference segmentations can be considered ($n > 1$ in the preceding formula): this increases the number of "markables", but avoids the need to modify many boundaries. In the case of named entities annotation, one could for instance consider all the words and all the phrases as "markables". This approach must be used if it seems less costly than the previous one.

3. Finally, the annotation task can be decomposed into several layers, corresponding to distinct EATs. In that case, the discrimination weights of the layers are computed independently, each one with a specific reference segmentation. Such a decomposition is needed only when the different types of "markables" resulting from the various segmentations are annotated differently. It would be artificial for instance to decompose the task of named entities annotation into several EATs if the same tagset is used for words and phrases.

3.2.2 Boundary delimitation

Identifying points of interest in the flow of data is not enough, as the elements to be annotated are often data segments. To identify what to annotate, one should also delimit the boundaries of the segment to be annotated.

Here again, the task is easy when a reliable reference segmentation can be computed. However, segmentations that can be computed automatically are often approximations and the annotator must locally modify the boundaries of the discriminated units: for instance, if one starts with a segmentation into words to annotate named entities or terms, the segmentation must be

[3] We assume here that the gene names are pre-annotated (or identified in a different EAT) and that all gene name couples in the same abstract are "markable", *i.e.* subject to expressing a renaming relation.

corrected for all the multi-word expressions. In most cases, delimiting the boundaries consists in enlarging or reducing the discriminated unit based on the reference segmentation, but there are also cases in which a discriminated unit should be decomposed in several units or where several contiguous discriminated units are grouped together into one annotation.[4]

Delimitation of boundaries represents a second complexity factor, $Delimitation_a(F)$, which is computed by comparing the segmentation obtained after the annotation to the reference segmentation. This factor is inspired from the slot error rate (Makhoul et al., 1999), a metric that is used in named entities recognition and classification and which takes boundary errors into account. Once the discriminated units before and after the boundary delimitation are optimally aligned, we can compute the following discrimination rate:

$$Delimitation_a(F) = min(\frac{S+I+D}{|A_a(F)|}, 1)$$

where $|A_a(F)|$ is the final number of discriminated units, I the number of inserted units, obtained by initial units decomposition, D the number of units deleted when grouping some of the initial units, and S is the number of substitutions, *i.e.* the number of discriminated units that underwent a change in their boundaries other than that of the previous decomposition and grouping cases.

This delimitation factor is worth 0 when no discriminated unit has been modified and it rises with the number of decompositions, groupings and boundary modifications performed by the annotator with respect to the reference segmentation. The value is limited to 1 to keep the same interval $[0, 1]$ for the six complexity factors.

The delimitation cost is measured *a posteriori*, but it can be estimated based on a sample of annotated corpus or by comparison with a similar task.

3.3 How to annotate?

Once the units have been discriminated and delimited, they have to be characterized.

3.3.1 Expressiveness of the annotation language

We distinguish between three types of annotation languages: type, relational and higher order languages.

In the simplest case, annotation consists in associating a type with a data segment, *i.e.* in labeling it. Many annotation tasks rely on such a type language: words of a text are associated with part-of-speech, speech turns are associated with interlocutors or with rhetorical functions, phrases are associated with named entities types. In some cases, the label that is used is itself structured, as with morpho-syntactic labels associating a part-of-speech with a lemma and its morpho-syntactic features, but such structuring increases the size of the tagset (see section 3.3.2) without changing the expressiveness of the annotation language.

Establishing relations between units is also a common task, but it is more complex. The complexity of relational annotations should not be underestimated: even if annotators do not always proceed in exactly this way, they have to locate and type the arguments of the relation, discriminate the couples, triplets, and more generally the n-uplets of segments to annotate among the set of n-uplets that are markable, and finally type the relation existing between the elements of the n-uplet.

[4] We neglect here the case of discontinuous units, which are often not annotated as such and which are not numerous enough to impact the present analysis.

A higher-order language is used when annotations are added to other annotations, but the complexity of this type of language, which is difficult to formalize and manipulate, is such, that it is often reduced to a simpler language. For example, to qualify an annotation as uncertain, the generally preferred option is to increase the tagset size so as to add the qualifier as an attribute associated with a main type. Alternatively, the decomposition of an annotation task into EATs also allows to qualify or relate, in a later step, annotations added in a previous one: you obtain several EATs, each one relying on a first order annotation language.

The degree of complexity entailed by the expressiveness of the annotation language is naturally represented by an ordinal qualitative scale, but for the sake of homogeneity with the previous factors, described by numeric values, we associate the different levels of expressiveness with graduations on a numeric scale from 0 to 1. In this scale, 0.25 corresponds to type languages, 0.5 and 0.75 to relational languages, respectively of arity 2 and higher than 2, while the maximal value, 1, is dedicated to higher-order languages.

3.3.2 Tagset dimension

The annotator generally selects the value of an annotation in a predefined tagset that is presented in the annotation guide and this choice is all the more difficult to make as it is open. The size of the tagset is therefore a new factor of complexity.

In the simplest case, the choice is boolean and annotating amounts to assigning the discriminated units into two categories:[5] sentences may be marked as relevant or not; the occurrences of the *it* pronoun are marked as anaphoric or impersonal.

However, the choice is often more open, for instance for representing the diversity of morpho-syntactic units, annotating syntactic dependencies or typing named entities. For the richest annotations, structured labels are often proposed: the annotator adds several labels on a single given unit, the combination of which forms the final annotation (Dandapat et al., 2009).

Finally, there are annotation tasks for which the choice of a label is entirely left to the annotator, as in speech transcription, where there may be as many labels as words. In such cases, we consider that we have a huge tagset, even though the annotation effort is probably of a slightly different nature for the annotator.

If an annotation A is formed of a sequence of m labels ($A = E_1 E_2 \ldots E_m$) and each label E_i can take n_i different values, the complete tagset theoretically contains n different labels, with $n = n_1 * n_2 * \ldots * n_m$. However, in practice, constraints are defined which reduce the number of combinations: the annotator does not have to choose one label from n at once, but instead first select 1 label among n_1 labels, then 1 among at most n_2, etc. up to 1 among at most n_m. The size of the tagset does not depend on the total number of possible labels but on the degrees of freedom of the successive choices that the annotator has to make. The total degree of freedom ν for the choice of m labels is given by the following formula:

$$\nu \leq \nu_1 + \nu_2 + \ldots + \nu_m$$

where ν_i is the maximal degree of freedom the annotator has when choosing the i^{th} label ($\nu_i = n_i - 1$).[6] For instance, the tagset for the POS part of the Penn Treebank contains 36

[5] This boolean annotation is similar to the discrimination task, though the units to be annotated should be not only located but labeled.

[6] The formula gives a high boundary of the global degree of freedom because the choice of the i^{th} label is often constrained by the labels already added, so the annotator has in practice a degree of freedom that is less than $(n_i - 1)$, if n_i is the number of available labels at this point.

tags (Santorini, 1990), so v equals 35, but as some tags are subtypes of others (like JJR and JJS for JJ) there are 21 in fact tags corresponding to main categories, and, as the maximum number of subtypes is 6 (for verbs), we may consider that $v = 20 + 5 = 25$.

The tagset dimension can be computed using the following formula:

$$Dimension_a(F) = min(\frac{v}{\tau}, 1)$$

where v is the global degree of freedom the annotator has when choosing a label for an annotation task a within a flow of data F, and τ is the threshold from which we consider the tagset as arbitrarily large. In the experiments detailed below, τ is worth 50, based on the feedback of the annotators.

The tagset dimension is worth 0 for the binary tagsets presenting a degree of freedom of 1 and it increases with the size of the tagset and the correspondingly increasing degree of freedom. It is worth 0.5 for the Penn Treebank POS tag annotation (0.7, if we consider the tagset as flat). It reaches a ceiling at 1. Annotation tasks with large tagsets ($> \tau$) are very difficult to manage.

3.3.3 Degree of ambiguity

The need to disambiguate the units to annotate introduces a fifth complexity factor, which is more difficult to estimate than the previous ones. As the role of the annotator is precisely to resolve ambiguous cases whenever possible, ambiguities are difficult to observe. Still, we can evaluate the ambiguity degree the annotator must resolve for a given task in two ways.

The first method consists in measuring the residual ambiguity degree by observing the traces left by the annotator during the annotation: the annotation protocol may allow the annotator to annotate with several labels in case of ambiguity, to use an under-determined label, or even to add an uncertainty feature to the chosen label. This allows measurement of the degree of residual ambiguity:

$$Ambiguity_{Res,a}(F) = \frac{|Annot_A|}{|Annot|}$$

where a and F are the annotation task and the flow of data to be considered and where $|Annot_A|$ and $|Annot|$ are respectively the number of annotations bearing an ambiguity mark and the total number of annotations added to F. By definition, this residual degree of ambiguity can only be measured *a posteriori*, once the annotation has been performed, or from a sample of it.

The degree of residual ambiguity is worth 0 when no ambiguity mark was added by the annotator and would be worth 1 in the case (in real life, absurd) where all the annotations were marked as ambiguous, in one way or another. Obviously, depending on the traces which are used to compute it and on the directions given to the annotators, this metric can be more or less reliable and it should be associated, whenever possible, with results from another method.

This second method consists in measuring the theoretical degree of ambiguity for the tasks where several occurrences of the same vocables (vocabulary units) are annotated: this method applies to morpho-syntactic annotation or to semantic disambiguation but not to speech turn analysis or gene renaming relations. This metric relies on the idea that ambiguous vocables have occurrences that are annotated differently at different locations in the flow of data. The ambiguity factor is given by the proportion of units to be annotated that correspond to ambiguous vocables. The theoretical ambiguity can be measured from a dictionary that lists the possible labels for all the vocables, if the annotation relies on such a dictionary or, directly,

on a sample of annotated text. The theoretical ambiguity also depends on the frequency of ambiguous vocables in the flow of data to be annotated. It is computed in the following way:

$$Ambiguity_{Th,a}(F) = \frac{\sum_{i=1}^{|Voc(F)|} (Ambig_a(i) * freq(i,F))}{|Units_a(F)|}$$

with

$$Ambig_a(i) = \begin{cases} 1 & \text{if } \quad |Labels_a(i)| > 1 \\ 0 & \text{else} \end{cases}$$

where Voc is the vocabulary of the units of the flow of data F, $|Voc(F)|$ the size of the vocabulary, $freq(i,F)$ the frequency of the vocable i in F, $|Units_a(F)|$ the number of units to annotate in F and $|Labels_a(i)|$ the number of labels available for the vocable i for the annotation task a.

When there is no ambiguous vocable, $|Labels_a(i)|$ is worth 1 and $Ambig_a(i)$ is worth 0 for every i, the annotation task is trivial and can be easily automated by projecting on the flow of data a dictionary establishing the correspondence between the vocables and their labels. In this case, $Ambiguity_{Th,a}(F)$ is worth 0. Conversely, if all the vocables are ambiguous, $Ambiguity_{Th,a}(F)$ is worth 1. Note that the weight of an ambiguous vocable influences the degree of theoretical ambiguity in proportion to its frequency.

Theoretical ambiguity tends to overestimate the weight of the ambiguity for the annotator as some ambiguities are probably trivial to solve.

3.4 The weight of the context

The weight of the context is a sixth complexity factor. Even though it is not independent from the previous factors as they were from each other (it makes discrimination, delimitation and disambiguation more complex for the annotator), we represent it here as such, for the sake of simplicity.

The complexity of an annotation task increases with the size of the window to take into account around the unit to annotate and with the number of knowledge elements to be rallied. While it is difficult to determine the number of words participating in the resolution of an annotation task and *a fortiori* the number of knowledge elements at issue[7], we can identify two qualitative scales: the size of the data to be taken into account around the unit to be annotated and the degree of accessibility of the sources of knowledge that are consulted.

For the sake of homogeneity with the previous complexity factors, we translate the two preceding qualitative scales into a common discreet scale from 0 to 1:

- 0 corresponds to cases where no data around the unit to be annotated and no additional knowledge come into play. This is a theoretical case, since an annotation task where no context whatsoever is needed should actually be automated.
- Conversely, 1 is for the most complex cases, where the whole data flow and exterior sources of knowledge are necessary to annotate the units. The gene renaming relations annotation is one such, as the renaming relation, often hardly noticeable in the data flow, can often be confused with an isotopy (resemblance) relation or the membership in a common family. One must read the whole abstract to determine the semantics of the relation and sometimes the annotators must refer to an external source to better understand the properties of the genes and support their decision.

[7] Assuming that the elements of knowledge are countable, which is obviously an oversimplification.

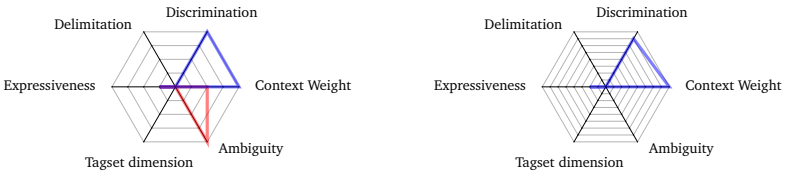
- We estimate at 0.25 the weight of the context in two cases: 1) if the annotation depends on the immediate environment of the unit to be annotated, or 2) if only provided sources, such as annotation guidelines, are to be consulted.
- The complexity is evaluated at 0.5 in three cases: 1) if the two previous difficulties combine; 2) if a larger part of the data is needed; or 3) if a well-identified exterior source of knowledge is to be consulted.
- Finally, the complexity is evaluated at 0.75 in three cases: 1) if the annotator must read the whole sentence and consult exterior sources to add annotations; 2) if s/he must take the entire flow of data into account; 3) if s/he must look for new knowledge sources.

This scale is obviously oversimplifying, but it is important to take that factor into consideration when planing an annotation campaign and the above criteria are meant for guiding the analysis and facilitating the comparison between complexity dimensions.

3.5 Synthesis

Since the six complexity factors are normalized on the same scale, once the complexity of different tasks is analyzed, it is easy to represent the various dimensions on a spider graph.[8]

Let us consider for instance the simple task, already mentioned, that consists in classifying pronoun occurrences as impersonal or anaphoric (REF OMITTED). Since the pronouns are previously tagged, both discrimination and delimitation are worth 0. The tagset being composed of two tags, this dimension is also at 0 and the expressiveness of the annotation language is 0.25 (type language). However, the ambiguity degree is high (1) as all occurrences are ambiguous. From our personal experience of annotation, we know that the context is worth 0.5 (or more) as the whole sentence must be considered to understand the role of the pronouns. The complexity of this task is represented on Figure 1a.



(a) Synthesis of the complexity dimensions of the pronouns classification (red) and gene names tagging (blue) campaigns

(b) Synthesis of the complexity dimensions of the whole gene names renaming campaign (2 EATs, double scale)

Figure 1: Two synthesis examples

The case of gene renaming (Jourde et al., 2011) is a little bit more complex and it is best analyzed as a combination of two EATs. Representing each EAT on a separate spider graph gives no idea of the whole task. We rather recommend to represent all EATs of a task in a single graph, which scale must be enlarged in proportion (from 0 to 2), as in Figure 1b. In the present case, the first EAT is the tagging of gene names in the sequence of words that composes the text. The discrimination factor is high (0.9), as only few words are gene names. Delimitation is 0, as

[8]The results presented in all the graphs in the article are rounded up/down to the nearest integer.

gene names, in our example, are simple tokens. On the contrary, the characterizing factors are low: the tagset is boolean (Dimension=0), a type language is used (Expressiveness=0.25) and ambiguity is very low as only few gene names are also common names (theoretical ambiguity can be approximated at 0.01[9] and residual ambiguity is on average of 0.04 for two annotators). In this case, the context is the highest factor as it is often necessary to read the whole PubMed abstract to understand the role of a mentioned entity and as annotators sometimes consult external resources (context weights 1) (Fort et al., 2010).

This first EAT is represented on the same graph as the pronoun classification, thus enabling the comparison of the two tasks (see Figure 1a). If they both show little complexity on three dimensions (delimitation, expressiveness, tagset dimension), the first one (pronouns) presents a high ambiguity dimension and no discrimination problem, while the second one (gene names) shows high complexity levels on the discrimination and context dimensions and no ambiguity. The solutions to alleviate the costs of these campaigns should therefore be adapted (pre-annotation and easy access to context for gene name tagging, and probably a carefully designed documentation for the pronoun classification).

Gene renaming consists in linking the gene name occurrences that hold a renaming relation. The whole task therefore comprises a second EAT, which consists in marking the renaming relations or its absence on any couple of gene names co-occurring in the same abstract. In that case, the discrimination is high (0.95) as only few gene couples are actually renaming each other. Delimitation is null since the gene names have already been annotated. The tagset is composed of 3 tags as the renaming relation is oriented (Dimension=0.04). Even if the annotations carry relational information, the language is a type language (a couple of gene names bears a tag, which expresses the direction of the relation). Ambiguity is very low (residual ambiguity is on average of 0.02 for the two annotators), but the context is high, as in the previous case. Figure 1b shows how the two EATs are combined to provide an analysis of the complexity of the whole task, which proves to be focused on discrimination and context.

4 Validation and illustration

4.1 Experimental validation

Although, as we showed in section 2, this grid of analysis has never been identified as such, some existing results or experiments confirm its applicability.

One first example of this is related to discrimination. In experiments led on the effects of pre-annotation on POS-tagging, the authors of (Fort and Sagot, 2010) showed that if the automatic pre-annotation is of very good quality, *i.e.* if only few tokens disseminated within the text have to be corrected, very good annotators can end up with less good annotation results, due to lapses in their concentration. This corresponds directly to the discrimination dimension we present here, which tends to get higher if the annotations to perform are submerged within the text.

Also, it has been observed that the quality of the annotation decreases with the number of tags involved (Bayerl and Paul, 2011). Structuring the tagset allows to reduce the degree of freedom at one point of choice and an appropriate annotation tool can help efficiently dealing with the hierarchy of tags (Dandapat et al., 2009).

Such an annotation tool can also help with complex language types, like relations, providing a graphical, user-friendly way to annotate them, like with Knowtator (Ogren, 2006). However, the number of manipulations involved is still higher than for simple type languages.

[9] To obtained this, we checked how many of the tagged gene names could also be tokens from the Brown corpus.

From the ambiguity point of view, the most obvious way to reduce it is to identify and remove the cases which are problematic. In the syntactic part of the Penn Treebank annotation, significant time was saved by reducing the ambiguity causes: "It proved to be very difficult for annotators to distinguish between a verb's arguments and adjuncts in all cases. Allowing annotators to ignore this distinction when it is unclear (attaching constituents high) increases productivity by approximately 150-200 words per hour." (Marcus et al., 1993). The analysis we propose here should help identifying these cases sooner in the process.

In the context of the gene renaming annotation campaign, we discovered that in some cases, the annotators needed the whole text to make their final decisions, and not only the abstract they had to annotate, as it was initially planned. If this need could have been identified beforehand, it would have been taken into account and the annotation tool would have been parameterized to give an easy access to the whole documents to annotators.

4.2 Example: structured named entities

We applied this analysis on a structured named entity annotation campaign for French described in (Grouin et al., 2011).

Structured named entities Within the Quaero program[10], a new definition of structured named entities was proposed. This new structure relies on two principles: the entities are both hierarchical and compositional. An entity is then composed of two kinds of elements: the 7 types (and 32 subtypes) which refer to a general segmentation of the world into major categories and the 31 components which allow to tag every word of the entity expression. Following this definition, the annotation campaign can be decomposed into two EATs: types and components. Figure 2 illustrates this definition on a French phrase. The types (EAT 1) are shown in red tags and the components (EAT 2) in blue tags.

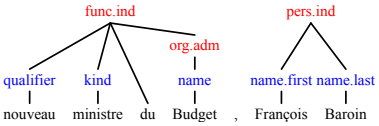


Figure 2: Multi-level annotation of entity types (red tags) and components (blue tags): *new minister of budget, François Baroin*.

Analysis For this illustration, we used the French spoken data provided by the mini-reference corpus (see (Grouin et al., 2011)) comprising 11,532 tokens, 1,161 entity types and 1,778 components.[11] Figure 3 represents the overall scores for this task using a spider graph.

The discrimination score is of 0.90 for the annotation of the types and 0.14 for the annotation of the components. To compute a delimitation score on such data, we first replaced in the annotated file every tag with a simple tag (*annot*) and in the same, but unannotated, file (as a reference file, under the hypothesis where tokens constitute the normal delimitation) we added the same tag around each token. Then we computed a slot error rate (SER) between these two files. The SER for types is over 100% (Delimitation=1) and the SER for components is 30% (Delimitation=0.3).

[10] www.quaero.org

[11] This mini-reference corpus is a sub-part of the whole corpus, that contains 1,291,225 tokens, 113,885 types and 146,405 components.

The annotation language is a type language, the degree of expressiveness is therefore of 0.25 for each of the EATs. Taking into account the types and sub-types structure, the total degree of freedom v for the annotation of the different types (EAT 1) is 10. The dimension score is then of 0.2. Concerning the components EAT, the total degree of freedom is 30 and the dimension score of 0.6. The theoretical ambiguity is computed for each EATs. The score is of 0.15 and 0.12 for the types and components respectively. These scores are low and the tasks do not seem very complex from the point of view of this dimension. The sample size is probably to blame here, as, if we compute this score *a posteriori* using the overall corpus (which is of course not possible in a real situation) the scores are of 0.49 for the components and of 0.36 for the types. Additional experimental results are necessary to take this scaling factor into account in the measure of ambiguity.

Concerning the weight of the context we estimate it at 0.75 because the annotators had to take into account the entire flow of data (the entity definition is contextual). Moreover, it was sometimes necessary to validate a choice by exploring external data (such as Wikipedia).

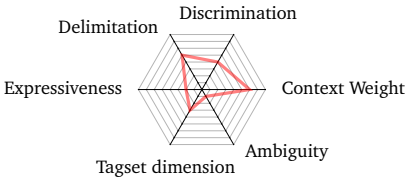


Figure 3: Synthesis of the complexity for the structured named entities campaign (2 EATs, double scale)

This analysis validates the choice of the hierarchical tagset that has been done for the annotation campaign described in(Grouin et al., 2011). Had a flat tagset been chosen, the dimension score would have been 1 ($v = 61$). Moreover, this analysis is in line with what was observed concerning the context weight within the campaign(Rosset et al., 2012). Of course, it presents some limits as it has been shown for the ambiguity score computation.

Conclusion

The grid of analysis we propose here should be used as part of the preparatory work of any annotation campaign, large or small. Obviously, when done *a priori*, on a small sample of annotations, the results will be approximate, but we believe that the analysis itself helps asking the right questions and finding the appropriate solutions.

Obviously, this pre-campaign work should be supported by an appropriate tool, so that this grid of analysis is computed more or less automatically. We are writing specifications for such a module, intended to be plugged into annotation management tools like Slate (Kaplan et al., 2010) or GATE Teamware (Bontcheva et al., 2010).

Acknowledgments

This work was realized as part of the Quæro Programme[12], funded by OSEO, French State agency for innovation.

[12] `http://quaero.org/`

References

Abeillé, A., Clément, L., and Toussenel, F. (2003). Building a treebank for French. In Abeillé, A., editor, *Treebanks*, pages 165 –187. Kluwer, Dordrecht.

Bayerl, P. S. and Paul, K. I. (2011). What determines inter-coder agreement in manual annotations? a meta-analytic investigation. *Computational Linguistics*, 37(4):699–725.

Bonneau-Maynard, H., Rosset, S., Ayache, C., Kuhn, A., and Mostefa, D. (2005). Semantic annotation of the French media dialog corpus. In *Proceedings of the InterSpeech*, Lisboa, Portugal.

Bontcheva, K., Cunningham, H., Roberts, I., and Tablan, V. (2010). Web-based collaborative corpus annotation: Requirements and a framework implementation. In Witte, R., Cunningham, H., Patrick, J., Beisswanger, E., Buyko, E., Hahn, U., Verspoor, K., and Coden, A. R., editors, *Proceedings of the workshop on New Challenges for NLP Frameworks (NLPFrameworks 2010)*, Valletta, Malta. ELRA.

Cook, P. and Stevenson, S. (2010). Automatically identifying changes in the semantic orientation of words. In Chair), N. C. C., Choukri, K., Maegaard, B., Mariani, J., Odijk, J., Piperidis, S., Rosner, M., and Tapias, D., editors, *Proceedings of the Seventh conference on International Language Resources and Evaluation (LREC'10)*, Valletta, Malta.

Dandapat, S., Biswas, P., Choudhury, M., and Bali, K. (2009). Complex linguistic annotation - no easy way out! a case from bangla and hindi POS labeling tasks. In *Proceedings of the third ACL Linguistic Annotation Workshop*, Singapore.

Erk, K., Kowalski, A., Padó, S., and Pinkal, M. (2003). Towards a resource for lexical semantics: a large German corpus with extensive semantic annotation. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics - Volume 1*, ACL 03, pages 537–544, Morristown, NJ, USA. Association for Computational Linguistics.

Fort, K., François, C., and Ghribi, M. (2010). Evaluer des annotations manuelles dispersées : les coefficients sont-ils suffisants pour estimer l'accord inter-annotateurs ? In *Proceedings of the Traitement Automatique des Langues Naturelles (TALN)*, Montreal, Canada. 10 pages.

Fort, K. and Sagot, B. (2010). Influence of pre-annotation on POS-tagged corpus development. In *Proceedings of the Fourth ACL Linguistic Annotation Workshop*, pages 56–63, Uppsala, Sweden.

Goecke, D., Lüngen, H., Metzing, D., Stührenberg, M., and Witt, A. (2010). Different views on markup. In Witt, A., Metzing, D., and Ide, N., editors, *Linguistic Modeling of Information and Markup Languages*, volume 40 of *Text, Speech and Language Technology*, pages 1–21. Springer Netherlands.

Grouin, C., Rosset, S., Zweigenbaum, P., Fort, K., Galibert, O., and Quintard, L. (2011). Proposal for an extension of traditional named entities: From guidelines to evaluation, an overview. In *Proceedings of the 5th Linguistic Annotation Workshop*, pages 92–100, Portland, Oregon, USA. (poster).

Gut, U. and Bayerl, P. S. (2004). Measuring the reliability of manual annotations of speech corpora. In *Proceedings of the Speech Prosody*, pages 565–568, Nara, Japan.

Ide, N. and Romary, L. (2006). Representing linguistic corpora and their annotations. In *Proceedings of the Fifth International Conference on Language Resources and Evaluation(LREC 2006)*, Genoa, Italy.

Jourde, J., Manine, A.-P., Veber, P., Fort, K., Bossy, R., Alphonse, E., and Bessières, P. (2011). BioNLP shared task 2011 – bacteria gene interactions and renaming. In *Proceedings of the BioNLP Shared Task 2011 Workshop*, pages 65–73, Portland, Oregon, USA. Association for Computational Linguistics.

Kaplan, D., Iida, R., and Tokunaga, T. (2010). Annotation process management revisited. In *Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC 2010)*, pages 365 – 366.

Krippendorff, K. (2004). *Content Analysis: An Introduction to Its Methodology, second edition*, chapter 11. Sage, Thousand Oaks, CA., USA.

Leech, G. (1993). Corpus annotation schemes. *Literary and Linguistic Computing*, 8(4):275–281.

Leech, G. (2005). *Developing Linguistic Corpora: a Guide to Good Practice*, chapter Adding Linguistic Annotation, pages 17–29. Oxford: Oxbow Books.

Makhoul, J., Kubala, F., Schwartz, R., and Weischedel, R. (1999). Performance measures for information extraction. In *Proceedings of DARPA Broadcast News Workshop*, pages 249–252.

Marcus, M., Santorini, B., and Marcinkiewicz, M. A. (1993). Building a large annotated corpus of English : The Penn Treebank. *Computational Linguistics*, 19(2):313–330.

Ogren, P. (2006). Knowtator: A plug-in for creating training and evaluation data sets for biomedical natural language systems. In *Protégé Conference*, Stanford, USA.

Palmer, M., Gildea, D., and Kingsbury, P. (2005). The proposition bank: An annotated corpus of semantic roles. *Computational Linguistics*, 31(1):71–106.

Popescu-Belis, A. (2007). Le rôle des métriques d'évaluation dans le processus de recherche en tal. *T.A.L. : Traitement automatique de la langue*, vol. 48, n. 1:67–91.

Rosset, S., Grouin, C., Fort, K., Galibert, O., Kahn, J., and Zweigenbaum, P. (2012). Structured named entities in two distinct press corpora: Contemporary broadcast news and old newspapers. In *Proceedings of the 6th Linguistic Annotation Workshop (LAW VI)*, pages 40–48, Jeju, Republic of Korea.

Santorini, B. (1990). Part-of-speech tagging guidelines for the Penn Treebank Project. Technical Report MS-CIS-90-47, Department of Computer and Information Science, University of Pennsylvania, USA.

Tomanek, K., Hahn, U., Lohmann, S., and Ziegler, J. (2010). A cognitive cost model of annotations based on eye-tracking data. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, ACL '10, pages 1158–1167, Stroudsburg, PA, USA. Association for Computational Linguistics.

Extractive Multi-Document Summarization with Integer Linear Programming and Support Vector Regression

Dimitrios Galanis, Gerasimos Lampouras and Ion Androutsopoulos
Department of Informatics
Athens University of Economics and Business
Patission 76, GR-104 34 Athens, Greece
galanisd@aueb.gr, lampouras06@aueb.gr, ion@aueb.gr

ABSTRACT
We present a new method to generate extractive multi-document summaries. The method uses Integer Linear Programming to jointly maximize the importance of the sentences it includes in the summary and their diversity, without exceeding a maximum allowed summary length. To obtain an importance score for each sentence, it uses a Support Vector Regression model trained on human-authored summaries, whereas the diversity of the selected sentences is measured as the number of distinct word bigrams in the resulting summary. Experimental results on widely used benchmarks show that our method achieves state of the art results, when compared to competitive extractive summarizers, while being computationally efficient as well.

KEYWORDS: Text Summarization; Integer Linear Programming; Support Vector Regression.

Proceedings of COLING 2012: Technical Papers, pages 911–926,
COLING 2012, Mumbai, December 2012.

1 Introduction

A multi-document summarization system aims to generate a single summary from an input set of documents. The input documents may have been obtained, for example, by submitting a query to an information retrieval engine and retaining the most highly ranked documents, or by clustering the documents of a large collection and then using each cluster as a set of documents to be summarized. Although evaluations with human judges also examine the coherence, referential clarity, grammaticality, and readability of the summaries (Dang, 2005, 2006; Dang and Owczarzak, 2008), and some of these factors have also been considered in recent summarization algorithms (Nishikawa et al., 2010b; Woodsend and Lapata, 2012), most current multi-document summarization systems consider only the importance of the summary's sentences, their non-redundancy (also called diversity), and the summary length (McDonald, 2007; Berg-Kirkpatrick et al., 2011; Lin and Bilmes, 2011).

An *extractive* multi-document summarizer forms summaries by extracting (selecting) sentences from the input documents, without modifying the selected sentences. By contrast, an *abstractive* summarizer may also shorten or, more generally, rephrase the selected sentences. In practice, the additional processing of the selected sentences may only marginally improve or even reduce the perceived quality of the resulting summaries (Gillick and Favre, 2009), though recent work has produced abstractive summarization methods that perform better than extractive ones (Berg-Kirkpatrick et al., 2011; Woodsend and Lapata, 2012). Nevertheless, the difference in the performance of extractive and abstractive summarizers is often small, and abstractive summarizers typically require more processing time, as well as tools and resources (e.g., reliable large coverage parsers, paraphrasing rules) that are often not available in less widely spoken languages. Hence, it is still worth trying to improve extractive summarizers, at least from a practical, application-oriented point of view.

Many multi-document summarizers, especially extractive ones, adopt a greedy search when constructing summaries. For example, they may rank the sentences of the input documents by importance, and then iteratively add to the summary (and remove from the ranked list of input sentences) the sentence with the highest importance score, until the maximum allowed summary length has been reached, possibly discarding sentences that are too similar to sentences already included in the summary. Recent work has shown that adopting more principled optimization methods based on Integer Linear Programming (ILP), instead of greedy search, can lead to summaries that are better or at least comparable to those of state of the art summarizers (McDonald, 2007; Gillick and Favre, 2009; Nishikawa et al., 2010a).

In this paper, we introduce a new extractive multi-document summarization method that uses ILP to jointly optimize the importance of the summary's sentences and their diversity (non-redundancy), also respecting the maximum allowed summary length. Our method is more efficient than the seminal ILP-based summarizer of McDonald (2007), because of its simpler ILP model. The main competitor of our method, if we exclude abstractive summarizers, is the extractive version of Berg-Kirkpatrick et al.'s (2011) summarizer, which has the best previously reported results in extractive multi-document summarization. Inspired by Berg-Kirkpatrick et al.'s work, we include in the objective function of our ILP model the number of distinct word bigrams (of the input documents) that occur in the summary, but we use that number to measure diversity, unlike Berg-Kirkpatrick et al.'s work, where bigrams are weighted to measure importance. To obtain an importance score for each sentence, we use a Support Vector Regression (SVR) model (Vapnik, 1998), which has no direct counter-part

in Berg-Kirkpatrick et al.'s method. We show that our ILP method achieves state of the art ROUGE scores (Lin, 2004) on widely used benchmark datasets, when compared to Berg-Kirkpatrick's and other competitive extractive summarizers, also outperforming two greedy baselines that use only the importance scores of the SVR. For completeness, we also discuss and compare against the abstractive version of Berg-Kirkpatrick et al.'s summarizer, and the state of the art abstractive summarizer of Woodsend and Lapata (2012).

Section 2 below discusses previous work on ILP methods for summarization. Section 3 presents our own ILP model, after first introducing the SVR model of sentence importance and the greedy baselines. Section 4 presents the experiments that we conducted and discusses their results. Section 5 concludes and proposes directions for further research.

2 Related work

The first ILP method for summarization was proposed by McDonald (2007). It constructs summaries by maximizing the importance of the selected sentences and minimizing their pairwise similarity, as shown below. No sentence ordering is performed.

$$\max_{x,y} \sum_{i=1}^{n} imp(s_i) \cdot x_i - \sum_{i=1}^{n} \sum_{j=i+1}^{n} sim(s_i, s_j) \cdot y_{i,j} \tag{1}$$

subject to:

$$\sum_{i=1}^{n} l_i \cdot x_i \leq L_{max} \tag{2}$$

and (for $i = 1, \ldots, n$ and $j = i+1, \ldots, n$):

$$y_{i,j} - x_i \leq 0 \tag{3}$$

$$y_{i,j} - x_j \leq 0 \tag{4}$$

$$y_i + x_j - y_{i,j} \leq 1 \tag{5}$$

Here, n is the number of sentences in the input documents; $imp(s_i)$ is the importance score of sentence s_i; l_i is the length of s_i; $sim(s_i, s_j)$ is the similarity of sentences s_i and s_j; and L_{max} is the maximum allowed length. The x_i variables, jointly denoted x, are binary and indicate whether or not the corresponding sentences s_i are included (selected) in the summary. The $y_{i,j}$ variables, jointly denoted y, are also binary and indicate whether or not both s_i and s_j are included in the summary. Constraint 2 ensures that the maximum total length is not exceeded. Constraints 3–5 ensure that the values of x_i, x_j, and $y_{i,j}$ are consistent (e.g., if $y_{i,j} = 1$, then $x_i = x_j = 1$; and if $y_{i,j} = 0$, then $x_i = 0$ or $x_j = 0$).

McDonald showed experimentally that the ILP model above achieves better ROUGE scores (Lin, 2004) than a greedy method that attempts to maximize the same objective (1). However, McDonald also showed that the ILP model above corresponds to an NP-hard problem and is, therefore, intractable for a large number of sentences. A set of experiments by McDonald confirmed that the model does not scale up well in practice, mostly because of the $O(n^2)$ $y_{i,j}$ variables that are used to model the redundancy between sentences. Furthermore, the ROUGE scores of McDonald's ILP model were not always better than those obtained using a modified version of the Knapsack dynamic programming algorithm (Cormen et al., 2001).

In a more recent approach, Berg-Kirkpatrick et al. (2011) presented an ILP method based on the notion of 'concepts', a notion initially introduced by Gillick and Favre (2009). The

so called 'concepts' are actually word bigrams, all the word bigrams of the documents to be summarized. Each bigram has a weight w_i that indicates its importance. The ILP objective (6) of Berg-Kirkpatrick et al. prefers summaries with many important concepts, i.e., summaries whose bigrams have a large sum of weights w_i; below b_i are binary variables indicating which bigrams ($|B|$ in total) are present in the summary. An additional constraint, not shown here, ensures that the maximum allowed summary length is not exceeded.

$$\max_{b,c} h(b,c) = \max_{b,c} \sum_{i=1}^{|B|} w_i \cdot b_i + \sum_{i=1}^{|C|} u_i \cdot c_i = \sum_{i=1}^{|B|} (W^T \cdot \Phi_i) \cdot b_i + \sum_{i=1}^{|C|} (U^T \cdot \Psi_i) \cdot c_i \qquad (6)$$

Berg-Kirkpatrick et al.'s model also takes into account the possible subtree cuts (deletions) of the parse trees of the sentences of the input documents. The cuts give rise to different compressions (shortenings) of the sentences; hence, Berg-Kirkpatrick et al.'s summarizer is an abstractive one. In the objective (6), u_i are the weights of the possible subtree cuts of all the sentences of the input documents, and c_i are binary variables indicating which cuts ($|C|$ in total) are used. Additional contraints, not shown above, ensure that the values of b_i and c_i are consistent. Overall, Berg-Kirkpatrick et al.'s method aims to produce summaries that contain many important bigrams, while also performing many desirable subtree cuts.

The weights w_i and u_i are themselves estimated as weighted sums of features, i.e., $w_i = W^T \cdot \Phi_i$, where Φ_i is a feature vector describing the bigram of the binary variable b_i, and W is a vector of feature weights; similarly, $u_i = U^T \cdot \Psi_i$, where Ψ_i is a feature vector describing the subtree cut of the binary variable c_i, and U is a vector of feature weights.[1] The feature vector Φ_i includes, for example, the frequency of the corresponding bigram in the documents to be summarized, and the minimum sentence position (e.g., 3rd sentence in a document) of the sentences that contain that bigram in the input documents. The features of Ψ_i show, for example, if a relative clause or a temporal phrase was cut.

Berg-Kirkpatrick et al. use a structured Support Vector Machine (SVM) (Vapnik, 1998; Tsochantaridis et al., 2004) that assigns to each candidate summary the score $h(b,c)$ of the objective function (6), given W and U. During training, the SVM searches for the values of W and U that allow it to prefer the gold summary (of each training set of input documents) to all the other possible summaries (of the same input documents) by a margin determined by a loss function. Berg-Kirkpatrick et al. use a bigram recall loss function similar to ROUGE-2 (Lin, 2004). The loss function causes more emphasis (larger margin) to be placed on preferring the gold summaries to other summaries that share many bigrams with the gold ones. The learnt W and U are then used in the objective (6).

Berg-Kirkpatrick et al. report that their full method achieves higher ROUGE scores than an extractive version of their method (without the subtree cuts, i.e., without sentence compression) with no significant decrease in grammaticality (when sentence compression is used), unlike other work (Gillick and Favre, 2009), where sentence compression was found to reduce grammaticality. The extractive version of Berg-Kirkpatrick et al.'s method omits the second term of Formula (6), as in the previous work of Gillick and Favre (2009). As already noted, the extractive version of Berg-Kirkpatrick et al.'s method has the best previously published results in extractive multi-document summarization.

[1] Berg-Kirkpatrick et al. (2011) use different terminology. A minor difference from their description of their method is that they seem to set $W = U$, but in a more general formulation this does not seem to be necessary.

More recently, Woodsend and Lapata (2012) proposed an ILP-based method that forms a summary by maximizing the objective function shown below. The objective function combines the importance $f_B(z)$ of the bigrams in the summary's sentences, the salience $f_S(z)$ of the parse tree nodes of the summary's sentences, and a unigram language model $f_{LR}(z)$, which penalizes sentences containing words that are unlikely to appear in summaries; we do not discuss $f_{LR}(z)$ further to save space.

$$max_z f_B(z) + f_S(z) + f_{LR}(z) \tag{7}$$

The argument z collectively denotes binary variables z_i, one z_i for each node of the parse tree of each sentence of the input documents. Each z_i shows whether or not the corresponding node has been retained or deleted. By deleting nodes, the method can compress sentences; hence, this is also an abstractive summarization method. The $f_B(z)$ component is the same as in Berg-Kirkpatrick et al.'s work ($f_B(z) = \sum_{i=1}^{|B|} w_i \cdot b_i$). Additional constraints ensure that a bigram can be selected only if at least a parse tree node that subsumes it has been selected, that the maximum summary length is not exceeded etc.

To compute $f_S(z)$, Woodsend and Lapata train a linear SVM, with separating hyperplane $W^T \cdot \Phi_i = 0$, to predict whether or not a sentence s_i of an input set of documents would be selected by a human creating a summary. The $f_S(z)$ score, defined below, is the sum of the trained SVM's predictions, for all the phrases that correspond to the retained parse tree nodes of the input sentences; Φ_i is a feature vector describing each retained phrase, with features indicating, for example, if the phrase was obtained from the first sentence of an input document, if it contains pronouns etc.; W are the feature weights learnt by the SVM.

$$f_S(z) = \sum_i (W^T \cdot \Phi_i) \cdot z_i \tag{8}$$

A second SVM is trained to exclude sentences that are too long, contain quotations etc. The predictions of the second SVM are used to add hard constraints to the ILP model, rather than including the predictions in the ILP objective function (7).

The method of Woodsend and Lapata optionally employs a quasi-synchronous tree grammar (QSTG), to generate candidate compressions and paraphrases of the source sentences. The QSTG grammar is learnt from aligned summary and source sentences. When the grammar is used, the rest of the method does not operate only on the sentences of the input documents, but also on rephrasings of these sentences, produced by the grammar. Hence, in its full form, the method of Woodsend and Lapata is abstractive not only because it can delete tree nodes of the parse trees, but also because it can rephrase sentences using the grammar. It can be turned into an extractive summarization method by disabling the QSTG grammar and disallowing tree node deletions. Woodsend and Lapata provide experimental results of their method without the QSTG grammar, but not without tree node deletions; hence, we could not compare directly to a purely extractive version of Woodsend and Lapata'a summarizer.

Lin and Bilmes (2011) construct summaries by maximizing a monotone submodular function. This is an NP-hard problem; however, there is a greedy algorithm that approximates the optimum by a constant factor. Lin and Bilmes show that several previous summarization approaches can be described in terms of submodular functions. They also propose their own submodular functions for summarization, which combine importance and diversity.

3 Our method

In this section, we first discuss our SVR model that assigns importance scores to the sentences of the input documents, and two greedy baseline summarizers that use the SVR without ILP. We then introduce our ILP method, which jointly maximizes the importance and diversity of the selected sentences, while respecting the maximum allowed summary length.

3.1 The SVR model of sentence importance

A Support Vector Regression (SVR) model aims to learn a function $f : \mathbb{R}^n \mapsto \mathbb{R}$, which will be used to predict the value of a variable $y \in \mathbb{R}$ given a feature vector $X \in \mathbb{R}^n$. In particular, given l training instances $(X_1, y_1), \ldots, (X_l, y_l)$, an SVR model is learnt by solving the following optimization problem (Vapnik, 1998); W is a vector of feature weights; ϕ is a function that maps feature vectors to a new vector space of higher dimensionality to allow non-linear functions to be learnt in the original space; $C > 0$ and $\epsilon > 0$ are given.

$$\min_{W,b,\xi,\xi^*} \frac{1}{2}\|W\|^2 + C\sum_{i=1}^{l}\xi_i + C\sum_{i=1}^{l}\xi_i^* \tag{9}$$

subject to (for $i = 1, \ldots, l$):

$$W^T \cdot \phi(X_i) + w_0 - y_i \leq \epsilon + \xi_i \tag{10}$$

$$y_i - W^T \cdot \phi(X_i) - w_0 \leq \epsilon + \xi_i^* \tag{11}$$

$$\xi_i \geq 0 \tag{12}$$

$$\xi_i^* \geq 0 \tag{13}$$

The goal is to learn a linear (in the new space) function, whose prediction (value) $W^T \cdot \phi(X_i) + w_0$ for each training instance X_i will not to be farther than ϵ from the target (correct) value y_i. Since this is not always feasible, two slack variables ξ_i and ξ_i^* are used to measure the prediction's error above or below the target y_i. The objective (9) jointly minimizes the total prediction error and $\|W\|$, to avoid overfitting.[2]

In our case, X_i is the feature vector of a sentence and y_i is the sentence's importance score. During training, the target score y_i of each sentence s, i.e., the score that the SVR should ideally return, is taken to be the average of the ROUGE-2 and ROUGE-SU4 scores (Lin, 2004) of s, comparing s against the corresponding gold (human-written) summaries; the latter are included in the training datasets that we used. We took the average of ROUGE-2 and ROUGE-SU4, because they are the two most commonly used measures to automatically evaluate machine-generated summaries against gold ones. Roughly speaking, both measures compute the bigram recall of a summary (or individual sentence) being evaluated against multiple gold summaries (provided by different human authors), but ROUGE-SU4 also considers skip bigrams with a maximum distance of 4 words between the words of each skip bigram. Both measures have been found to correlate well with human judgements in extractive summarization (Lin, 2004). Hence, training the SVR to predict the average ROUGE-2 and ROUGE-SU4 of each sentence can be particularly useful. Intuitively, a sentence with a high ROUGE score has a high overlap with the gold summaries; and since the gold summaries

[2] We use the SVR implementation of LIBSVM (`http://www.csie.ntu.edu.tw/~cjlin/libsvm/`) with an RBF (non-linear) kernel and LIBSVM's parameter tuning facilities.

contain the sentences that human authors considered most important, a sentence with a high ROUGE score is most likely also important. This is why we view our SVR, which attempts to predict the ROUGE score (average ROUGE-2 and ROUGE-SU4) of each sentence, as a component that assigns an importance score to each sentence.

The idea to use ROUGE during training is also present in the work of Berg-Kirkpatrick et al. (Section 2). The SVM that Berg-Kirkpatrick et al. use, however, in effect attempts to separate (prefer) the gold summaries from the other possible summaries; ROUGE (more precisely, a modified version of ROUGE-2) is included in the SVM as a loss function to force the SVM to place more emphasis on separating gold summaries from other possible summaries with high ROUGE scores. By contrast, the SVR that we use attempts to directly output the ROUGE score of each sentence. Furthermore, the RBF kernel that we use in the SVR allows the SVR to learn non-linear functions, whereas the linear SVM of Berg-Kirkpatrick et al. can learn only linear functions. We also note that the two SVMs used by Woodsend and Lapata (Section 2) perform binary classification (not regression), attempting to separate sentences that a human would include in a summary from sentences that would not be included. The (unsigned) distance from the learnt separating hyperplane of the first SVM is included in the objective function of the ILP model, in effect treating the distance as a confidence score. We believe that our use of a regression model (SVR) is a better choice, because the distance from an SVM's separating hyperplane is often a poor confidence estimate. We also note that the second SVM of Woodsend and Lapata contributes only hard constraints to the ILP model, without taking into account the SVM's confidence.

We include the following features in the feature vector X of each sentence s:

- Sentence position $SP(s)$:
$$SP(s) = \frac{pos(s, d(s))}{|d(s)|}$$
where $pos(s, d(s))$ is the position (sentence order) of sentence s in its document $d(s)$, and $|d(s)|$ is the number of sentences in $d(s)$.

- Named entities $NE(s)$:
$$NE(s) = \frac{n(s)}{len(s)}$$
$n(s)$ is the number of named entities in s, and $len(s)$ is the number of words in s.[3]

- Levenshtein distance $LD(s, q)$: The Levenshtein Distance (Levenshtein, 1966) between the user's query q and sentence s; insertions, deletions, and replacements affect entire words. In the datasets we experimented with, the documents to be summarized were relevant to a query q, which was always available.

- Word overlap $WO(s, q)$: The number of words shared by the query q and sentence s, after removing stop words and duplicate words from both q and s.

- Content word frequency $CF(s)$ and document frequency $DF(s)$: We use these measures as defined by Schilder and Ravikumar (2008). $CF(s)$ is defined as follows:
$$CF(s) = \frac{\sum_{i=1}^{c_s} p_c(w_i)}{c_s}$$

[3]We use Stanford University's named entity recognizer (consult `http://nlp.stanford.edu/`).

where c_s is the number of content words in sentence s, $p_c(w) = \frac{m}{M}$, m is the number of occurrences of content word w in the input documents, and M is the total number of content word occurences in the input documents. Similarly, $DF(s)$ is defined as:

$$DF(s) = \frac{\sum_{i=1}^{c_s} p_d(w_i)}{c_s}$$

where $p_d(w) = \frac{d}{D}$, d is the number of input documents the content word w occurs in, and D is the number of all input documents.

Experiments on the development set (see below) confirmed that all the features have a positive impact, i.e., the results are worse, if any of the features are removed.

3.2 The baseline summarizers

We compare against two greedy baselines that use the SVR model of sentence importance, but not ILP. The first one, called GREEDY, uses the trained SVR model of the previous section to assign importance scores to all the sentences of the documents to be summarized. It then ranks the sentences by decreasing importance score and constructs the summary by iteratively selecting (and removing from the ranked list of sentences) the sentence with the highest importance score that fits in the summary space left.

The second baseline, called GREEDY-RED, operates in the same way, but it also takes into account redundancy. When a new sentence (with the highest importance score among the remaining sentences in the ranked list) is about to be added to the summary, its cosine similarity (computed on words) to all the sentences that have already been included in the summary is computed. If the similarity between the new and any of the already selected sentences exceeds a threshold t, the new sentence is discarded and a new iteration starts, where the next sentence of the ranked list of remaining sentences is considered. In our experiments, t was determined by tuning GREEDY-RED on development data (see below).

3.3 Our ILP summarization model

Instead of directly using the importance score $f_{SVR}(s_i)$ of each sentence s_i, as returned by the SVR model of Section 3.1, we normalize it using the maximum and mimimum values that the SVR model returns for the $j = 1, \ldots, n$ sentences of the input documents:

$$a_i = \frac{f_{SVR}(s_i) - \min_j f_{SVR}(s_j)}{\max_j f_{SVR}(s_j) - \min_j f_{SVR}(s_j)} \tag{14}$$

The objective (15) of our summarization ILP model sums the normalized relevance scores a_i of the selected sentences to estimate the overall importance $imp(S)$ of the resulting summary S. It also estimates the diversity $div(S)$ of S by calculating how many word bigrams of the documents being summarized are present in the selected sentences; when more bigrams are present in the summary, the summary's sentences share fewer bigrams, i.e., they are less redundant. Notice that we do not assign importance scores to the bigrams, unlike the work of Berg-Kirkpatrick et al. and Woodsend and Lapata (Section 2). The binary variables x_i

and b_j indicate which sentences s_i and which word bigrams g_j, respectively, are present in the summary; see Figure 1 for an example of the relations between the x_i and b_j variables.

$$\max_{b,x} \lambda_1 \cdot imp(S) + \lambda_2 \cdot div(S) = \max_{b,x} \lambda_1 \cdot \sum_{i=1}^{n} \frac{a_i}{k_{max}} \cdot x_i + \lambda_2 \cdot \sum_{j=1}^{|B|} \frac{b_j}{n} \tag{15}$$

$$\sum_{i=1}^{n} l_i \cdot x_i \leq L_{max} \tag{16}$$

$$\sum_{g_j \in B_i} b_j \geq |B_i| \cdot x_i, \text{ for } i = 1, \ldots, n \tag{17}$$

$$\sum_{s_i \in S_j} x_i \geq b_j, \text{ for } j = 1, \ldots, |B| \tag{18}$$

Again, l_i is the length of sentence s_i, L_{max} is the maximum allowed summary length, and n is the number of input sentences. $imp(S)$ is normalized to $[0, 1]$ using the maximum number of sentences k_{max} that can be included in the summary. To estimate k_{max} we divide the maximum available space L_{max} by the length of the shortest input sentence. We also divide $div(S)$ by n, which causes $div(S)$ to range mostly in $[0, 1]$ in our experiments. The values of λ_1 and λ_2 are tuned on development data. We set $\lambda_1 + \lambda_2 = 1$. Constraint 16 guarantees that L_{max} is not exceeded. The other two constraints are explained below:

- Constraint 17: B_i is the set of bigrams that appear in sentence s_i, $|B_i|$ is the cardinality of B_i, g_j ranges over the bigrams in B_i, and b_j is the binary variable that shows if bigram g_j has been selected. If a sentence s_i is selected ($x_i = 1$), then all of its bigrams must also be selected, i.e., $\sum_{g_j \in B_i} b_j = |B_i|$ and Constraint 17 holds. If sentence s_i is not selected ($x_i = 0$), then some of its bigrams may still be selected, if they occur in another selected sentence; hence $\sum_{g_j \in B_i} b_j \geq 0$ and Constraint 17 holds again.
- Constraint 18: Again, b_j is the binary variable that shows if g_j has been selected. $|B|$ is the total number of (distinct) bigrams of the n input sentences; S_j is the set of sentences that bigram g_j appears in; and x_i is the binary variable that shows if sentence s_i has been selected. If a bigram g_j is selected ($b_j = 1$), then at least one sentence that contains that bigram must also be selected; hence, $\sum_{s_i \in S_j} x_i \geq 1$ and Constraint 18 holds. If bigram g_j is not selected ($b_j = 0$), then none of the sentences that contain it may be selected; hence, $\sum_{s_i \in S_j} x_i = 0$ and Constraint 18 holds again.

In preliminary experiments, we noticed that our ILP model above, called ILP1, tended to select many short sentences, which had a poor ROUGE match with the gold summaries. To address this issue, we developed an alternative ILP model, called ILP2, whose objective function (19) rewards longer sentences by multiplying their importance scores a_i with their lengths l_i (in words). The constraints of ILP2 remain as in ILP1 (Constraints 16–18).

$$\max_{b,x} \lambda_1 \cdot \sum_{i=1}^{n} a_i \cdot \frac{l_i}{L_{max}} \cdot x_i + \lambda_2 \cdot \sum_{j=1}^{|B|} \frac{b_j}{n} \tag{19}$$

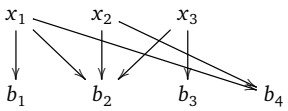


Figure 1: There are 3 sentences (corresponding to the binary variables x_1, x_2, x_3) containing 4 word bigrams (corresponding to variables b_1, b_2, b_3, b_4). For example, sentence s_1 contains the first, second, and fourth bigrams; and if sentences s_1 and s_2 are selected ($x_1 = 1$ and $x_2 = 1$), then the bigrams they contain must also be selected ($b_1 = 1$, $b_2 = 1$, $b_4 = 1$).

4 Experiments

We now present the experiments that we performed, starting from the datasets we used.

4.1 Datasets and experimental setup

We used the datasets of DUC 2005, DUC 2006, DUC 2007, and TAC 2008 (Dang, 2005, 2006; Dang and Owczarzak, 2008).[4] Each dataset contains document clusters. Each cluster contains documents relevant to a query (a question or topic description), which is also given. For each cluster, a summary not exceeding a maximum allowed length has to be produced, so that the summary will provide an answer to the corresponding query. Multiple reference (gold, human-authored) summaries are also provided per cluster. Table 1 provides more information on the datasets we used. For our experiments, we extracted all the sentences from the documents of each cluster, discarding sentences shorter than or equal to 7 words. We also applied a small set of cleanup rules to remove unnecessary formatting tags.

dataset	documents per cluster	clusters	reference summaries	word limit (in words)
DUC 2005	25–50	50	4–9	250
DUC 2006	25	50	4	250
DUC 2007	25	45	4	250
TAC 2008	10	48	4	100

Table 1: Datasets used in our experiments.

The SVR model of sentence importance (Section 3.1) was trained on the sentences of DUC 2006 (i.e., DUC 2006 was our training dataset) and it was used to assign importance scores to the sentences of the clusters of DUC 2005, DUC 2007, and TAC 2008. For each document cluster, we used the $n = 100$ sentences with the highest importance scores as input to the baseline and ILP summarizers of Sections 3.2 and 3.3.

We note that ILP problems are in the worst case (for the most difficult ILP problems) NP-hard. Our ILP1 and ILP2 models (Section 3.3) are generalizations of the 0-1 Knapsack problem, which is known to be NP-hard; hence, our models also constitute NP-hard problems. Nevertheless, very efficient ILP solvers are available.[5] In the worst case, the off-the-shelf solver that we use finds a solution (for ILP1 or ILP2, per summary) in 1.25 seconds and

[4] Consult also `http://duc.nist.gov/` and `http://www.nist.gov/tac/`.

[5] We use the implementation of the Branch and Cut algorithm of the GNU Linear Programming Kit (GLPK); consult `http://sourceforge.net/projects/winglpk/`.

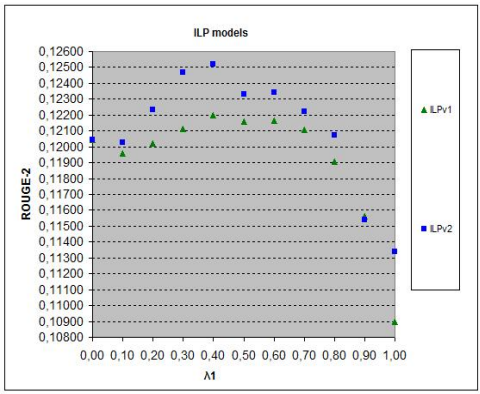


Figure 2: ROUGE-2 scores of the two versions of our ILP model on DUC 2007 data, used as development data, with the SVR model of sentence importance trained on DUC 2006 data.

0.9 seconds in the DUC 2007 and TAC 2008 datasets, respectively. The solver takes more time in DUC 2007 than in TAC 2008, because DUC 2007 summaries are longer (cf. Table 1) and, therefore, the search space is larger. This efficiency is mostly due to the fact that the x_i and b_j variables of ILP1 and ILP2 are in the order of hundreds and grow approximately linearly to the number and size (word bigrams) of the input sentences, as opposed to the quadratic (to the number of sentences) growth of the number of variables in McDonald's model (Section 2). Berg-Kirkpatrick et al. (2011) report very similar execution times; they report that the solver they use finds the solution of their extractive formulation in less than a second for most summaries of TAC 2008 and TAC 2009. Our method (ILP1 or ILP2) takes on average 10–11 seconds to form each summary, including the time to read and preprocess the input documents, formulate the ILP model etc. By contrast Woodsend and Lapata (2012) report that their method takes 55 seconds on average for each summary, though presumably this also includes parsing the input and applying the QSTG grammar.

4.2 Experiments on development data

To determine which of the two versions (ILP1 or ILP2) of our ILP model performs best and to tune their parameters, we experimented on the DUC 2007 dataset, i.e., we used the DUC 2007 dataset as our development data; recall that the DUC 2006 dataset was used as training data in all cases. We used 11 different values of λ_1 ($\lambda_2 = 1 - \lambda_1$) in both ILP1 and ILP2, and we evaluated the generated summaries using ROUGE-2. The results of these experiments are presented in Figure 2. ILP2 is better than ILP1 for all values of λ_1, and its best ROUGE-2 score is obtained for $\lambda_1 = 0.4$ ($\lambda_2 = 0.6$). The fact that the best results were obtained for non-zero λ_1 and λ_2 values also shows that both the sentence importance component (SVR) and the diversity component (bigram count) contribute to the results of our ILP models.

We also compared the average number of selected sentences per cluster of ILP1 and ILP2 on DUC 2007 data. As already noted and illustrated in Figure 3, ILP1 tends to select more and, therefore, shorter sentences than ILP2; these shorter sentences have worse ROUGE matches

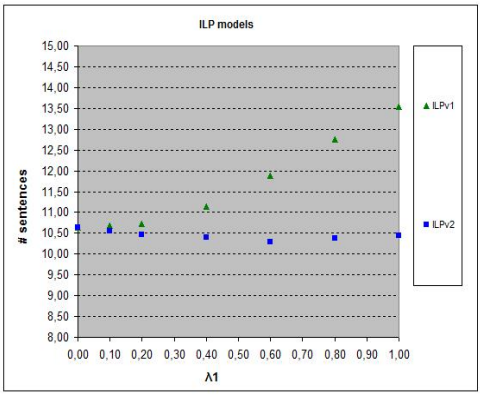


Figure 3: Average number of sentences selected by the two versions of our ILP model on DUC 2007 data, used as development data, with the SVR model trained on DUC 2006 data.

with the reference summaries, which is why ILP1 performs worse than ILP2. Figure 3 also shows that ILP2 selects approximately the same number of sentences for all λ_1 values; this is because ILP2 tends to always select relatively long sentences and, hence, the number of selected sentences that fit in the available space cannot vary as much as in ILP1.

In Table 2, we present the ROUGE scores of the two versions of our ILP model (for $\lambda_1 = 0.4$) on the DUC 2007 dataset, along with the corresponding scores of the GREEDY and GREEDY-RED baselines (Section 3.2), which use only the SVR without ILP. We also show the scores of several state of the art systems, both extractive and abstractive, as they were reported in the corresponding articles; more recently published results are shown first. Our ILP2 model has the best reported ROUGE-2 score on the DUC 2007 dataset, and the second best ROUGE-SU4 score, though one should keep in mind that the DUC 2007 dataset was our development set.

4.3 Experiments on test data

We then evaluated ILP2 with $\lambda_1 = 0.4$, which was our best system in the experiments on the development data (DUC 2007), against the systems with the highest published ROUGE scores on TAC 2008 and DUC 2005 data, our two test datasets.[6] The results of these experiments are listed in Tables 3 and 4, respectively. On the TAC 2008 dataset (Table 3), the most recent of the datasets we experimented with, our ILP2 method achieves the second best ROUGE-SU4 score and the second best ROUGE-2 score, following the method of Woodsend and Lapata with the QSTG grammar enabled, and the abstractive (full) method of Berg-Kirkpatrick et al., respectively (see Section 2). Our ILP2 method performs better than the method of Woodsend and Lapata *without* the QSTG grammar, even though the method of Woodsend and Lapata is still an abstractive one, even without the QSTG grammar (it can still delete parse tree nodes), whereas our method is purely extractive. If we exclude abstractive summarizers, our ILP2 method has the best ROUGE-2 and ROUGE-SU4 scores.

[6] We used Set A of TAC 2008.

system	ROUGE-2	ROUGE-SU4
ILP2	**0.12517**	0.17603
ILP1	0.12201	0.17283
GREEDY-RED	0.11591	0.16908
GREEDY	0.11408	0.16651
Lin and Bilmes 2011	0.12380	N/A
Celikyilmaz and Hakkani-Tur 2010	0.11400	0.17200
Haghighi and Vanderwende 2009	0.11800	0.16700
Schilder and Ravikumar 2008	0.11000	N/A
Pingali et al. 2007 (DUC 2007)	0.12448	**0.17711**
Toutanova et al. 2007 (DUC 2007)	0.12028	0.17074
Conroy et al. 2007 (DUC 2007)	0.11793	0.17593
Amini and Usunier 2007 (DUC 2007)	0.11887	0.16999

Table 2: Comparison of our ILP method against greedy baselines that use the same SVR model of sentence importance without ILP, and against other state of the art summarizers on DUC 2007 data (our development dataset). Our ILP method was trained on DUC 2006 data.

On the DUC 2005 dataset (Table 4), our ILP2 method has the best reported ROUGE-2 and ROUGE-SU4 scores. Berg-Kirkpatrick et al. and Woodsend and Lapata provide no results of their systems for this dataset. They also provide no results for the more recent TAC 2009 dataset, because they used it as their training set. We did not experiment with the TAC 2009 dataset, because our main competitors have not published results for that dataset.

We used paired t-tests ($p < 0.05$) to check if the differences between the scores of ILP2 and the other systems were statistically significant. In Tables 3–4, + and − denote the existence or absence of statistical significance, respectively. Unfortunately, the tests were possible only when comparing ILP2 against the few systems we had ROUGE scores for per topic.

5 Conclusions

We presented a new ILP method (in two versions) for multi-document summarization. Our method jointly maximizes the importance of the sentences it includes in a summary and their diversity, without exceeding a maximum allowed summary length. To obtain an importance score for each sentence, it uses an SVR model, trained on human-authored summaries to predict the ROUGE score of each sentence. Diversity is measured as the number of word bigrams of the input documents that occur in the resulting summary. Experimental results on widely used benchmarks for news summarization show that our ILP method achieves state of the art results among extractive summarizers. It also outperforms two greedy baselines that use the same SVR model of sentence importance without ILP, and it performs better than some abstractive summarizers. Our method is also very fast, and it does not require a parser or other resources that are not always available in less widely spoken languages.

We are already experimenting with an extended version of our method that also performs sentence compression. In future work, we hope to extend our ILP model to consider discourse coherence, sentence aggregation, and referring expression generation.

Acknowledgements

This research was funded by the Research Centre of the Athens University of Economics and Business.

system	ROUGE-2	ROUGE-SU4
ILP2	0.11168	0.14413
Woodsend and Lapata 2012 (with QSTG)	0.11370	**0.14470**
Woodsend and Lapata 2012 (without QSTG)	0.10320	0.13680
Berg-Kirkpatrick et al. 2011 (with subtree cuts)	**0.11700**	0.14380
Berg-Kirkpatrick et al. 2011 (without subtree cuts)	0.11050	0.13860
Shen and Li 2010	0.09012	0.12094
Gillick and Favre 2009 (with sentence compression)	0.11100	N/A
Gillick and Favre 2009 (without sentence compression)	0.11000	N/A
Gillick et al. 2008 (run 43 in TAC 2008)	0.11140^{-}	0.14298^{-}
Gillick et al. 2008 (run 13 in TAC 2008)	0.11044^{-}	0.13985^{-}
Conroy and Schlesinger 2008 (run 60 in TAC 2008)	0.10379^{-}	0.14200^{-}
Conroy and Schlesinger 2008 (run 37 in TAC 2008)	0.10338^{-}	0.14277^{-}
Conroy and Schlesinger 2008 (run 06 in TAC 2008)	0.10133^{+}	0.13977^{-}
Galanis and Malakasiotis 2008 (run 02 in TAC 2008)	0.10012^{+}	0.13694^{-}

Table 3: Comparison of our best ILP summarizer (ILP2) against state of the art summarizers on TAC 2008 data (one of our two test datasets). Our ILP method was trained on DUC 2006 data. It has the best ROUGE-2 and ROUGE-SU4 scores among extractive summarizers.

system	ROUGE-2	ROUGE-SU4
ILP2	**0.08174**	**0.13640**
Lin and Bilmes 2011	0.07820	N/A
Shen and Li 2010	0.07311	0.13061
McDonald 2007 (ILP)	0.06100	N/A
McDonald 2007 (Knapsack)	0.06700	N/A
Ye et al. 2005	0.0744^{+}	0.13461^{-}
Li et al. 2005	0.07313^{+}	0.13158^{-}
Daume and Marcu 2005	0.07089^{+}	0.12649^{+}

Table 4: Comparing our best ILP summarizer (ILP2) against state of the art summarizers on DUC 2005 data (one of our two test datasets). Our method was trained on DUC 2006 data.

References

Amini, M. R. and Usunier, N. (2007). A contextual query expansion approach by term clustering for robust text summarization. In *Proceedings of the Document Understanding Conference*, Rochester, NY.

Berg-Kirkpatrick, T., Gillick, D., and Klein, D. (2011). Jointly learning to extract and compress. In *Proceedings of the 49th Annual Meeting of the ACL-HLT*, pages 481–490, Portland, OR.

Celikyilmaz, A. and Hakkani-Tur, D. (2010). A hybrid hierarchical model for multi-document summarization. In *Proceedings of the 48th Annual Meeting of the ACL*, pages 815–824, Uppsala, Sweden.

Conroy, H. and Schlesinger, J. (2008). CLASSY and TAC 2008 Metrics. In *Proceedings of the Text Analysis Conference*, Gaithersburg, MD.

Conroy, H., Schlesinger, J., and O'Leary, D. (2007). CLASSY 2007 at DUC 2007. In *Proceedings of the Document Understanding Conference*, Rochester, NY.

Cormen, T., Leiserson, C., Rivest, R., and Stein, C. (2001). *Introduction to Algorithms*. MIT Press.

Dang, H. (2005). Overview of DUC 2005. In *Proceedings of the Document Understanding Conference*, Vancouver, B.C., Canada.

Dang, H. (2006). Overview of DUC 2006. In *Proceedings of the Document Understanding Conference*, Brooklyn, NY.

Dang, H. and Owczarzak, K. (2008). Overview of the TAC 2008 update summarization task. In *Proceedings of Text Analysis Conference*, Gaithersburg, MD.

Daume, H. and Marcu, D. (2005). Bayesian summarization at DUC and suggestion for extrinsic evaluation. In *Proceedings of the Document Understanding Conference*, Vancouver, B.C., Canada.

Galanis, D. and Malakasiotis, P. (2008). AUEB at TAC 2008. In *Proceedings of the Text Analysis Conference*, Gaithersburg, MD.

Gillick, D. and Favre, B. (2009). A scalable global model for summarization. In *Proceedings of the Workshop on Integer Linear Programming for Natural Language Processing*, pages 10–18, Boulder, CO.

Gillick, D., Favre, B., and Hakkani-Tur, D. (2008). The ICSI Summarization System at TAC 2008. In *Proceedings of the Text Analysis Conference*, Gaithersburg, MD.

Haghighi, A. and Vanderwende, L. (2009). Exploring content models for multi-document summarization. In *Proceedings of Human Language Technologies: The Annual Conference of the NAACL*, pages 362–370, Boulder, CO.

Levenshtein, V. (1966). Binary codes capable of correcting deletions, insertions, and reversals. *Soviet Physice-Doklady*, 10(8):707–710.

Li, W., Li, B., Chen, Q., and Wu, M. (2005). The Hong Kong Polytechnic University at DUC 2005. In *Proceedings of the Document Understanding Conference*, Vancouver, B.C., Canada.

Lin, C. (2004). ROUGE: A package for automatic evaluation of summaries. In *Proceedings of the ACL Workshop "Text Summarization Branches Out"*, pages 74–81, Barcelona, Spain.

Lin, H. and Bilmes, J. (2011). A class of submodular functions for document summarization. In *Proceedings of the 49th Annual Meeting of the ACL-HLT*, volume 1, pages 510–520, Portland, OR.

McDonald, R. (2007). A study of global inference algorithms in multi-document summarization. In *Proceedings of the European Conference on Information Retrieval*, pages 557–564, Rome, Italy.

Nishikawa, H., Hasegawa, T., Matsuo, Y., and Kikui, G. (2010a). Opinion summarization with integer linear programming formulation for sentence extraction and ordering. In *Proceedings of the 23rd International Conference on Computational Linguistics: Posters*, pages 910–918.

Nishikawa, H., Hasegawa, T., Matsuo, Y., and Kikui, G. (2010b). Optimizing informativeness and readability for sentiment summarization. In *Proceedings of the ACL Conference Short Papers*, pages 325–330, Uppsala, Sweden.

Pingali, P., Rahul, K., and Vasudeva, V. (2007). IIIT Hyderabad at DUC 2007. In *Proceedings of the Document Understanding Conference*, Rochester, NY.

Schilder, F. and Kondadadi, R. (2008). FastSum:Fast and Accurate Query-based Multi-document Summarization. In *Proceedings of the 46th Annual Meeting of the ACL-HLT: Short Papers*, pages 205–208, Columbus, OH.

Shen, C. and Li, T. (2010). Multi-document summarization via the minimum dominating set. In *Proceedings of the 23rd International Conference on Computational Linguistics*, pages 984–992, Beijing, China.

Toutanova, K., Brockett, C., Gamon, M., Jagarlamudi, J., Suzuki, H., and Vanderwende, L. (2007). The PYTHY summarization system: Microsoft Research at DUC 2007. In *Proceedings of the Document Understanding Conference*, Rochester, NY.

Tsochantaridis, I., Hofmann, T., Joachims, T., and Altun, Y. (2004). Support Vector Machine learning for independent and structured output spaces. *Machine Learning Research*, 6:1453–1484.

Vapnik, V. (1998). *Statistical Learning Theory*. John Wiley.

Woodsend, K. and Lapata, M. (2012). Multiple aspect summarization using integer linear programming. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 233–243, Jesu Island, Korea.

Ye, S., Qiu, L., Chua, T., and Kan, M. (2005). NUS at DUC 2005: Understanding documents via concept links. In *Proceedings of the Document Understanding Conference*, Vancouver, B.C., Canada.

Cross-Lingual Topical Relevance Models

Debasis Ganguly Johannes Leveling Gareth J.F. Jones
Centre for Next Generation Localisation (CNGL),
School of Computing, Dublin City University,
Dublin, Ireland
{dganguly, jleveling, gjones}@computing.dcu.ie

Abstract

Cross-lingual relevance modelling (CLRLM) is a state-of-the-art technique for cross-lingual information retrieval (CLIR) which integrates query term disambiguation and expansion in a unified framework, to directly estimate a model of relevant documents in the target language starting with a query in the source language. However, CLRLM involves integrating a translation model either on the document side if a parallel corpus is available, or on the query side if a bilingual dictionary is available. For low resourced language pairs, large parallel corpora do not exist and the vocabulary coverage of dictionaries is small, as a result of which RLM-based CLIR fails to obtain satisfactory results. Despite the lack of parallel resources for a majority of language pairs, the availability of *comparable* corpora for many languages has grown considerably in the recent years. Existing CLIR techniques such as cross-lingual relevance models, cannot effectively utilize these comparable corpora, since they do not use information from documents in the source language. We overcome this limitation by using information from retrieved documents in the source language to improve the retrieval quality of the target language documents. More precisely speaking, our model involves a two step approach of first retrieving documents both in the source language and the target language (using query translation), and then improving on the retrieval quality of target language documents by expanding the query with translations of words extracted from the top ranked documents retrieved in the source language which are thematically related (i.e. share the same concept) to the words in the top ranked target language documents. Our key hypothesis is that the query in the source language and its equivalent target language translation retrieve documents which share topics. The ovelapping topics of these top ranked documents in both languages are then used to improve the ranking of the target language documents. Since the model relies on the alignment of topics between language pairs, we call it the cross-lingual topical relevance model (CLTRLM). Experimental results show that the CLTRLM significantly outperforms the standard CLRLM by upto 37% on English-Bengali CLIR, achieving mean average precision (MAP) of up to 60.27% of the Bengali monolingual IR MAP.

Keywords: Cross-lingual Information Retrieval, Relevance Model, Topic Model, Pseudo-Relevance Feedback, Latent Dirichlet Allocation.

Proceedings of COLING 2012: Technical Papers, pages 927–942,
COLING 2012, Mumbai, December 2012.

1 Introduction

Cross-language information retrieval (CLIR) involves retrieving documents in a language (target language), different from the language in which the users formulate their search (source language). A simple low resourced bi-lingual dictionary-based query translation followed by monolingual IR does not yield satisfactory results in CLIR mainly due to the poor vocabulary coverage of such a low resourced dictionary and the inherent ambiguities in query term senses (Hull and Grefenstette, 1996). More complex methods of query translation, e.g. statistical machine translation (SMT), perform better, but not entirely satisfactorily, due to the lack of availability of parallel resources such as sentence aligned corpora between resource-poor language pairs (Nie et al., 1999).

Regional languages of India are typical examples of languages with poor linguistic resources. The dominance of English, which has been used extensively as a medium of instruction and official work, can be exemplified by the fact that, while the English Wikipedia has almost 4M documents, the number of documents in the Hindi and Bengali Wikipedia are only around 100K and 23K respectively, although Hindi and Bengali ranks fourth and sixth respectively in terms of the number of native speakers[1]. The multi-linguality of Indian culture provides an ample motivation for the study of CLIR, e.g. a native Indian language speaker would often prefer to type his query in English due to his acquaintance with the English keyboard, although seeking to retrieve documents in his native language. Querying in English to a regional Indian language is thus a widespread real-life potential application for CLIR. The major hindrance to developing effective Indian language CLIR is the lack of *parallel corpora*, i.e. sentence aligned manually translated texts, to enable the development of effective standard translation tools. *Comparable corpora*, i.e. non sentence aligned texts which are not exact translations of each other but are roughly on the same topic, however are abundantly available owing to the growth of digital text in regional Indian languages. News articles published from the same source or from the same location in both English and a regional language over an identical time period are examples of such comparable corpora. Thus, despite the scarcity of parallel resources, significant comparable corpora are available for English and many Indian languages. This motivates us to research into new techniques effectively exploit these corpora for enhanced CLIR performance. This paper introduces and evaluates our proposed method to do this.

The main idea of our work can be outlined as follows. We assume that there exists a comparable corpus of documents in both the language of the query (source language) and the target language in which the documents need to be presented to the user. A query in the source language is translated into the target language by any available resource which is typically a small bi-lingual dictionary for low resourced language pairs. Documents are then retrieved using both queries from the corresponding collections. It is a common practise in IR to improve upon the initial retrieval quality by utilizing information from the top-ranked documents (which are assumed to be relevant), the method being called pseudo-relevance feedback (PRF). PRF often does not work well when precision at top ranked documents is low. For our case, retrieval quality of the source language documents is expected to be much better than those of the target language ones, since the translated query in the target language is likely to be ambiguous and imprecise due to the lack of sufficient context in the short queries and the poor vocabulary coverage particularly for low resourced language pairs. Since the top ranked documents on the source side are more likely to be relevant to the information need, it can be hypothesized that utilizing this information for PRF can potentially improve retrieval results of the target language documents. Our proposed method thus relies on extracting translations of terms from the source language documents to expand the query

[1] http://en.wikipedia.org/wiki/Most_spoken_languages

in the target language so as to improve the retrieval quality on the target side.

The novelty of this paper is in the proposal of a PRF method which utilizes information from the source language documents to enhance retrieval effectiveness in the target language. The remainder of this paper is organized as follows. Section 2 describes the related work on CLIR and topic modelling. Section 3 describes the proposed method in details. Section 4 details the experimental setup, followed by Section 5, which evaluates the proposed method. Finally, Section 6 outlines the conclusions and directions for future work.

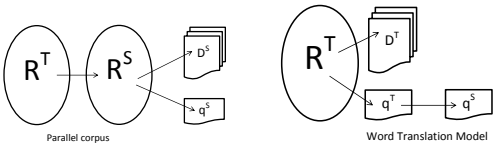
2 Related Work

This section starts with a brief review of the existing literature on general pseudo-relevance feedback (PRF). This is followed by a review on relevance models in IR, since our proposed method is a generalization of the cross-lingual relevance model (CLRLM). We then provide a brief survey on topic modelling applications in IR, as topic modelling is an integral part of our proposed method. Finally, we review existing work on combining PRF evidences in multiple languages, since our method involves a combination of pseudo-relevance information from the source and the target languages.

Pseudo-Relevance Feedback (PRF). PRF is a standard technique in IR which seeks to improve retrieval effectiveness in the absence of explicit user feedback (Robertson and Sparck Jones, 1988; Salton and Buckley, 1990). The key idea of PRF is that the top ranked initially retrieved documents are relevant. These documents are then used to identify terms which can be added to the original query followed by an additional retrieval run with the expanded query often involving re-weighting of the query terms (Robertson and Sparck Jones, 1988; Hiemstra, 2000), and re-ranking initially retrieved documents by recomputing similarity scores (Lavrenko and Croft, 2001).

Relevance modelling in (CL)IR. Relevance modelling (RLM) is a state-of-the-art PRF technique involving estimation of a model of relevance generating terms both from pseudo-relevant documents and the query terms (Lavrenko and Croft, 2001). Terms which co-occur frequently with query terms are assigned a high likelihood of being generated from the RLM. In addition to monolingual IR, RLM has also been applied successfully to cross-lingual IR (CLIR) (Lavrenko et al., 2002), under the name of cross-lingual relevance model (CLRLM). A limitation of CLRLM is that it depends either on a parallel corpus or on a bilingual dictionary. However for low resourced languages, parallel corpus seldom exist and dictionaries have poor vocabulary coverage. We address this limitation by exploiting the topical overlap of top ranked documents retrieved in the source and target languages to improve the relevance model estimation. Our method thus only requires a comparable corpus instead of the more stringent requirement of a parallel corpus.

Topic modelling applications in IR. A widely used technique for topic modelling is the latent Dirichlet allocation (LDA) which treats every document as a mixture of multinomial distributions with Dirichlet priors (Blei et al., 2003). Various inference techniques have been proposed to estimate the probabilities in LDA, including variational Bayes, expectation propagation, and Gibbs sampling (Blei et al., 2003; Griffiths and Steyvers, 2004). We use the Gibbs sampling method for LDA inference because it is computationally faster and has been shown to outperform the other two (Griffiths and Steyvers, 2004). LDA was applied for monolingual IR by (Wei and Croft, 2006). Their work involves estimating the LDA model for the whole collection by Gibbs sampling and then linearly combining the LM term weighting with LDA-based term weighting. An early attempt to utilize the topical structure in language pairs for CLIR can be found in (Littman et al., 1998), which involved automatic construction of a multi-lingual semantic space using a topic modelling technique



(a) Parallel corpus (b) Bilingual dictionary

Figure 1: Schematic diagrams of a CLRLM.

called latent semantic indexing (LSI). The major limitation of their method is that it relies on the existence of a parallel corpus for the training phase. In contrast, we leverage upon the existence of a comparable corpus to improve search results in the target language.

A recent work (Vulić et al., 2011) overcomes the parallel corpus restriction of CLRLMs by training a CLRLM on a comparable corpus using topic models inferred by bilingual LDA (Bi-LDA), which is a special case of polylingual LDA (Mimno et al., 2009). A major difference of their work with ours is that their method requires a separate training phase on comparable corpora to estimate the latent topic models in Bi-LDA. In fact, the authors used external resources such as Europarl and Wikipedia for the training purpose. In contrast, our method does not require a separate training phase or additional external resources. Moreover, our method applies topic modelling only on the set of pseudo-relevant documents. Furthermore, the idea of topic modelling in (Vulić et al., 2011) is only loosely coupled within the CLRLM framework, whereas we tightly integrate the topic modelling step in a graphical model with added nodes for the latent topics.

Feedback model combination. (Chen et al., 2010) exploited comparable corpora for CLIR by training *learning-to-rank* methods on out-of-domain source data to improve the retrieval effectiveness of the target domain. The Multi-PRF method improves monolingual retrieval by applying PRF in an assisting language, and mapping the feedback terms back in the language of the query with the help of a dictionary (Chinnakotla et al., 2010). The similarity of our method with Multi-PRF is that both involve an intermediate retrieval step in a language different from the language of the query. However, there are several differences which are highlighted as follows. Firstly, Multi-PRF improves monolingual retrieval by information from another language (typically English), whereas our proposed method improves CLIR performance. Secondly, Multi-PRF does not take into consideration the latent topics in the pseudo-relevant documents of the two languages, whereas topic modelling plays a crucial role in our approach.

3 Cross-lingual Topical Relevance Models

This section describes our proposed model in detail. We start the section with a brief motivation, where we discuss the limitations of CLRLM and how these can possibly be addressed. We then describe the schematics of our model which is then followed by the estimation details. Finally we present the algorithmic details for implementing the model.

3.1 Motivation

A limitation of CLRLM is that it depends either on a parallel corpus or on a bilingual dictionary to estimate the target language document models essential for estimating the relevance model for the query (source) language. The schematic diagrams of Figure 1a and 1b illustrate this. In the parallel corpus based approach, for every top ranked document retrieved in the source language

D_j^S, the corresponding document D_j^T in the target language is used to compute R^T, the estimated relevance model for the target language. This is shown in Figure 2a where the edge from $\mathbf{D}^T$ to $\mathbf{D}^S$ represents the event of transforming each document of the target language to its equivalent in the source language. The estimated probability of relevance is thus

$$P(w^T|\mathbf{q}^S) = \sum_{j=1}^{R} \underbrace{P(w^T|D_j^T)\, P(D_j^S|\mathbf{q}^S)}_{D_j^S \text{ parallel to } D_j^T} \tag{1}$$

where $P(D_j^S|\mathbf{q}^S)$ is the standard LM similarity of the given query $\mathbf{q}^S$ with a document D_j^S (Hiemstra, 2000). The parallel corpus based approach to CLRLM thus involves document side translation. A complementary approach is query side translation, as done in the bilingual dictionary-based CLRLM method shown in Figure 2b. The edge from $\mathbf{q}^T$ to $\mathbf{q}^S$ indicates generation of the query vector in the source language from a query vector in the target language via translation. The estimated probability of relevance in this case is given by

$$P(w^T|\mathbf{q}^S) = \sum_{j=1}^{R} P(w^T|D_j^T) P(D_j^T|\mathbf{q}^S) = \sum_{j=1}^{R} P(w^T|D_j^T) \prod_{i'=1}^{n^T} P(D_j^T|q_{i'}^T) \underbrace{\sum_{i=1}^{n^S} P(q_{i'}^T|q_i^S) P(q_i^S)}_{\text{word-based query translation}} \tag{2}$$

While it is possible to apply CLRLM in the presence of a bilingual dictionary, these dictionaries for low resourced languages cover only a very small part of the vocabulary, as a result of which it becomes impossible to compute the probabilities $P(s|t)$ for most (s, t) pairs (s and t refer to a word in the source language and the target language respectively). This in turn results in a poor performance of CLRLM for such low resourced languages.

The limitations of the current CLRLM method are that: a) it depends either on document translation with the help of a parallel corpus or on query translation with the help of a dictionary, without making provisions for a combination of the two; b) the document side translation depends on the availability of a parallel corpus which is rare for resource-poor languages; and c) it does not model the multiple aspects of relevance that are implicitly or explicitly expressed in a query. Assuming that there exists a comparable document collection in the two languages, the first two restrictions can be overcome with a two step retrieval process, one with the source language query and the other with a translated query in the target language to obtain separate working sets of documents in both the languages. The working set of the top ranked documents in the two languages, which

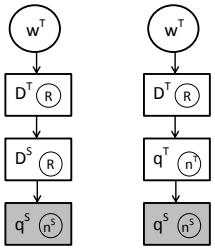


Figure 2: CLRLM dependence graphs for a) Parallel corpus (left) and b) Dictionary (right).

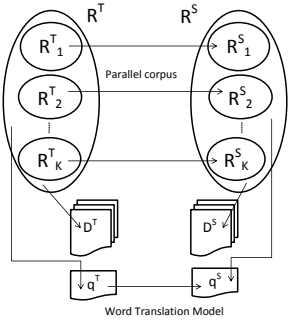


Figure 3: Schematic representation of a CLTRLM.

we refer to as pseudo-relevant document sets from now on, can thus potentially replace the parallel corpus requirement of the CLRLM. However, it is impractical to assume a one to one document level alignment between the pseudo-relevant documents in the two languages. Segmenting the pseudo-relevant documents into topics can result into a more accurate alignment at the level of topics rather than at the level of documents, with the underlying hypothesis that the documents retrieved in the two languages comprise of words related to overlapping concepts or topics. Topic modelling on the pseudo-relevant sets of documents also help in overcoming the third limitation where each topic addresses one particular sub information need associated with the query.

Our proposed methodology which we call cross-lingual topical relevance model (CLTRLM), involves estimation of two separate relevance models for both the source and target language pseudo-relevant documents. The target language relevance model is then updated by applying a word translation model to transform a word from each topic in the source language to a word in the target language. CLTRLM aims to achieve the benefits of a parallel corpus without actually having one. A topic level decomposition and mapping is helpful in adding contributions from the most likely topic, i.e. aspect of relevance, from the source language to a topic on target side. Note that the documents on which we apply topic modelling are the top ranked documents retrieved in response to the query in both the source language and its target language translation. Both document sets being focused on the query ensures that these documents share common topics. This is contrary to the approach of (Vulić et al., 2011), where the full comparable corpus is used for topic modelling. The working principle of a CLTRLM is illustrated schematically in Figure 3, which shows that the relevance models for both the source and the target languages have been split into topics. Each topic in a relevance model may refer to an individual aspect of the broad information need expressed in the query, and is thus associated with generating relevant documents related to that particular aspect. In contrast to the broad information need of a query, each particular aspect is more focused, and is thus easier to align from the target side to the source side. Although Figure 3 shows that the number of topics in the source and the target relevance models are identical, the number of topics may in fact be different on the source and the target sides. The next section presents the details of CLTRLM more formally.

3.2 Formal Description

Figure 4 shows the dependence network of CLTRLM in plate notation. Let w^T be a word in the target language. The top-most circle in Figure 4 represents a word in the target language for which

the objective is to calculate the probability $P(w^T|\mathbf{q}^S)$, i.e. to estimate the probability of generating this word from a hypothetical relevance model R^T. It is not possible to estimate this model directly because in ad-hoc retrieval, no prior information is provided about the relevant documents. The only observable entities are the query terms in the source language as shown in Figure 4. Let us denote this query by $\mathbf{q}^S$. The shaded rectangle at the bottom represents the vector q_i^S of observed variables having dimensionality n^S, each component representing the term frequency of a unique query term in the source language. The best way to estimate the probability $P(w^T|R^T)$ is thus to approximate it with $P(w^T|\mathbf{q}^S)$ i.e. $P(w^T|R^T) \approx P(w^T|\mathbf{q}^S)$. The rectangles $\mathbf{z}^T$ and $\mathbf{z}^S$ denote vectors of dimensionality K^T and K^S, the number of topics on the target and source sides respectively. While estimating the model, we assume that the topics for both the source and target langages have been obtained by latent Dirichlet allocation (LDA) (Blei et al., 2003), and thus we use the LDA estimated values viz. $\hat{\boldsymbol{\theta}}^T$, $\hat{\boldsymbol{\phi}}^T$, $\hat{\boldsymbol{\theta}}^S$ and $\hat{\boldsymbol{\phi}}^S$ in CLTRLM inference. The rectangles marked by $\mathbf{D}^T$ and $\mathbf{D}^S$ denote the set of top ranked R^T and R^S documents retrieved respectively for the target and source languages. $\mathbf{q}^T$ represents the translation of the observed query vector in the source language viz. $\mathbf{q}^S$, and is obtained by a bilngual dictionary or using a machine translation (MT) system. With these notations, we are now ready to work out the estimation details in the following subsection.

3.3 Estimation Details

With reference to Figure 4, the estimation of CLTRLM proceeds as follows.

$$\begin{aligned} P(w^T|\mathbf{q}^S) = & \underbrace{P(w^T|\mathbf{z}^T)P(\mathbf{z}^T|\mathbf{q}^S)}_{\text{target language generation event}} + \underbrace{P(w^T|\mathbf{w}^S)P(\mathbf{w}^S|\mathbf{q}^S)}_{\text{source language generation event}} \\ & = \sum_{k=1}^{K^T} P(w^T|z_k^T, \hat{\phi^T}_{k,w^T})P(z_k^T|\mathbf{q}^S) + \sum_{i=1}^{t(w^T)} P(w^T|w_i^S)P(w_i^S|\mathbf{q}^S) \end{aligned} \tag{3}$$

Equation (3) represents two chains of events via the two components shown: one associated with the target language retrieved documents obtained by query translation, and the other associated with $t(w^T)$ possible translations of word w^T in the source language (denoted by the set w^S), which in turn correspond to the words in the source language retrieved documents. Note that the two generation events pertain to the topic level alignment introduced informally in the previous section, where a word in the source language query can either be generated from a topic in the source language or from an *equivalent* topic in the target language. Inferencing along the left chain proceeds as follows.

$$\begin{aligned} & P(z_k^T|\mathbf{q}^S) = \sum_{j=1}^{R^T} P(z_k^T|D_j^T)P(D_j^T|\mathbf{q}^S) = \sum_{j=1}^{R^T} P(z_k^T|D_j^T) \sum_{i'=1}^{n^T} P(D_j^T|q_{i'}^T)P(q_{i'}^T|\mathbf{q}^S) = \\ & \sum_{j=1}^{R^T} P(z_k^T|D_j^T) \sum_{i'=1}^{n^T} P(D_j^T|q_{i'}^T) \sum_{j'=1}^{n^S} P(q_{i'}^T|q_{j'}^S)P(q_{j'}^S) = \sum_{j=1}^{R^T} P(z_k^T|D_j^T, \hat{\theta}_{j,k}^T) \underbrace{\sum_{i'=1}^{n^T} \frac{P(q_{i'}^T|D_j^T)}{R^T}}_{\text{LM similarity}} \underbrace{\sum_{j'=1}^{n^S} P(q_{i'}^T|q_{j'}^S)}_{\text{query translation}} \end{aligned} \tag{4}$$

In the last line of Eq. (4) we have ignored the prior probability of $P(q_{j'}^S)$, and used the LDA estimated $\hat{\boldsymbol{\theta}}^T$ values for computing $P(z_k^T|D_j^T)$ and the standard LM similarity score $P(q_{i'}^T|D_j^T)$ to compute the probability of generating a target language query term from a target language document model.

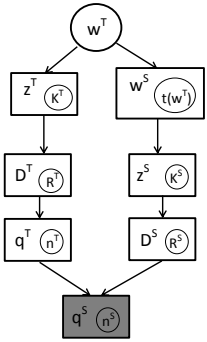


Figure 4: CLTRLM dependence graph in plate notation.

Similarily, the right side chain can be inferred as

$$P(w_i^S|\mathbf{q}^S) = \frac{1}{R^S}\underbrace{\sum_{k=1}^{K^S} P(w_i^S|z_k^S,\hat{\boldsymbol{\phi}}^S)\sum_{j=1}^{R^S} P(z_k^S|D_j^S,\hat{\boldsymbol{\theta}}^S)}_{\text{LDA document model of } D_j^S}\underbrace{P(D_j^S|\mathbf{q}^S)}_{\text{LM similarity}} = \sum_{j=1}^{R^S}\frac{P_{LDA}(w_i^S,D_j^S,\hat{\boldsymbol{\theta}}^S,\hat{\boldsymbol{\phi}}^S)P(D_j^S|\mathbf{q}^S)}{R^S} \tag{5}$$

where we have used the notation $P_{LDA}(.)$ to denote the LDA estimated probabilities marginalized over the latent topic variables. Substituting Eq. (4) and Eq. (5) into (3) gives the full expression of the probability of generating a target language word from the relevance model in case of CLTRLM.

$$P(w^T|\mathbf{q}^S) = \underbrace{\left(\sum_{k=1}^{K^T} P(w^T|z_k^T,\hat{\phi}_{k,w^T}^T)\sum_{j=1}^{R^T} P(z_k^T|D_j^T,\hat{\theta}_{j,k}^T)\sum_{i'=1}^{n^T}\frac{P(q_{i'}^T|D_j^T)}{R^T}\sum_{j'=1}^{n^S} P(q_{i'}^T|q_{j'}^S)\right)}_{\text{target language contribution estimated by query translation}} + \underbrace{\left(\sum_{i=1}^{t(w^T)} P(w^T|w_i^S)\times\sum_{j=1}^{R^S} P_{LDA}(w_i^S,D_j^S,\hat{\boldsymbol{\theta}}^S,\hat{\boldsymbol{\phi}}^S)\frac{P(\mathbf{q}^S|D_j^S)}{R^S}\right)}_{\text{source language contribution estimated by document word translation}} \tag{6}$$

In Equation 6, $\hat{\phi}_{k,w^T}^T$ denotes the probability of the word w^T belonging to the topic k, whereas $\hat{\theta}_{j,k}^T$ denotes the probability of the k^{th} topic in the j^{th} document. Both these quantities can be computed from the LDA estimation output matrices $\boldsymbol{\theta}^T$ and $\boldsymbol{\phi}^T$.

Also note that the CLTRLM as shown in Figure 4 involves two possible ways of generating the query in the source language, i.e. either directly using documents retrieved in the target language, or by using translations of words in documents retrieved in the source language. Thus, a natural question which arises is whether we need to introduce a new linear combination parameter to choose the two event paths with relative weights similar to (Chinnakotla et al., 2010). However, a closer look at Equation 3 reveals that the contribution from each path is inherently controlled by the two coefficients $P(w^T|\mathbf{z}^T)$ and $P(w^T|\mathbf{w}^S)$, thus eliminating the need for an extra parameter.

3.4 Estimation with Bi-LDA

In Section 3.3 we worked out the estimation details assuming that the source and the target language documents have different topic distributions denoted by the parameters $\hat{\boldsymbol{\theta}}^S$ and $\hat{\boldsymbol{\theta}}^T$ respectively. The CLTRLM estimation can also be performed with a stronger assumption that document pairs in the source and target languages share the same distribution of topics say $\hat{\boldsymbol{\theta}}$, with different topic-word distribution parameters say $\hat{\boldsymbol{\phi}}^S$ and $\hat{\boldsymbol{\phi}}^T$ respectively. This is a special case of polylingual LDA as proposed in (Mimno et al., 2009). For Bi-LDA estimation of CLTRLM, firstly we impose the restriction of $R^T = R^S$, i.e. to retrieve the same number of documents on the source and target sides, and secondly we set $\hat{\boldsymbol{\theta}} = \hat{\boldsymbol{\theta}}^T = \hat{\boldsymbol{\theta}}^S$ in Equation 6. We refer to CLTRLM instances with Bi-LDA estimation as JCLTRLM (Joint CLTRLM). Note that a JCLTRLM has two parameters $R = R^T = R^S$ and $K = K^T = K^S$ as opposed to four of CLTRLM.

3.5 Algorithm

After presenting the estimation details, we now provide the implementation steps for CLTRLM.

1) Run initial retrieval on the source language query $\mathbf{q}^S$ using standard LM to obtain documents $\{D_j^S\}_{j=1}^R$ in the source language (Ponte, 1998) and let R^S be the number of top ranked documents assumed to be pseudo-relevant.
2) Use a source-target language dictionary to get the equivalent query $\mathbf{q}^T$ in the target language.
3) Retrieve documents $\{D_j^T\}_{j=1}^R$ using LM for the target language query $\mathbf{q}^T$, and assume that the top ranked among these are pseudo-relevant.
4) Perform LDA inference by N iterations of Gibbs sampling on the working sets $\{D_j^S\}_{j=1}^{R^S}$ and $\{D_j^T\}_{j=1}^{R^T}$ to estimate the parameters $\hat{\boldsymbol{\theta}}^S$, $\hat{\boldsymbol{\phi}}^S$, $\hat{\boldsymbol{\theta}}^T$ and $\hat{\boldsymbol{\phi}}^T$. For the case of JCLTRLM, use Bi-LDA to estimate parameters.
5) Let V^T be the vocabulary of $\{D_j^T\}_{j=1}^{R^T}$. For each word $w^T \in V^T$, use Eq. (6) to compute the probability of relevance $P(w|R^T) \approx P(w|\mathbf{q}^S)$.
6) Rerank every target language document $\{D_j^T\}_{j=1}^{R^T}$ by the KL divergence between its LM document model (as obtained by the initial retrieval) and the estimated $P(w^T|R^T)$ so as to get the final retrieval result.

The computational complexity of the above algorithm is $O((V^T + V^S)(R^T + R^S)(K^T + K^S)N)$ where V^T, V^S are the vocabulary sizes of $\{D_j^T\}_{j=1}^{R^T}$ and $\{D_j^S\}_{j=1}^{R^S}$ respectively, R^T and R^S is the number of pseudo-relevant documents, K is the number of topics, and N is the number of iterations used for Gibbs sampling. The computational complexity of CLRLM on the other hand is $O(V^T R^T)$. CLTRLM, as compared to CLRLM, has the added computational cost for the source language retrieved documents. However, it is expected that $V^S = O(V^T)$, $R^S = O(R^T)$, and that both K^T, K^S and N are small constant numbers independent of R^T and V^T. Thus, CLTRLM is only a constant times more computationally expensive than CLRLM.

4 Experimental Setup

In this section, we describe the details of our experimental setup for evaluating CLTRLM. Our experiments explore the following questions: a) Does integrating the event of generating a target language word from the source language lead to a better estimation of the generative model for relevance compared to CLRLM? b) Does the use of latent topics benefit the alignment between source and target language documents and thus leads to a better estimation of relevance? c) What is the effect of translation quality on the performance of CLTRLM estimation? and d) How does the performance of Bi-LDA estimation for JCLTRLM, and separate LDA estimation for CLTRLM compare against each other? To answer a), we compare CLTRLM against CLRLM on cross-lingual ad-hoc search. To explore question b), we instantiate CLTRLM with the number of topics on the

	Documents			Queries			
Language	Type	# Docs.	Vocab. size [words]	Field	# Queries	Avg. query len.	Avg. # rel. docs.
Bengali	News articles	123,048	641,308	title	50	3.6	13.6
English	News articles	125,586	318,922	title	50	4.9	10.2

Table 1: FIRE-2010 document and query characteristics.

source and target sides set to 1, i.e. we use $K^S = K^T = 1$ in Equation 6, and use this instantiation of CLTRLM as one of our baselines. To answer c), we obtain different translation qualities by applying a bilingual dictionary and Google translate[2], which is a free to use statistical machine translation service. The presence of OOV words across language pairs can impair the estimation quality of CLTRLM because for every target language word whose translation is not found in the dictionary, we fail to get the source language contribution in the generation probability (see Equation 3). To reduce the vocabulary gap we use transliteration of OOV words, since it has been reported that transliteration helps improve the retrieval quality of CLIR (Udupa et al., 2009). Finally to address question d), we evaluate the relative performance of CLTRLM and JCLTRLM.

Data set. We perform CLIR experiments from English to Bengali, i.e. the query is expressed in English (source language), and the objective is to retrieve documents in Bengali (target language). Experiments are conducted on the English and Bengali ad-hoc collections of FIRE-2010 dataset (Majumder et al., 2010), the documents of which comprise a comparable corpus of news articles published from the same news agency in Calcutta namely *Ananadabazar* and *The Telegraph* in Bengali and English respectively. Table 1 outlines the document collection and query characteristics. Note that we do not use any external parallel or comparable resources to train our model as was done in (Vulić et al., 2011; Littman et al., 1998; Chen et al., 2010).

Stopwords. The stopword list used for Bengali retrieval was the one provided by the track organizers[3] generated by following an approach of extracting the N most frequent words from the document collection (Fox, 1992; Savoy, 1999). This list comprises of 384 Bengali words. The stopword list used for English was the standard SMART stopword list which comprises of 573 words.

Stemming. We used a moderately aggressive rule-based stemmer[4] for Bengali retrieval (Leveling et al., 2010). The stemmer used for our experiments is able to increase mean average precision (MAP) by 21.82% (0.2250 to 0.2741) on the monolingual title-only Bengali queries of the FIRE-2010 test collection. Although there are more complex corpus-based approaches reported for Bengali stemming (Majumder et al., 2007; Paik et al., 2011), the focus of this paper is not on improving stemming, but rather to improve on cross-lingual retrieval performance from English to Bengali. We thus applied a simple rule based approach as a stemmer which does not require computationally intensive pre-processing over the vocabulary of the corpus. The stemmer used on the English side is the default stemmer of SMART, a variant of Lovin's stemmer (Lovins, 1968).

Translation. One of the major components of CL(T)RLM is the bilingual dictionary to translate a given query in the source language viz. $\mathbf{q}^S$ to the corresponding representation in the target language $\mathbf{q}^T$. In our experiments, we used the open source English-Bengali dictionary *Ankur*[5]. The dictionary

[2] http://translate.google.com/#en|bn|
[3] http://www.isical.ac.in/~fire/stopwords_list_ben.txt
[4] http://www.computing.dcu.ie/~dganguly/rbs.tar.gz
[5] http://www.bengalinux.org/english-to-bengali-dictionary/

Translation Method	BLEU	
	Unstemmed	Stemmed
Dictionary	6.13	6.71
Dictionary + Google transliterate	7.46	**8.79**
Google translate	6.50	7.61
Google translate + Google transliterate	7.47	8.64

Table 2: English-Bengali query translation qualities.

at the time of writing this paper comprises of 9180 English words for each of which one or more Bengali word translations are provided. It can be seen that the vocabulary coverage of the *Ankur* English-Bengali dictionary is very small, in fact covering only 1.43% of the total vocabulary size of the English corpus (see Table 1), as a result of which a significant number of query words remain untranslated and hence play no role in estimating the CLTRLM. To increase the vocabulary coverage, and hence improve on the retrieval quality, we used Google translator, which is a statistical MT web-service, to translate the English queries to Bengali.

Transliteration. The out of vocabulary (OOV) words with respect to both the Google translator and the English-Bengali dictionary were left in the original English form. A manual inspection of these translated queries by one of the authors, who is a native Bengali speaker, revealed that most of these OOV words are English proper nouns. Proper nouns are important for retrieval (Xu and Croft, 2000), and thus need to be handled appropriately. An intuitive approach is to transliterate English names into Bengali which has proved to be beneficial for Indian language CLIR (Udupa et al., 2009). For transliteration, we applied *Google transliterate*[6] on the untranslated words of the Bengali queries as obtained by the dictionary-based and the Google translator approaches. Google transliterate returns five transliterations for each given word in decreasing order of likelihood. Out of these five Bengali transliterations for each English word, we use the top ranked one i.e. the most likely one. This simplistic approach of taking the most probable candidate from Google transliterate may not yield accurate transliterations. However, the focus of the paper is not to improve on the English-Bengali transliteration process itself, but rather to use transliteration as an intermediate tool to improve CLIR performance. Furthermore, an incorrect transliteration of a query term hardly has any effect on retrieval performance, since it is highly unlikely for an incorrectly transliterated word to match with the indexed vocabulary.

Table 2 shows the quality of the query translations obtained by the four methods. Translation quality of the English queries translated into Bengali is measured by the metric BLEU (Papineni et al., 2002), by using FIRE-2010 Bengali query titles as reference translations. We in fact report two versions, one which computes the BLEU score using the original word forms, and the other on the stemmed versions of the translated text and reference. The latter is a more appropriate measure for CLIR, because for the case of CLIR it is sufficient to match the stemmed forms rather than matching corresponding original word forms of a translated word with its reference. It can be seen that the BLEU scores are rather low compared to language pairs such as English-French (Dandapat et al., 2012), which is indicative of the fact that translation from English-Bengali is a difficult problem indeed. Application of transliteration however results in a significant improvement of BLEU score, indicating the importance of handling the OOV words.

[6] https://developers.google.com/transliterate/

CL(T)RLM Implementation. CLTRLM has been implemented as an extension to SMART[7] along with the CLRLM approach which is used as one of the baselines. The other baseline using Google translator involves estimating a monolingual relevance model (Lavrenko and Croft, 2001) which is also implemented in SMART. GibbsLDA++[8] was employed for Gibbs sampling for LDA inference. A modified version of GibbsLDA++ was used for Bi-LDA inference for JCLTRLM estimation.

5 Results

This section reports the results of the experiments and analyzes the observations. We start with a detailed description of the retrieval runs and parameter settings.

Retrieval run description. The CLRLM baselines shown in Table 3 start with the prefix "CLRLM". Results are shown for each method of translation (Google translator or the dictionary-based translation), with or without transliteration on the translation output, thus yielding 4 cases. The CLRLM approach does not use any information from the documents retrieved in the source language. To show that source language information is beneficial for retrieval, we report runs which use only the target language path in the generative model shown in Figure 4, i.e. for these runs we set $P(w^T|\mathbf{w}^S)$ (see Equation 3) to zero. These runs are shown the prefix "TgtCLTRLM" in Table 3. To show that topic decomposition is essential, we set the number of topics to 1 on both the source and the target sides, i.e. we set K^T and K^S to 1. These runs benefit from the information on the source side, but do not use the topical structures of documents to help achieve a fine grained alignment of the topics. These run names are prefixed with "UCLTRLM" in Table 3. Finally, we report (J)CLTRLM results prefixed with (J)CLTRLM.

Parameter settings. Each run reported in Table 3 has been optimized with the best parameter settings. The parameters were varied as follows. The Jelinek-Mercer language modelling smoothing parameter (Hiemstra, 2000) of initial retrieval for all the runs was empirically chosen as 0.4. The hyper-parameters α and β which control the Dirichlet distributions for CLTRLM, were set to $\frac{50}{K}$ (K being the number of topics) and 0.1 respectively as described in (Griffiths and Steyvers, 2004). The number of iterations for Gibbs sampling i.e. N, was set to 1000 for all CLTRLM experiments. We tuned the common parameters i.e. R^S and R^T, i.e. the number of top ranked documents used for pseudo-relevance in the source and target languages respectively, within the range of $[10, 50]$ in steps of 10 so as to obtain the best settings. An important parameter to CLTRLM is the number of topics on the source and target sides viz. K^S and K^T. These parameters were empirically optimized within the range of $[5, 50]$. The justification of using a much smaller value range for the number of topics, in comparison to the global LDA based approach (Wei and Croft, 2006) which used much higher values of K in the range of 100 to 1500, comes from the fact that LDA estimation in the CLTRLM is done on only a small number of documents rather than on the full corpus.

Observations. With reference to Table 3, it can be seen that the initial retrieval run NOFDBK-DICT performs very poorly achieving only 21.58% of the MAP as compared to the monolingual retrieval run NOFDBK-MONOLINGUAL. This shows that a low resourced dictionary-based query translation does not yield satisfactory retrieval performance. A cross-lingual relevance model based feedback approach is able to significantly[9] improve on the initial retrieval MAP by 42.40% as can be seen by comparing run CLRLM-DICT with NOFDBK. The CLRLM run only retrieves documents in the target language i.e. Bengali in this case. MAP is further improved by 10.91% by using documents retrieved in the source language i.e. English as seen from the run UCLTRLM-DICT

[7] `ftp://ftp.cs.cornell.edu/pub/smart/`

[8] `http://gibbslda.sourceforge.net/`

[9] Statistical *significance* or statistically *(in)distinguishable* henceforth refer to Wilcoxon test with 95% confidence measure.

	Query Processing		Parameters					Results	
Approach	Translation	Transliteration	PRF	R^T	R^S	K^T	K^S	MAP	P@5
NOFDBK-DICT	Dictionary	N	N	-	-	-	-	0.0592	0.0653
CLRLM-DICT	Dictionary	N	Y	30	-	1	-	0.0843	0.1102
UCLTRLM-DICT	Dictionary	N	Y	30	20	1	1	0.0935	0.1224
TgtCLTRLM-DICT	Dictionary	N	Y	30	0	15	0	0.1069	0.1388
CLTRLM-DICT	Dictionary	N	Y	30	20	15	5	**0.1130**	**0.1592**
JCLTRLM-DICT	Dictionary	N	Y	10	10	10	10	0.1086	0.1551
NOFDBK-DICT-TLIT	Dictionary	Y	N	-	-	-	-	0.0996	0.1120
CLRLM-DICT-TLIT	Dictionary	Y	Y	30	-	1	-	0.1156	0.1440
UCLTRLM-DICT-TLIT	Dictionary	Y	Y	30	20	1	1	0.1237	0.1560
TgtCLTRLM-DICT-TLIT	Dictionary	Y	Y	20	0	10	0	0.1369	0.1400
CLTRLM-DICT-TLIT	Dictionary	Y	Y	20	5	10	5	0.1446	0.1720
JCLTRLM-DICT-TLIT	Dictionary	Y	Y	10	10	20	20	**0.1588**	**0.1800**
NOFDBK-SMT	Google	N	N	-	-	-	-	0.0843	0.1080
CLRLM-SMT	Google	N	Y	30	-	1	-	0.1208	0.1520
UCLTRLM-SMT	Google	N	Y	30	20	1	1	0.1274	0.1640
TgtCLTRLM-SMT	Google	N	Y	30	0	15	0	0.1373	**0.1840**
CLTRLM-SMT	Google	N	Y	30	20	15	5	0.1425	0.1800
JCLTRLM-SMT	Google	N	Y	30	30	10	10	**0.1441**	0.1800
NOFDBK-SMT-TLIT	Google	Y	N	-	-	-	-	0.1024	0.1240
CLRLM-SMT-TLIT	Google	Y	Y	30	-	1	-	0.1393	0.1720
UCLTRLM-SMT-TLIT	Google	Y	Y	30	20	1	1	0.1483	0.1840
TgtCLTRLM-SMT-TLIT	Google	Y	Y	30	0	10	0	0.1523	0.1680
CLTRLM-SMT-TLIT	Google	Y	Y	30	20	10	5	0.1648	**0.2000**
JCLTRLM-SMT-TLIT	Google	Y	Y	20	20	5	5	**0.1652**	0.1920
NOFDBK-MONOLINGUAL	-	-	N	-	-	-	-	0.2741	0.3160

Table 3: Results for English-Bengali CLIR experiments.

in comparison to CLRLM-DICT. This run is a corner-case of a CLTRLM without using topical decompositions on the source and target sides i.e. by setting $K^T = K^S = 1$. Topical decomposition on the target side alone (see "TgtCLTRLM" prefixed runs) produces better results than the CLRLM runs, but are outperformed by the runs which use information from the source side as well, as shown by the (J)CLTRLM runs. It can be seen that both topical decomposition and using information from the source-side play a crucial role in correctly estimating the relevance model.

It turns out that Bi-LDA inference of CLTRLM, i.e. JCLTRLM performs better than separately inferencing the topic models on the source and target sides for all scenarios except the one which only uses the dictionary. The reason JCLTRLM performs poorly on dictionary-based approach is that the initial retrieval results on the target language is very poor (MAP: 0.0592). It is thus not a reasonable assumption that the document pairs on the source and target sides share the same topic distribution $\hat{\boldsymbol{\theta}}$. In such a scenario, it is helpful to use different number of documents on the source and target sides for the LDA estimation. However, when the initial retrieval quality improves on the target side with the help of transliteration or SMT or both, it is observed that Bi-LDA estimation proves more beneficial because of a better level of agreement between the pseudo-relevant document sets in the source and target languages. The improvements obtained by JCLTRLM over CLTRLM

are statistically indistinguishable with respect to MAP, except for the DICT-TLIT case.

To summarize, we observe that the CLTRLM runs (even with only one topic), using information from the source side, perform significantly better than CLRLM runs which in turn do not use any source information, thus indicating that *documents retrieved in the source language lead to a better relevance estimation*. This observation conforms to the Multi-PRF results where it was shown that using information from another language retrieval results helps improve the monolingual retrieval result in the language of the query (Chinnakotla et al., 2010). The CLTRLM approach achieves a similar benefit for CLIR. We also observe that the CLTRLM runs outperform the UCLTRLM prefixed runs, thus establishing that *the use of latent topics does benefit the alignment between source and target language documents*. The reason is that incorporating latent topic nodes in the estimation process helps, firstly in focusing co-occurrence computation on topics rather than on full documents, and secondly the alignment process of the fine grained topics is more reliable than aligning full documents in the source and target languages. The fact that CLTRLM improves retrieval effectiveness significantly over a range of query translation qualities, suggests that the method is robust to translation quality. Furthermore, JCLTRLM turns out to be slightly better than CLTRLM suggesting that Bi-LDA estimation is marginally better than separate LDA estimation.

Comparison to other reported results on Indian language CLIR. To the best of our knowledge, no results have been reported for fully automatic English-Bengali CLIR. (Leveling et al., 2010) report that manual English-Bengali query translation by native Bengali speakers achieves MAP up to 83.3% compared to monolingual Bengali retrieval, but on longer *TD* (title-description) queries. The fact that manual query translation by native Bengali speakers achieves 83.3% retrieval effectiveness in comparison to monolingual IR, demonstrates that English-Bengali CLIR is a considerably hard problem to solve. Our fully automatic approach achieves a satisfactory performance increasing MAP by 90.87% (see the rows CLTRLM-DICT and NOFDBK-DICT) compared to translation with a base dictionary, and achieves a MAP of 60.27% of the monolingual upper baseline.

6 Conclusions and Future work

This paper presented CLTRLM, a novel theoretical framework for exploiting the topical association of terms between pseudo-relevant documents of the source and target languages, to improve CLIR effectiveness. CLTRLM is a generalization of the standard cross-lingual relevance model, overcoming its limitations of: a) incompatibility with a comparable corpus; and b) co-occurrence computation at the level of whole documents, instead of likely relevant topics. CLTRLM uses a *comparable* corpus for IR on the retrieved set of top-ranked documents without the requirement of a separate training phase on the whole corpus. Empirical evidence of the effectivness of the method, specially on low resourced languages, is provided by achieving 41% (MAP: 0.1130) with a base dictionary of about 10K words, and 60.27% (MAP: 0.1652) with freely available SMT web-services of the monolingual MAP on English-Bengali CLIR (MAP: 0.2741).

The work presented in this paper treats an entire pseudo-relevant document as one document unit in the LDA estimation. A possible extension to this approach, which will be investigated as part of future work, would be to use smaller textual units, i.e. sentences or paragraphs as document units in the LDA estimation. This would naturally take into account the proximity evidence as well, in addition to the topical distribution of terms.

Acknowledgments This research is supported by the Science Foundation Ireland (Grant 07/CE/I1142) as part of the Centre for Next Generation Localisation (CNGL) project. The authors express their gratitude to the anonymous reviewers whose feedback helped improve the quality of this paper.

References

Blei, D. M., Ng, A. Y., and Jordan, M. I. (2003). Latent Dirichlet allocation. *J. Mach. Learn. Res.*, 3:993–1022.

Chen, D., Xiong, Y., Yan, J., Xue, G.-R., Wang, G., and Chen, Z. (2010). Knowledge transfer for cross domain learning to rank. *Inf. Retr.*, 13(3):236–253.

Chinnakotla, M. K., Raman, K., and Bhattacharyya, P. (2010). Multilingual PRF: English lends a helping hand. In *Proceedings of the SIGIR '10*, pages 659–666, New York, USA. ACM.

Dandapat, S., Morrissey, S., Way, A., and van Genabith, J. (2012). Combining EBMT, SMT, TM and IR technologies for quality and scale. In *Proceedings of ESIRMT and (HyTra)*, pages 48–58. ACL.

Fox, C. (1992). *Lexical analysis and stoplists*, pages 102–130. Prentice-Hall.

Griffiths, T. L. and Steyvers, M. (2004). Finding scientific topics. *PNAS*, 101(suppl. 1):5228–5235.

Hiemstra, D. (2000). *Using Language Models for Information Retrieval*. PhD thesis, Center of Telematics and Information Technology, AE Enschede.

Hull, D. A. and Grefenstette, G. (1996). Querying across languages: A dictionary-based approach to multilingual information retrieval. In *Proceedings of SIGIR*, pages 49–57.

Lavrenko, V., Choquette, M., and Croft, W. B. (2002). Cross-lingual relevance models. In *Proceedings of the SIGIR '02*, pages 175–182, New York, USA. ACM.

Lavrenko, V. and Croft, B. W. (2001). Relevance based language models. In *Proceedings of the SIGIR '01*, pages 120–127. ACM.

Leveling, J., Ganguly, D., and Jones, G. J. F. (2010). DCU@FIRE2010: Term Conflation, Blind Relevance Feedback, and Cross-Language IR with Manual and Automatic Query Translation. In *Working Notes of the FIRE*.

Littman, M., Dumais, S. T., and Landauer, T. K. (1998). Automatic cross-language information retrieval using latent semantic indexing. In *Cross-Language Information Retrieval, chapter 5*, pages 51–62.

Lovins, J. B. (1968). Development of a stemming algorithm. *Mechanical translation and computation*, 11(1-2):22–31.

Majumder, P., Mitra, M., Pal, D., Bandyopadhyay, A., Maiti, S., Pal, S., Modak, D., and Sanyal, S. (2010). The FIRE 2008 evaluation exercise. *ACM Trans. Asian Lang. Inf. Process.*, 9(3).

Majumder, P., Mitra, M., Parui, S. K., Kole, G., Mitra, P., and Datta, K. (2007). YASS: Yet another suffix stripper. *ACM Trans. Inf. Syst.*, 25(4).

Mimno, D. M., Wallach, H. M., Naradowsky, J., Smith, D. A., and McCallum, A. (2009). Polylingual topic models. In *EMNLP*, pages 880–889.

Nie, J.-Y., Simard, M., Isabelle, P., and Durand, R. (1999). Cross-language information retrieval based on parallel texts and automatic mining of parallel texts from the web. In *Proceedings of SIGIR '99*, pages 74–81.

Paik, J. H., Pal, D., and Parui, S. K. (2011). A novel corpus-based stemming algorithm using co-occurrence statistics. In *Proceedings of the SIGIR '11*, pages 863–872.

Papineni, K., Roukos, S., Ward, T., and Zhu, W.-J. (2002). BLEU: a method for automatic evaluation of machine translation. In *Proceedings of the ACL*, pages 311–318, Stroudsburg, PA, USA.

Ponte, J. M. (1998). *A language modeling approach to information retrieval*. PhD thesis, University of Massachusetts.

Robertson, S. E. and Sparck Jones, K. (1988). *Relevance weighting of search terms*, pages 143–160. Taylor Graham Publishing, London, UK.

Salton, G. and Buckley, C. (1990). Improving retrieval performance by relevance feedback. *JASIS*, 41(4):288–297.

Savoy, J. (1999). A stemming procedure and stopword list for general French corpora. *Journal of the American Society for Information Science*, 50(10):944–952.

Udupa, R., Saravanan, K., Bakalov, A., and Bhole, A. (2009). "They Are Out There, If You Know Where to Look": Mining Transliterations of OOV Query Terms for Cross-Language Information Retrieval. In *ECIR*, pages 437–448.

Vulić, I., De Smet, W., and Moens, M.-F. (2011). Cross-language information retrieval with latent topic models trained on a comparable corpus. In *Proceedings of the 7th Asia conference on Information Retrieval Technology*, AIRS'11, pages 37–48, Berlin, Heidelberg. Springer-Verlag.

Wei, X. and Croft, W. B. (2006). LDA-based document models for ad-hoc retrieval. In *SIGIR '06*, pages 178–185, New York, USA. ACM.

Xu, J. and Croft, W. B. (2000). Improving the effectiveness of informational retrieval with Local Context Analysis. *ACM Transactions on information systems*, 18:79–112.

Structured Term Recognition in Medical Text

Michael Glass *Alfio Gliozzo*
IBM T.J. Watson Research, Yorktown Heights
mrglass@us.ibm.com, gliozzo@us.ibm.com

ABSTRACT
In this work we present a novel approach to bootstrap domain specific terminology, namely Structured Term Recognition, and we apply it to the medical domain. In contrast to previous approaches, based on observing distributional properties of terminology with respect to their contexts, our method analyzes the "internal structure" of multi-word terms by learning patterns of word clusters. Evaluation shows that our method can be used to boost the performances of term recognition systems based on dictionary lookup while extending the coverage of large ontologies like UMLS.

KEYWORDS: terminology recognition, bootstrapping, medical terminology, named entity recognition, joint inference.

Proceedings of COLING 2012: Technical Papers, pages 943–958,
COLING 2012, Mumbai, December 2012.

1 Introduction

Recognizing occurrences of domain specific terms and their types is important for many text processing applications. This problem is not easy, particularly in domains like medicine, where very rich terminology is generated by domain experts on a daily basis.

In spite of the large interest in statistical term recognition in Natural Language Processing (NLP), state of the art approaches for term recognition in the medical domain are still based on dictionary lookup with some heuristics for partial mapping (Aronson, 2001). In fact, very large terminological resources, such as the Unified Medical Language System (UMLS) (Bodenreider, 2004), have been developed in the medical domain. The reason is that medical terminology can not be identified by looking at superficial features only, such us capitalization of words, prefixes and suffixes. In fact, diseases names, symptoms and most medical terms are not proper names, so they are not capitalized. In addition, they are usually characterized by a rather complex internal structure and composed by many words. In addition, distributional similarity metrics, i.e. recognition approaches based on the analysis of the local context where the term is located, work well when applied to single words or very frequent words, which is not the case for most of the medical terms we are interested in.

In the context of the research on adapting a question answering system to the medical domain, we encountered the term recognition problem in many places. For example, recognizing names of diseases, symptoms and treatments is necessary to answer most of the Doctor's Dilemma^{TM}questions (American College of Physicians, 2012), an evaluation benchmark we used to measure the ability of our system to answer medical questions. To assess the validity of the answer "HFE hereditary hemochromatosis" with respect to the question "Metabolic condition that can set off airport metal detector", it is important to know that the answer is a type of metabolic condition. One way to address this problem is to use medical lexica where different terms are associated to semantic types, and then check whether the type of the candidate answer matches the type required by the question.

On the other hand, UMLS is far from complete. Many disease names (especially multi-words) are not recognized by dictionary lookup approaches. In many cases, specific terms are missing (e.g. "HFE hereditary hemochromatosis" in the question above). The problem is particularly evident when examining the answers to the Doctor's Dilemma questions. Analyzing 704 doctor's dilemma questions where the system located the correct answer as candidate, in 16.5% of the cases one of the correct answers was not in UMLS. About half of them were quantities (e.g. 20mg) while the remaining half were medical terms that are not in UMLS. On the latter subset of questions, the end to end accuracy of the full QA system dropped by almost 50% if compared to the performance on questions where the answer was an UMLS term. The explanation for this drop in performance is mostly due to type recognition problems: the system is not able to select the right answer on the basis of its type, for example confusing treatments with diseases.

This observation motivates the work presented in this paper. Our method is able to recognize the type of domain specific terms based on an innovative principle, called Structured Term Recognition (STR). We begin with a simple observation: in many technical domains - and in the medical domain in particular, terms have repetitive patterns. For example, terms describing cancer include "bladder cancer", "brain tumor", "skin cancer" and "uterine sarcoma". These terms are all composed of a word indicating a part of the human anatomy followed by a word similar to "cancer". If the sequence of words "pituitary tumor" appears in text, and "pituitary" is known to be an a word indicating a part of the anatomy, it is reasonable to identify this

as a cancer term even if it is not in the dictionary of cancer terms. A simple baseline where any noun phrase with the word "cancer" as it's head is identified as a cancer term will lead to such misidentifications as "some types of cancer" and "a different cancer". In fact, syntactic information alone is not enough to recognize technical terms, additional semantic information reflecting the type of the term constituents should be considered.

Structured Term Recognition (STR) is a way to address this problem. It works by examining the semantic structure of terms, acquiring semantic patterns describing it, and identifying new terms with similar structure. In this paper we focus on multi-word terms, in the case of single word terms our method becomes indistinguishable from prior work on extending a thesaurus by distributional similarity.

Similar to other existing bootstrapping methods, STR begins with an existing dictionary and uses patterns to extend its coverage. In contrast, it does not use contextual information or shallow features describing the term itself, instead it uses "internal deep semantic" features describing what a domain term should look like, i.e. its semantic pattern. Combinations of STR and techniques based on contextual information are obviously possible. However, in this paper we decided to focus on the contribution of the STR technique in isolation because we want to show that in the medical domain this information is a strong indicator of the entity type if well represented and handled. We present three different approaches to solve the problem, two based on recognizing terms on the basis of the sequences of the word clusters and a third one based on Probabilistic Context Free Grammars (PCFG) and we evaluate them on the task of predicting the semantic group of a held out subset of terms extracted from UMLS.

The paper is structured as follows. Section 2 describe STR into details, while section 3 is about evaluating our method. In section 4 a literature review is done, while section 5 concludes the paper and highlight directions for future work.

2 Structured Term Recognition

The inputs of STR are a set of terms $t \in T$ from a dictionary, each belonging to a set of types $\Gamma(t) = \{\gamma_1, \gamma_2, \ldots\}$, and a domain specific corpus U. Terms are composed of lists of words $t = w_1, w_2, \ldots, w_n$ as identified by a tokenizer. The union of all tokens for all of the terms is the word set W.

The output of STR is a set of pattern-to-type mappings $p \mapsto \gamma$ that can be used to recognize terms and their type in new texts. These terms are not necessarily present in U.

STR combines two key components:

1. Clustering words in W by their distributional and type similarity.
2. Constructing patterns describing terms in the dictionary.
   (a) Sequences of Clusters are patterns.
   (b) Patterns are the left hand side of PCFG rules.

2.1 Word Clustering

The goal of this component is to cluster the words in W into clusters $C_1, C_2, \ldots, C_n$. Clusters are partly determined by distributional similarity, meaning that words in the same cluster

have similar meanings because they are substitutable in context. This follows a large body of work in unsupervised clustering of words. Early work used distributional similarity to perform part-of-speech induction (Schütze, 1993). Later work has developed automatic methods of thesaurus construction using similar techniques (Lin, 1998).

Using a large, domain specific, unannotated corpus U, STR gathers vectors describing the context in which each word is found. Since parsing is generally lower quality in non-newswire domains, we do not use dependency based contexts. Instead the corpus is simply tokenized. Contexts are built from a small window of surrounding words: two words to the left and two words to the right. The information about the direction and distance from the word is retained. For example, in the sentence "Early detection of **cancer** increases the odds of successful treatment." the contexts for the word "cancer" are given in table 1.

cancer	
Direction	Context Word
Left-2	detection
Left-1	of
Right-1	increases
Right-2	the

Table 1: Contexts for "cancer"

For each word we build a vector of contexts. Note here that in order to build the context vector we do not restrict context words to words in W, instead we consider any possible token in the large corpus. Each dimension of the vector is an individual context f and its value is the number of times it occurred with the word. This vector of raw counts must be re-weighted, otherwise very high frequency but uninformative contexts, such as "of" and "the", will dominate. We use Pointwise Mutual Information (PMI) as a re-weighting method. PMI measures the strength of association between two events such as the occurrence of a word and the occurrence of a context.

$$\text{pmi}(w; f) = \log \frac{p(w, f)}{p(w)p(f)} \tag{1}$$

Equation 1 gives the formula for PMI, where $p(w)$ is the frequency of the word in the corpus divided by the number of words in the corpus and $p(f)$ is the frequency of the context divided by the total number of contexts.

Once the context vectors for each word are gathered and re-weighted they can be compared for similarity using the cosine similarity metric.

The domain specific dictionary may have many of the words in W mapped to types. However, in general a word need not be mapped to one type. It may have multiple senses, each with a distinct type. For each word we construct a type vector, where the dimensions are types and the value is either a one or zero depending on whether the word is given as that type in the dictionary. Since the dictionary groups words with similar meanings in the same type, words with overlapping type vectors will have some meanings that are similar.

2.2 Pattern Acquisition

The goal of this step is to learn a set of rules $R = \{p_1 \mapsto \gamma_1, p_2 \mapsto \gamma_2, \ldots, p_k \mapsto \gamma_k\}$ which are able to recognize most terms in T and at the same time can be used to identify new terms in the large corpus. This is done by generating a set of candidate rules and scoring them according to how they perform on the training set. We trained two different models for clusters and patterns.

In the sequence of clusters model, each word is assigned to a single cluster, patterns are sequences of clusters, and rules map a pattern to a type. Each possible sequence of clusters maps to at most one type.

In the probabilistic context free grammar (PCFG) model, words are assigned to multiple clusters, each assignment has some weight corresponding to a probability. The rules are a set of binary PCFG rules where two non-terminals are mapped to a single non-terminal. All non-terminals on the right hand side of such rules are types. Each possible sequence of clusters has a Viterbi (most likely) parse ending in a single type.

2.3 Sequence of Clusters

For the Sequence of Clusters model, the clustering is a mapping function $\Theta : W \rightarrow C$ assigning each word in W to its corresponding cluster. The patterns are the mappings of each term $t \in T$ into its corresponding pattern $P(t) = p = \Theta(w_1), \Theta(w_2), \ldots, \Theta(w_j)$, where t is the sequence of words $t = w_1, w_2, \ldots, w_j$. A term is typed if there is a rule $P(t) \mapsto \gamma \in R$. This typing is correct ($t \in Correct$) if t has γ as a type in the dictionary, $\gamma \in \Gamma(t)$, otherwise it is incorrect ($t \in Incorrect$).

An example sequence of clusters pattern and some of its recognized terms are given in Table 2. The clusters are named according to their most frequent members. So "{sensation, fear, feeling}" is the cluster that includes those words, as well as other distributionally similar words.

Pattern		
{spastic, flaccid, tonic}	{sensation, fear, feeling}	{tenderness, headache, tremor}
Recognized Terms		
horizontal	gaze	paresis
tonic	reflex	spasm
dural	puncture	headache
exertional	heat	exhaustion
charley	horse	cramps

Table 2: Example pattern and recognized terms

The training seeks to create rules and word-to-cluster mappings to maximize intra-cluster similarity and the performance of the rules on the training data. The intra-cluster similarity is measured by the cosine of each word to its cluster's centroid. The performance of the rules is given by $|Correct| - |Incorrect| + \theta_{REG} \cdot |R|$. The θ_{REG} is a regularization parameter to penalize the creation of too many rules.

2.3.1 Non-Joint Optimization

A straightforward way of doing this is to first optimize intra-cluster similarity, which can easily be converted to the objective function in spherical KMeans. We use GMeans (Dhillon et al.,

2001), a variant of KMeans that uses first variation to escape from some local maxima, to cluster. Then, given fixed word-to-cluster assignments, the optimal set of rules may be found in linear time. By mapping each term to its pattern and constructing a hash map from patterns to sets of terms we can determine for any possible θ_{REG} if mapping the pattern to the most frequent type γ_{ts} in its term set will improve rule performance. This method is fast, but it does not perform nearly as well as a joint optimization.

2.3.2 Joint Optimization

Our method of joint optimization is to cast the problem as a MAP (Maximum a posteriori) inference task in probabilistic logic and use a general purpose probabilistic inference engine to do the optimization. Different sets of word-to-cluster mappings and rules are different possible worlds. The probability of a world is given by the objective function and the inference attempts to find the most likely world.

The objective function is precisely described by the formulas in Figure 1, which closely matches the description given to the probabilistic inference engine. In this description, all variables are implicitly universally quantified. The quantity preceding the "#" in the formulas that have them is a weight, every true grounding of the formula on the right adds the quantity on the left to the weight of the possible world. Like Markov Logic (Richardson and Domingos, 2006), the semantics of the weight are that it is the log of an unnormalized probability. Note that unlike Markov Logic, we are not limited to first order logic.

$$\theta_{DC} \cdot \textbf{cosine}(v, cv) \;\#\; inCluster(w, c) \wedge distVect(w, v) \wedge distCentroid(c, cv)$$
$$cluster(c) \Rightarrow distCentroid(c, \textbf{sum}(\{v : inCluster(w, c) \wedge distVect(w, v)\}))$$
$$\theta_{TC} \cdot \textbf{cosine}(v, cv) \;\#\; inCluster(w, c) \wedge typeVect(w, v) \wedge typeCentroid(c, cv)$$
$$cluster(c) \Rightarrow typeCentroid(c, \textbf{sum}(\{tv : inCluster(w, c) \wedge typeVect(w, tv)\}))$$
$$wordAt(t, i, w) \wedge inCluster(w, c) \Rightarrow typeAt(t, i, c)$$
$$term(t) \Rightarrow patternOf(t, \textbf{makePattern}(\{(i, c) : typeAt(t, i, c)\}))$$
$$\theta_A \cdot \textbf{scoreType}(t, \gamma) \;\#\; isRule(p, \gamma) \wedge patternFor(p, t)$$
$$\theta_{REG} \;\#\; isRule(p, \gamma)$$

Figure 1: Logical description of the Sequence of Clusters model

There are three basic parts: the first four lines give the objective function for spherical KMeans, lines 1 and 2 using distributional similarity and lines 3 and 4 using type vector similarity. The next three lines evaluate the performance of the patterns on the training data. The final line is the regularization, penalizing the creation of too many patterns.

The predicates $distVect, typeVect, cluster, term, wordAt$ are given as evidence to the inference. Each word is related to its normalized distribution vector and type vector by $distVect$ and $typeVect$ respectively. A term, position and word at that position are related by $wordAt$. The function **makePattern** takes a set of index/cluster pairs and produces a sequence of clusters as a string so that for every term t, $patternOf(t, P(t))$. The distribution and type centroids are found by summing the corresponding vectors from the words in the clusters. Since cosine is our similarity metric, it is not necessary to renormalize these vectors. The function **scoreType**

implements the logic of returning a 1 for a correct type mapping, -1 for an incorrect type mapping and 0 for a $\emptyset$ type mapping.

The θ parameters control the balance of the different components of the optimization. We selected values of $\theta_{DC} = 1, \theta_{TC} = 1, \theta_A = 1, \theta_{REG} = -3$.

The predicates $inCluster, isRule$ are the proposal predicates. The basic inference algorithm is to make proposals which are then either accepted or rejected. We have three proposals: move a word to a different cluster, create a rule, and remove a rule. A proposal is accepted if and only if it improves the objective function. The proposals are described formally in Table 3. The variables in the proposal formula are bound to a random true grounding of the formula. Then the remove list is set false and the add list is set true.

Proposal Formula	Remove	Add
$inCluster(w, c_1) \wedge cluster(c_2) \wedge c_1 \neq c_2$	$inCluster(w, c_1)$	$inCluster(w, c_2)$
$patternFor(p, t) \wedge \gamma \in \Gamma(t) \wedge \neg isRule(p, \gamma)$	$\emptyset$	$isRule(p, \gamma)$
$isRule(p, \gamma)$	$isRule(p, \gamma)$	$\emptyset$

Table 3: Proposals for the Sequence of Clusters Model

A key optimization is to not recompute the centroid of each cluster on every proposal. Instead, it is recomputed with a probability that decreases as the size of the cluster increases. The intuition behind this idea is that the centroid does not change much if the size of the cluster is large.

A final step in sequence of clusters is to find the optimal pattern set given the word-to-cluster assignments, as in the non-joint approach. This step can also accommodate different precision-recall trade-offs by varying the θ_{REG} parameter.

In the sequence of clusters model we restrict our attention to terms of length two and three so that each sequence of clusters will have many terms that match it.

2.4 Probabilistic Context Free Grammar

In order to address longer terms, and to improve performance we developed a Probabilistic Context Free Grammar (PCFG) for terms. This grammar is binary with non-terminals consisting of types and clusters.

For all words in the dictionary as terms, we fix their cluster assignments to their types, initially with equal weight. Formally, for all single-word terms w and all types $\gamma \in \Gamma(w)$, $inCluster(w, \gamma, 1)$. We used the results of the previous model to assign clusters to the remaining words. This is a soft clustering.

An example PCFG parse is given in Figure 2. The non-terminal labeling a terminal (word) may be either a cluster or a type, but the non-terminal labeling a pair of non-terminals is always a type.

The objective function is to maximize the correctness of the Viterbi parses on the training data. The objective function is precisely described in Figure 3.

As before the predicates $wordAt, term$ are given as evidence. Now $term$ relates a term to its length. The function **maxType** takes a set of type/weight pairs and returns the type with the largest weight or $\emptyset$ if the set is empty. The key predicate is $chart$ which holds the chart parse as a set of ground predicates where $chart(t, i, j, \gamma, x)$ indicates that for term t the type γ spans

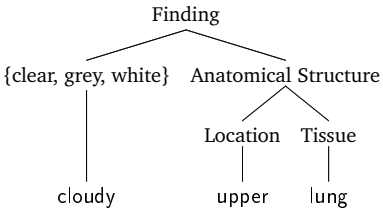


Figure 2: Example PCFG parse

$$wordAt(t,i,w) \wedge inCluster(w,\gamma,x) \Rightarrow chart(t,i,(i+1),\gamma,x)$$
$$isRule(\gamma_1,\gamma_2,\gamma_p,x_p) \wedge chart(t,i,j,\gamma_1,x_1) \wedge chart(t,j,k,\gamma_2,x_2) \Rightarrow$$
$$typeAt(t,i,k,\gamma_p,(x_1+x_2+x_p))$$
$$typeAt(t,i,j,\gamma,z) \Rightarrow chart(t,i,j,\gamma,\max(\{x : typeAt(t,i,j,\gamma,x)\}))$$
$$term(t,l) \Rightarrow vitParse(t,\textbf{maxType}(\{(\gamma,x) : chart(t,0,l,\gamma,x)\}))$$
$$\theta_A \cdot \textbf{scoreType}(t,\gamma) \ \#\ vitParse(t,\gamma)$$
$$\theta_{REG} \ \#\ isRule(\gamma_1,\gamma_2,\gamma_p,x_p)$$

Figure 3: Logical description of the PCFG model

i to j with log likelihood x. The proposal predicates are $isRule, inCluster$. The weights of rules and proposals are the log of their probability. The proposals are: change the weight of a parse rule or cluster assignment, add a parse rule, and remove a parse rule. Table 4 gives these proposals formally. The function $random(x,y)$ produces a random number between x and y.

Proposal Formula	Remove	Add
$inCluster(c,w,x)$	$inCluster(c,w,x)$	$inCluster(c,w,random(-2,2))$
$isRule(\gamma_1,\gamma_2,\gamma_p,x)$	$isRule(\gamma_1,\gamma_2,\gamma_p,x)$	$isRule(\gamma_1,\gamma_2,\gamma_p,random(-2,2))$
$\gamma_1 \in C \cup \Gamma \wedge \gamma_2 \in C \cup \Gamma \wedge \gamma_p \in \Gamma$	$\emptyset$	$isRule(\gamma_1,\gamma_2,\gamma_p,random(-2,2))$
$isRule(\gamma_1,\gamma_2,\gamma_p,x)$	$isRule(\gamma_1,\gamma_2,\gamma_p,x)$	$\emptyset$

Table 4: Proposals for the PCFG Model

Another key optimization is the use of declarative variable coordination inspired by the imperative variable coordination of Factorie (McCallum et al., 2009). The formulas with a "⇒" may be read as infinite weight implications, but unlike other approaches to combining hard and soft formulas we never consider states where the hard formulas are violated. Instead, for every true grounding of the left hand side, we immediately create a grounding of the right hand side. If all the left hand sides supporting a coordinated predicate become false, the corresponding coordinated predicate also becomes false. It may be more clear to consider the predicate on the right hand side as being defined by the disjunction of all its left hand sides.

Because the speed of the training is closely related to the length of the terms we only train on terms up to length four. However, we test the model on all terms.

3 Evaluation

The patterns found by the STR algorithm produce a term recognition function, able to decide the type of a term $t = w_1, w_w, \ldots, w_j$ by simply checking for a rule $P(t) \mapsto \gamma$ in the case of sequence of clusters, or by parsing the term with a chart parser in the PCFG model. We evaluate the quality of this recognition function by testing it on a held out set of terms in the dictionary. As with training, we consider a mapping correct if it maps the term to one of the types given in the dictionary.

3.1 Experimental Setup

For our input dictionary we use the Unified Medical Language System (UMLS). UMLS is an ontology of mostly medical terms called atoms. There is a type hierarchy with 133 semantic types these are collected into 15 semantic groups. The semantic groups are a flat and clear division of semantic types. Each UMLS atom may have multiple semantic types and possibly multiple semantic groups. In our test set 7.2% percent of the atoms had multiple semantic groups.

First we selected the multi-word UMLS atoms that occur in our corpus at least 20 times. Our corpus is 3.3 GB of text from Medline abstracts and medical textbooks. This first step is necessary because many UMLS atoms are not terms that occur in text and therefore there is no benefit to recognizing them. This set of UMLS atoms is then divided into a training and test set with 90 percent in training and the remaining 10 percent in test. We trained STR to recognize the semantic group of the terms in the training set, then evaluated the resulting rules on the test set. This produced a training size of around 72,000 terms.

3.2 Results

For the sequence of clusters model we obtain a precision recall curve by varying the pattern regularization parameter θ_{REG}. For the PCFG model we vary the parse score threshold to obtain a precision-recall trade-off.

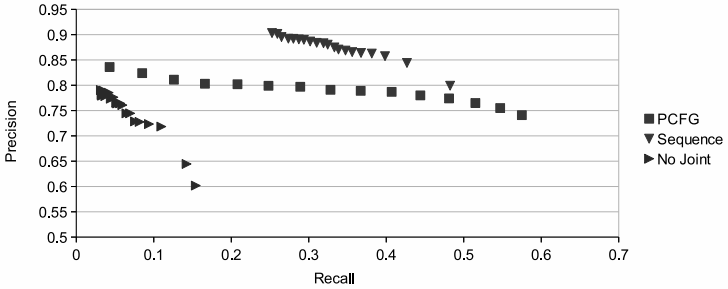


Figure 4: Precision Recall Curves

With a $\theta_{REG} = -20$ we obtain a precision of 90.3% and a recall of 25.3%. Setting the regularization parameter to zero maximizes recall at 48.2% with precision decreasing to 79.9%. With the parse score threshold of 3.5, we obtain a precision of 83.6%, but a very low recall,

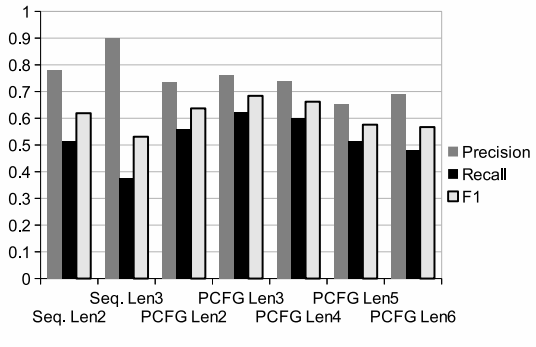


Figure 5: Performance per Term Length

4.3%. Dropping the threshold entirely gives a precision of 74.1% and a recall of 57.5%. Both of these models substantially outperform the non-joint method described in Section 2.3.1. A baseline model of selecting the most frequent semantic group would perform at precision = recall = F1 score = 23.8%. With a test set of approximately 8,000 all of these differences are highly significant.

Note that while the sequence of clusters model is evaluated only on length two and three terms, the PCFG model is evaluated on all terms. Figure 5 compares the performance of each model on different term lengths.

The performance of all models improves as the amount of optimization time increases. The learning curve for the sequence of clusters model is given in Figure 6. Note that the x-axis is the number of proposals, not the number of training instances. Performance grows from an F1 score of 49.4% with 500 thousand proposals to an F1 score of 60.2% after 27 million proposals.

The performance per group is given in Figure 7. The groups are listed in order of their frequency in the dictionary. There is a clear effect of term frequency on performance with all but one of the top performing half of groups in the top half of term frequency.

The performance of the PCFG model is broken down by group in Figure 8. The recall and F1 scores on “Genes and Molecular Sequences” and “Devices” show large gains relative to the sequence of clusters model. There is again a clear effect of term frequency.

Performance is generally lowest on the semantic groups that are not specifically biomedical: Objects, Devices, Phenomena, Activities & Behaviors, Organizations, Occupations and Geographic Areas. Other than a lower frequency in UMLS, this is likely due to a lower amount of regular structure in these less technical terms.

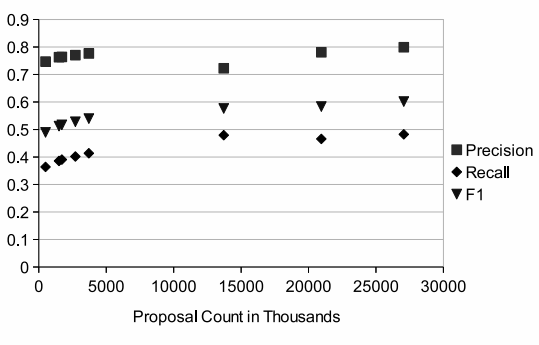


Figure 6: Learning Curve for Sequence of Clusters Model

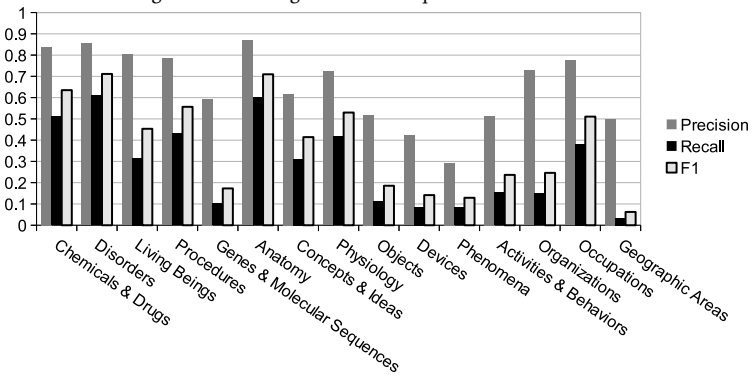


Figure 7: Performance per Group for Sequence of Clusters

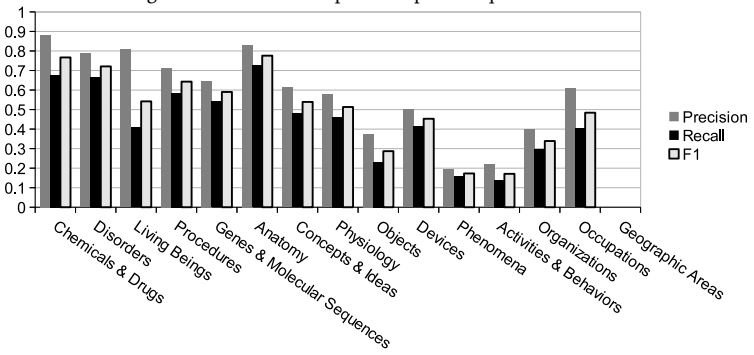


Figure 8: Performance per Group for PCFG

4 Related Work

The goal of our work is to extend UMLS with additional terms mapped to types. Other work has pursued similar goals. This section provides a state of the art about.

4.1 Automatic Term Recognition

There is a large body of work on Automatic Term Recognition ATR and term clustering. In STR we identify that a sequence of words is a term and that it has a particular type in a single process. ATR operates by first gathering strings that are terms and then clustering them, or extending existing clusters where each cluster of terms are terms of a common type.

After terms are identified, some measure of term similarity is applied to cluster the terms or extend existing clusters of terms. Some methods of term similarity are contextual, lexical and pattern-based. Contextual similarity uses information about what words appear before and after the term (Maynard and Ananiadou, 2000), or what words are nearby in a dependency parse (Grefenstette, 1994). The frequency of each of these contexts is a dimension in a vector describing the distribution of the term. These vectors can then be compared for similarity using some vector similarity function. Lexical similarity examines what words are in common between two terms. For example, two terms with the same head word are likely to have the same type. Pattern-based similarity (Nenadić et al., 2002) uses lexico-syntactic patterns like the such-as pattern "X such as Y and Z". These patterns can be used as evidence that the terms appearing as Y and Z have the same type.

ATR deals primarily with frequent words. The term must be frequent enough to be identified as a term and in the case of contextual similarity must be frequent enough to build a meaningful distribution vector. Only the lexical similarity metric examines the words inside the term and this metric only considers lexical matches between the words of the term, not the types of the words or the structure of the words in the term. The main advantage of this type of work is that it can proceed without an existing term dictionary.

4.2 Supervised Named Entity Recognition

Our methods are also similar to Named Entity Recognition (NER). NER is the task of identifying mentions of rigid designators, especially people, places and organizations (Coates-Stephens, 1992). Recently, biological types such as protein, DNA and cell line have gained attention (Settles, 2004). Unlike automatic term recognition, where the goal is to build a dictionary of terms, NER typically proceeds from a corpus annotated with mentions and their types and learns a model for detecting future mentions.

One common model is the linear chain Conditional Random Field (CRF) (McCallum and Li, 2003). This is the discriminatively trained variant of the Hidden Markov Model (HMM) (Bikel et al., 1997). In the simplest version of such models, each word belonging to a mention for some type is tagged with that type's identifier. One weakness of this simple version is that in the case where two mentions of a certain type are contiguous, it is not possible to tell where one begins and the other ends - or even that there are two mentions rather than one. To alleviate this problem, and enrich the model, more complex tag sets are used. A model might have a tag for the beginning, inside and end of each type of entity. This enables the model to learn words that are more likely to begin and end mentions, effectively learning a little of the structure of the entities.

NER systems have also addressed the lexical composition of mentions for each type. For example, organization names often include the name of a person or a place (Wolinski et al., 1995). Even common nouns such as "associates" can indicate that a mention is of type organization (Wolinski et al., 1995).

In the biomedical domain, supervised CRFs have shown success at identifying genes and proteins. Using an annotated corpus from Medline abstracts, state of the art systems reach F-Scores of 78.4% for protien names (Tsai et al., 2006). Unfortunately, over a broader set of medical term types, basic linear chain CRF models cannot generalize beyond the terms in the training data (Zhang et al., 2010), with only a handful of new, correct medical terms identified from thousands of candidate new terms.

Unlike most NER systems our focus is on terms composed primarily of common nouns. While NER systems are trained on a labeled corpus, STR instead uses a term dictionary and an unlabeled corpus.

4.3 Bootstrapping Named Entity Recognition

Other work has identified and typed named entities based on an existing list of entities for each type. Usually, NER bootstrapping focuses on building a gazetteer of relatively common entities given a very small initial set of entities for each type. One approach is to use a small set (4) of example entities and search the web for documents that contain all of the example entities (Nadeau et al., 2006). The resulting documents are likely to contain lists or tables of the entities for the relevant type. By using an HTML context classifier trained with the context of the search entities as positive examples, other entities in the same HTML contexts may be extracted (for example "<td> x </td>"). The extracted entities are added to the list of known entities enabling additional search queries to be generated.

Mutual bootstrapping (Riloff and Jones, 1999) simultaneously learns entities of the selected type and patterns for identifying the entities. From a small set of seed entities, lexico-syntactic patterns are learned that suggest an entity of the appropriate type. For example, the noun phrase following "headquarted in" is likely to be a location. Extraction patterns are scored by their frequency and (estimated) reliability and entities are scored by the weighted sum of patterns that identify them. This scoring helps to alleviate the core problem of bootstrapping: semantic drift and also allows for a precision/recall trade-off.

A related bootstrapping approach (Kozareva, 2006) trains an NER classifier based on the current gazetteer then runs the classifier over text and adds the new terms recognized by the classifier to the gazetteer. This method requires a recognition system that goes beyond dictionary matching and uses context to a significant degree. The features used by the classifier include capitalization, trigger words specific to locations, organizations and people, and whether words in the noun phrase belong to a gazetteer for the type of interest.

Unlike traditional NER bootstrapping, STR assumes an existing large dictionary with hundreds if not thousands of examples per type. STR can identify the long tail of entities that may never occur in an easily interpretable list, table or lexico-syntactic pattern.

4.4 Joint Clustering

Other work has pursued different goals using related methods. The segmentation and de-duplication of citations has received considerable attention as a joint inference task. The

decisions of linking author, title, venue and citation records are all interdependent, since linking two citations implies that the fields composing the citations should also be linked. The task of segmenting the citations and identifying their fields is also influenced by the record linking. An easily segmented citation linked to a difficult case may make the difficult case much easier.

Using Markov Logic a joint entity linking model showed improvement in citation, author and venue linking over an independent model based on the field similarities alone (Singla and Domingos, 2006). Imperatively-Defined Factor Graphs (IDFs) produced still higher performance by incorporating features that could not be tractably expressed in factor graph systems that fully ground the network before inference (Singh et al., 2009).

5 Conclusion and Future Work

In this paper we presented a novel approach to semi-supervised term recognition. Unlike previous approaches we induced structure of the known terms to predict new terms. This enables type recognition for terms that appear only once in the corpus. We compared three different methods. All of them substancially improved over a most frequent baseline. We proposed two methods based on recognizing sequences of clusters. The one based on joint optimization significantly increased both precision and recall over the same method based on static clustering, reaching 90% precision at almost 30% recall. PCFG grammars allow us to achieve better recall (60%) with reasonably high precision (75%). In addition, PCFG allows us to recognize terms of any length.

The results shown by this paper are only partial as they do not take into account the role of context in disambiguating the types of terms. This limitation is intentional because: (i) we are interested in recognizing the type of answers independently of their context in cases when they are generated from databases or other non-textual material (ii) we are interested in exploring to what extent the internal structure of terminology can be used to express the semantics of terms.

In the future we will integrate STR with existing approaches of contextual, lexical and pattern-based term recognition. In addition, we will apply our technique to more fine grained type systems and other domains.

The type of structure learned is currently very simple, either a sequence of word clusters or a binary grammar. Other types of regularities exist, such as reordering. The pair of terms “leg pain” and “pain in leg” as well as many other similar pairs suggest a general alternation rule. We are going to explore this direction in order to learn transformational rules that can lead to identification of synonyms, hypernyms and to a better understanding on the underlying linguistic phenomena characterizing the generation and recognition of domain specific terminology.

Acknowledgments

The authors gratefully acknowledge the support of Defense Advanced Research Projects Agency (DARPA) Machine Reading Program under Air Force Research Laboratory (AFRL) prime contract no. FA8750-09-C-0172. Any opinions, findings, and conclusion or recommendations expressed in this material are those of the authors and do not necessarily reflect the view of the DARPA, AFRL, or the US government. We would like to express our gratitude to the Watson Team and in particular to Chris Welty for inspiring leadership and David Gondek for helping us figuring out goals and technical solutions.

References

American College of Physicians (2012). Doctor's Dilemma^{TM}competition. `www.acponline.org/residents_fellows/competitions/doctors_dilemma`.

Aronson, A. R. (2001). Effective mapping of biomedical text to the umls metathesaurus: The metamap program. In *Proceedings of AMIA Symposium*, pages 17–21.

Bikel, D. M., Miller, S., Schwartz, R., and Weischedel, R. (1997). Nymble: a high-performance learning name-finder. In *Proceedings of the fifth conference on Applied natural language processing*, ANLC '97, pages 194–201, Stroudsburg, PA, USA. Association for Computational Linguistics.

Bodenreider, O. (2004). The Unified Medical Language System (UMLS): integrating biomedical terminology. *Nucleic acids research*, 32(Database issue):D267–D270.

Coates-Stephens, S. (1992). The analysis and acquisition of proper names for the understanding of free text. *Computers and the Humanities*, 26(5/6):pp. 441–456.

Dhillon, I. S., Fan, J., and Guan, Y. (2001). Efficient clustering of very large document collections. In R. Grossman, C. Kamath, V. K. and Namburu, R., editors, *Data Mining for Scientific and Engineering Applications*, pages 357–381. Kluwer Academic Publishers. Invited book chapter.

Grefenstette, G. (1994). *Explorations in Automatic Thesaurus Discovery*. Kluwer Academic Publishers, Norwell, MA, USA.

Kozareva, Z. (2006). Bootstrapping named entity recognition with automatically generated gazetteer lists. In *Proceedings of the Eleventh Conference of the European Chapter of the Association for Computational Linguistics: Student Research Workshop*, EACL '06, pages 15–21, Stroudsburg, PA, USA. Association for Computational Linguistics.

Lin, D. (1998). Automatic retrieval and clustering of similar words. In *Proceedings of the 17th international conference on Computational linguistics - Volume 2*, COLING '98, pages 768–774, Stroudsburg, PA, USA. Association for Computational Linguistics.

Maynard, D. and Ananiadou, S. (2000). Identifying terms by their family and friends. In *Proceedings of COLING 2000*, pages 530–536.

McCallum, A. and Li, W. (2003). Early results for named entity recognition with conditional random fields, feature induction and web-enhanced lexicons. In *Proceedings of the seventh conference on Natural language learning at HLT-NAACL 2003 - Volume 4*, CONLL '03, pages 188–191, Stroudsburg, PA, USA. Association for Computational Linguistics.

McCallum, A., Schultz, K., and Singh, S. (2009). Factorie: Probabilistic programming via imperatively defined factor graphs. In *Neural Information Processing Systems Conference (NIPS)*.

Nadeau, D., Turney, P. D., and Matwin, S. (2006). Unsupervised named-entity recognition: Generating gazetteers and resolving ambiguity. In Lamontagne, L. and Marchand, M., editors, *Canadian Conference on AI*, volume 4013 of *Lecture Notes in Computer Science*, pages 266–277. Springer.

Nenadić, G., Spasić, I., and Ananiadou, S. (2002). Automatic discovery of term similarities using pattern mining. In *COLING-02 on COMPUTERM 2002: second international workshop on computational terminology - Volume 14*, COMPUTERM '02, pages 1–7, Stroudsburg, PA, USA. Association for Computational Linguistics.

Richardson, M. and Domingos, P. (2006). Markov Logic networks. *Machine Learning*, 62:107–136.

Riloff, E. and Jones, R. (1999). Learning dictionaries for information extraction by multi-level bootstrapping. In *Proceedings of the sixteenth national conference on Artificial intelligence and the eleventh Innovative applications of artificial intelligence conference innovative applications of artificial intelligence*, AAAI '99/IAAI '99, pages 474–479, Menlo Park, CA, USA. American Association for Artificial Intelligence.

Schütze, H. (1993). Part-of-speech induction from scratch. In *Proceedings of the 31st annual meeting on Association for Computational Linguistics*, ACL '93, pages 251–258, Stroudsburg, PA, USA. Association for Computational Linguistics.

Settles, B. (2004). Biomedical named entity recognition using conditional random fields and rich feature sets. In *Proceedings of the International Joint Workshop on Natural Language Processing in Biomedicine and its Applications*, JNLPBA '04, pages 104–107, Stroudsburg, PA, USA. Association for Computational Linguistics.

Singh, S., Schultz, K., and Mccallum, A. (2009). Bi-directional joint inference for entity resolution and segmentation using imperatively-defined factor graphs. In *Proceedings of the European Conference on Machine Learning and Knowledge Discovery in Databases: Part II*, ECML PKDD '09, pages 414–429, Berlin, Heidelberg. Springer-Verlag.

Singla, P. and Domingos, P. (2006). Entity resolution with markov logic. In *Data Mining, 2006. ICDM '06. Sixth International Conference on*, pages 572 –582.

Tsai, T.-h., Chou, W.-C., Wu, S.-H., Sung, T.-Y., Hsiang, J., and Hsu, W.-L. (2006). Integrating linguistic knowledge into a conditional random fieldframework to identify biomedical named entities. *Expert Syst. Appl.*, 30(1):117–128.

Wolinski, F., Vichot, F., and Dillet, B. (1995). Automatic processing of proper names in texts. In *Proceedings of the seventh conference on European chapter of the Association for Computational Linguistics*, EACL '95, pages 23–30, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Zhang, X., Song, Y., and Fang, A. C. (2010). How well conditional random fields can be used in novel term recognition. In *Proceedings of the 24th Pacific Asia Conference on Language, Information and Computation*, pages 583–592, Tohoku University, Sendai, Japan. Institute of Digital Enhancement of Cognitive Processing, Waseda University.

A Dynamic Oracle for Arc-Eager Dependency Parsing

Yoav Goldberg[1] *Joakim Nivre*[1,2]
(1) Google Inc.
(2) Uppsala University
yogo@google.com, joakim.nivre@lingfil.uu.se

ABSTRACT
The standard training regime for transition-based dependency parsers makes use of an oracle, which predicts an optimal transition sequence for a sentence and its gold tree. We present an improved oracle for the arc-eager transition system, which provides a set of optimal transitions for every valid parser configuration, including configurations from which the gold tree is not reachable. In such cases, the oracle provides transitions that will lead to the best reachable tree from the given configuration. The oracle is efficient to implement and provably correct. We use the oracle to train a deterministic left-to-right dependency parser that is less sensitive to error propagation, using an online training procedure that also explores parser configurations resulting from non-optimal sequences of transitions. This new parser outperforms greedy parsers trained using conventional oracles on a range of data sets, with an average improvement of over 1.2 LAS points and up to almost 3 LAS points on some data sets.

KEYWORDS: dependency parsing, transition system, oracle.

Proceedings of COLING 2012: Technical Papers, pages 959–976,
COLING 2012, Mumbai, December 2012.

1 Introduction

The basic idea in transition-based dependency parsing is to define a nondeterministic transition system for mapping sentences to dependency trees and to perform parsing as search for the optimal transition sequence for a given sentence (Nivre, 2008). A key component in training transition-based parsers is an *oracle*, which is used to derive optimal transition sequences from gold parse trees. These sequences can then be used as training data for a classifier that approximates the oracle at parsing time in deterministic classifier-based parsing (Yamada and Matsumoto, 2003; Nivre et al., 2004), or it can be used to determine when to perform updates in online training of a beam search parser (Zhang and Clark, 2008).

Currently, such oracles work by translating a given tree to a static sequence of parser transitions which, if run in sequence, will produce the gold tree. Most transition systems, including the arc-eager and arc-standard systems described in Nivre (2003, 2004), exhibit *spurious ambiguity* and map several sequences to the same gold tree. In such cases, the oracles implicitly define a canonical derivation order. We call such oracles *static*, because they produce a single static sequence of transitions that is supposed to be followed in its entirety. Static oracles are usually specified as rules over individual parser configurations – if the configuration has properties X and the gold tree is Y, then the correct transition is Z – giving the impression that they define a function from configurations to transitions. However, these rules are only correct for configurations that are part of the canonical transition sequence defined by the oracle. Thus, static parsing oracles are only correct as functions from gold-trees to transition sequences, and not as functions from configurations to transitions.

There are at least two limitations to training parsers using static oracles. First, because of spurious ambiguity, it is not clear that the canonical transition sequence proposed by the oracle is indeed the easiest to learn. It could well be the case that a different sequence which also leads to a gold tree is preferable. Second, it happens often in greedy deterministic parsing that the parser deviates from the gold sequence, reaching configurations from which the correct tree is not derivable. The static oracle does not provide any means of dealing with such deviations. The parser's classifier is then faced with configurations which were not observed in training, and this often leads to a sequence of errors. It would be preferable for the parser to explore non-gold configurations at training time, thus mitigating the effect of error-propagation.

In this paper, we introduce the concept of a *dynamic* parsing oracle. Rather than defining a single static canonical transition sequence for producing a given gold tree, the dynamic oracle answers queries of the form: Is transition Z valid in configuration X for producing the best possible tree Y? A key difference compared to a static oracle is that the dynamic oracle no longer forces a unique transition sequence in situations where multiple sequences derive the gold tree. In this case, the dynamic oracle permits all valid transition sequences by answering "yes" on more than one transition Z in a given configuration.[1] The second crucial difference to a static oracle is that the dynamic oracle defined in this work is well-defined and correct for *all* configurations, including configurations which are not a part of a gold derivation. For configurations which are not part of a gold derivation (and from which the gold tree is not reachable), the dynamic oracle permits all transitions that can lead to a tree with minimum loss compared to the gold tree. In this paper, we provide a provably correct dynamic oracle for the arc-eager transition system of Nivre (2003, 2008).

One important use for a dynamic oracle is in training a parser that (a) is not restricted to a

[1] This is similar to the parsing oracle used in the EasyFirst parser of Goldberg and Elhadad (2010).

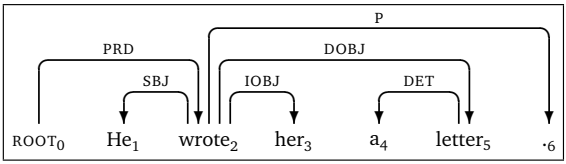


Figure 1: Projective dependency tree

particular canonical order of transitions and (b) can handle configurations that are not part of any gold sequence, thus mitigating the effect of error propagation. To this end, we provide an online training procedure based on the dynamic oracle that deals with spurious ambiguity by treating all sequences leading to a gold tree as correct, and with error-propagation by exploring transition sequences that are not optimal in the sense that they do not derive the gold tree, while training the parser to perform the optimal transitions on these non-optimal configurations. We show that both of these properties improve the accuracy of a deterministic left-to-right arc-eager parser by over 1.2 LAS points on average and up to almost 3 LAS points on some data sets, as compared to the conventional training procedure.

2 Background

2.1 Arc-Eager Transition-Based Dependency Parsing

Given a set L of dependency labels, we define a *dependency graph* for a sentence $x = w_1, \ldots, w_n$ as a labeled directed graph $G = (V_x, A)$, consisting of a set of nodes $V_x = \{0, 1, \ldots, n\}$, where each node i corresponds to the linear position of a word w_i in the sentence, plus an extra artificial root node 0, and a set of labeled arcs $A \subseteq V_x \times L \times V_x$, where each arc (i, l, j) represents a dependency with head w_i, dependent w_j, and label l. In order for a dependency graph to be well-formed, we usually require it to be a *dependency tree*, which is a directed spanning tree rooted at the node 0. In this paper, we further restrict our attention to dependency trees that are *projective*, that is, where the presence of an arc (i, l, j) entails that there is directed path from i to every node k such that $\min(i, j) < k < \max(i, j)$. Figure 1 shows a projective dependency tree for a simple English sentence.

In the arc-eager transition system of Nivre (2003), a *parser configuration* is a triple $c = (\Sigma, B, A)$ such that Σ (referred to as the *stack*) and B (the *buffer*) are disjoint sublists of the nodes V_x of some sentence x, and A is a set of dependency arcs over V_x (and some label set L); we take the initial configuration for a sentence $x = w_1, \ldots, w_n$ to be $c_s(x) = ([0], [1, \ldots, n], \{\})$; and we take a terminal configuration to be any configuration of the form $c = (\Sigma, [\,], A)$ (for any stack Σ and arc set A).[2] There are four types of *transitions*, defined formally in Figure 2, for going from one configuration to the next:

1. A LEFT-ARC$_l$ transition (for any dependency label l) adds the arc (b, l, s) to A, where s is the node on top of the stack and b is the first node in the buffer, and pops the stack. It has as a precondition that the token s is not the artificial root node 0 and does not already have a head.

[2] As is customary, we use the variables σ and β for arbitrary sublists of the stack and the buffer, respectively. For reasons of perspicuity, we will write Σ with its head (top) to the right and B with its head to the left. Thus, $c = (\sigma|s, b|\beta, A)$ is a configuration with the node s on top of the stack Σ and the node b as the first node in the buffer B.

Transition		Precondition
LEFT-ARC$_l$	$(\sigma\|i, j\|\beta, A) \Rightarrow (\sigma, j\|\beta, A \cup \{(j, l, i)\})$	$\neg[i = 0] \wedge \neg\exists k \exists l'[(k, l', i) \in A]$
RIGHT-ARC$_l$	$(\sigma\|i, j\|\beta, A) \Rightarrow (\sigma\|i\|j, \beta, A \cup \{(i, l, j)\})$	
REDUCE	$(\sigma\|i, \beta, A) \Rightarrow (\sigma, \beta, A)$	$\exists k \exists l[(k, l, i) \in A]$
SHIFT	$(\sigma, i\|\beta, A) \Rightarrow (\sigma\|i, \beta, A)$	

Figure 2: Transitions for the arc-eager transition system

2. A RIGHT-ARC$_l$ transition (for any dependency label l) adds the arc (s, l, b) to A, where s is the node on top of the stack and b is the first node in the buffer, and pushes the node b onto the stack.

3. The REDUCE transition pops the stack and is subject to the preconditions that the top token has a head.

4. The SHIFT transition removes the first node in the buffer and pushes it onto the stack.

A *transition sequence* for a sentence x is a sequence $C_{0,m} = (c_0, c_1, \ldots, c_m)$ of configurations, such that c_0 is the initial configuration $c_s(x)$, c_m is a terminal configuration, and there is a legal transition t such that $c_i = t(c_{i-1})$ for every i, $1 \leq i \leq m$. The dependency graph derived by $C_{0,m}$ is $G_{c_m} = (V_x, A_{c_m})$, where A_{c_m} is the set of arcs in c_m. The arc-eager system is sound and complete for the class of projective dependency forests, meaning that every legal transition sequence derives a projective dependency forest (soundness) and that every projective dependency forest is derived by at least one transition sequence (completeness) (Nivre, 2008). A projective dependency forest is a dependency graph where every connected component is a projective tree, with the special case of a projective dependency tree in case there is only one connected component. Hence, the arc-eager transition system is also complete (but not sound) for the class of projective dependency trees.

2.2 Static Oracles for Transition-Based Parsing

In the transition-based framework, parsing is implemented as search for an optimal transition sequence, that is, a transition sequence that derives the correct parse tree for a given sentence. The simplest version of this, and also the most efficient, is to train a classifier to predict the single best transition in each configuration and use a greedy deterministic procedure to derive a single dependency graph, which results in linear-time parsing provided that each transition can be predicted and executed in constant time (Nivre, 2003, 2008). The classifier is trained on a set of configuration-transition pairs, which can be derived from a dependency treebank by finding an optimal transition sequence for each sentence x with gold tree $G_{\text{gold}} = (V_x, A_{\text{gold}})$.

Algorithm 1 defines the standard oracle function used to find the next transition t for a configuration c and gold tree $G_{\text{gold}} = (V_x, A_{\text{gold}})$. This algorithm is provably correct in the sense that, for every sentence x and projective dependency tree $G_{\text{gold}} = (V_x, A_{\text{gold}})$, if we initialize c to $c_s(x)$ and repeatedly execute the oracle transition, then we derive exactly $G_{\text{gold}} = (V_x, A_{\text{gold}})$ (Nivre, 2006). Nevertheless, it has two important limitations.

Algorithm 1 Standard oracle for arc-eager dependency parsing

if $c = (\sigma|i, j|\beta, A)$ and $(j, l, i) \in A_{\text{gold}}$ **then**
  $t \leftarrow$ LEFT-ARC$_l$
else if $c = (\sigma|i, j|\beta, A)$ and $(i, l, j) \in A_{\text{gold}}$ **then**
  $t \leftarrow$ RIGHT-ARC$_l$
else if $c = (\sigma|i, j|\beta, A)$ and $\exists k[k < i \wedge \exists l[(k, l, j) \in A_{\text{gold}} \vee (j, l, k) \in A_{\text{gold}}]]$ **then**
  $t \leftarrow$ REDUCE
else
  $t \leftarrow$ SHIFT
return t

The first is that it ignores spurious ambiguity in the transition system, that is, cases where a given dependency tree can be derived in more than one way. The dependency tree in Figure 1 is derived by two distinct transition sequences:[3]

(1) SH, LA_{SBJ}, RA_{PRD}, RA_{IOBJ}, SH, LA_{DET}, RE, RA_{DOBJ}, RE RA_{P}
(2) SH, LA_{SBJ}, RA_{PRD}, RA_{IOBJ}, RE, SH, LA_{DET}, RA_{DOBJ}, RE RA_{P}

Algorithm 1 will predict (1) but not (2). More generally, whenever there is a SH-RE ambiguity, which is the only ambiguity that exists in the arc-eager system, the oracle prediction will always be SH. In this way, the oracle implicitly defines a canonical transition sequence for every tree.

The second limitation is that we have no guarantee for what happens if we apply the oracle to a configuration that does not belong to the canonical transition sequence. In fact, it is easy to show that the oracle prediction in such cases can be suboptimal. For example, suppose that we erroneously choose the SH transition instead of RA_{IOBJ} after the first three transitions in sequence (1). This results in the following parser configuration:

$$([0, 2, 3], [4, 5, 6], \{(0, \text{PRD}, 2), (2, \text{SBJ}, 1)\})$$

Starting from this configuration, the oracle defined by Algorithm 1 will predict SH, LA_{DET}, SH, SH, which derives the dependency graph in the left-hand side of Figure 3. Using labeled attachment score to measure loss, this graph has a loss of 3 compared to the correct tree in Figure 1, since it fails to include the arcs $(2, \text{IOBJ}, 3)$, $(2, \text{DOBJ}, 5)$, $(2, \text{P}, 6)$.[4] However, if we instead apply the transitions SH, LA_{DET}, LA_{DET}, RA_{DOBJ}, RE, RA_{P}, we end up with the tree in the right-hand side of Figure 3, which only has a loss of 1.

We say that Algorithm 1 defines a *static* oracle, because it produces a single static sequence of transitions that is supposed to be followed in its entirety. The main contribution of this paper is the notion of a *dynamic* oracle, which does not presuppose a single canonical transition sequence for each dependency tree and which can dynamically adapt to arbitrary configurations that arise during parsing and still make optimal predictions.

[3] To save space, we sometimes use the following abbreviations: LA_l = LEFT-ARC$_l$, RA_l = RIGHT-ARC$_l$, RE = REDUCE, SH = SHIFT.

[4] In most practical parser implementations, this graph is converted into a tree by adding arcs from the root node to all words that lack a head. However, the loss will be exactly the same.

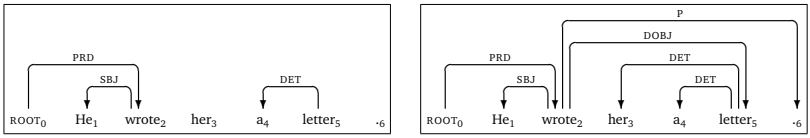


Figure 3: Dependency graphs with loss 3 (left) and loss 1 (right)

3 Dynamic Parsing Oracles

We want to define an oracle that (a) allows more than one transition sequence for a given tree and (b) makes optimal predictions in all configurations (not only configurations that are part of a globally optimal transition sequence). It follows from the first requirement that the oracle should define a *relation* from configurations to transitions, rather than a function, and we will represent it as a boolean function $o(t;c,G_{\text{gold}})$, which returns **true** just in case t is optimal in c relative to G_{gold}. But what does it mean for a transition to be optimal?

Intuitively, a transition t is optimal if it does not commit us to a parsing error, which we take to mean that the best dependency tree reachable from c is also reachable from $t(c)$. Consider the set of all dependency trees that are reachable from c. From this set, we pick the set of trees that minimize some loss function relative to G_{gold} and say that t is optimal if and only if at least one tree in this set is still reachable from $t(c)$. More precisely, we define the *cost* of t to be the difference in loss between the best tree reachable in c and the best tree reachable in $t(c)$ and say that t is optimal if it has zero cost. In the special case where the gold tree G_{gold} is reachable from c, the set of trees that minimize the loss function is the singleton set containing G_{gold}, which entails that t is optimal if and only if G_{gold} is still reachable from $t(c)$. Let us now try to make this precise.

3.1 Defining the Oracle

First, we define the *loss* $\mathscr{L}(G,G_{\text{gold}})$ of a dependency graph $G=(V_x,A)$ with respect to the gold tree $G_{\text{gold}}=(V_x,A_{\text{gold}})$ to be the number of arcs that are in G_{gold} but not in G:

$$\mathscr{L}(G,G_{\text{gold}}) = |A_{\text{gold}} \setminus A|$$

We then say that a dependency graph $G=(V_x,A)$ for a sentence x is *reachable* from a non-terminal configuration c for x, written $c \rightsquigarrow G$ if and only if there is a non-empty sequence of transitions $t_1,\ldots,t_m$ such that $[t_m \circ \cdots \circ t_1](c) = (\Sigma,[],A)$ and $G=(V_x,A)$.

Next, we define the *cost* $\mathscr{C}(t;c,G_{\text{gold}})$ of the transition t in the configuration c relative to the gold tree G_{gold} as the loss difference between the minimum loss tree reachable before and after the transition:

$$\mathscr{C}(t;c,G_{\text{gold}}) = \left[\min_{G:t(c)\rightsquigarrow G} \mathscr{L}(G,G_{\text{gold}})\right] - \left[\min_{G:c\rightsquigarrow G} \mathscr{L}(G,G_{\text{gold}})\right]$$

Note that, by definition, there must be at least one zero cost transition for every configuration c and gold tree G_{gold}. To see why, let G be some dependency graph with minimum loss reachable from c. Since G is reachable from c, there must be at least one transition t such that G is reachable from $t(c)$. And since $\mathscr{L}(G,G_{\text{gold}}) - \mathscr{L}(G,G_{\text{gold}}) = 0$, it follows that $\mathscr{C}(t;c,G_{\text{gold}}) = 0$.

Finally, we define the oracle $o(t; c, G_{\text{gold}})$ to return **true** just in case t has zero cost relative to c and G_{gold}:

$$o(t; c, G_{\text{gold}}) = \begin{cases} \textbf{true} & \text{if } \mathscr{C}(t; c, G_{\text{gold}}) = 0 \\ \textbf{false} & \text{otherwise} \end{cases}$$

3.2 A Dynamic Oracle for Arc-Eager Parsing

In order to implement the dynamic oracle in practice, we need an efficient method for computing the cost of each transition in a given configuration. A key property of the arc-eager system (stated here without proof) is that a dependency graph $G = (V_x, A)$ is reachable from a configuration c if G is a projective forest and if each individual arc in A is reachable in c. In the Arc-Eager system, an arc (i, l, j) is reachable in $c = (\Sigma, B, A')$ if either (i, l, j) is already in A' (since arcs can never be removed) or if $\min(i, j)$ is in Σ or B, $\max(i, j)$ is in B, and there is no arc in A that already assigns a head to j (since it is always possible to reach a later configuration where $\min(i, j)$ is at the top of the stack and $\max(i, j)$ is at the head of the buffer, in which case the arc can be added in a LEFT-ARC$_l$ or RIGHT-ARC$_l$ transition).

Given that our loss function (and hence our cost function) also decomposes into the loss of individual arcs, we can compute the cost of each transition by simply counting the gold arcs that are no longer reachable after that transition. We do this on a case by case basis. In all the cases, we assume a configuration c of the form $c = (\sigma|s, b|\beta, A)$.[5]

- $\mathscr{C}$(LEFT-ARC$_l$; c, G_{gold}): Adding the arc (b, l, s) and popping s from the stack means that s will not be able to acquire any head or dependents in β. The cost is therefore the number of arcs in A_{gold} of the form (k, l', s) or (s, l', k) such that $k \in \beta$. Note that the cost is 0 for the trivial case where $(b, l, s) \in A_{\text{gold}}$, but also for the case where b is not the gold head of s but the real head is not in β (due to an erroneous previous transition) and there are no gold dependents of s in β.[6]

- $\mathscr{C}$(RIGHT-ARC$_l$; c, G_{gold}): Adding the arc (s, l, b) and pushing b onto the stack means that b will not be able to acquire any head in σ or β, nor any dependents in σ. The cost is therefore the number of arcs in A_{gold} of the form (k, l', b), such that $k \in \sigma$ or $k \in \beta$, or of the form (b, l', k) such that $k \in \sigma$. Note again that the cost is 0 for the trivial case where $(s, l, b) \in A_{\text{gold}}$, but also for the case where s is not the gold head of b but the real head is not in σ or β (due to an erroneous previous transition) and there are no gold dependents of b in σ.

- $\mathscr{C}$(REDUCE; c, G_{gold}): Popping s from the stack means that s will not be able to acquire any dependents in $b|\beta$. The cost is therefore the number of arcs in A_{gold} of the form (s, l', k) such that $k \in b|\beta$. While it may seem that a gold arc of the form (k, l, s) should be accounted for as well, note that a gold arc of that form, if it exists, is already accounted for by a previous (erroneous) RIGHT-ARC$_l$ transition when s acquired its head.

- $\mathscr{C}$(SHIFT; c, G_{gold}): Pushing b onto the stack means that b will not be able to acquire any head or dependents in $s|\sigma$. The cost is therefore the number of arcs in A_{gold} of the form (k, l', b) or (b, l', k) such that $k \in s|\sigma$.

[5] This is a slight abuse of notation, since for the SHIFT transition s may not exist, and for the REDUCE transition b may not exist.

[6] One may want to associate a lower cost with cases in which the arc endpoints are correct and only the label is wrong. This extension is trivial.

$$
\begin{aligned}
\mathscr{C}(\text{LA}_l;\ c=(\sigma|s,b|\beta,A),G_g) &= \left|\{(k,l',s)\in A_g \mid k\in\beta\}\cup\{(s,l',k)\in A_g \mid k\in\beta\}\right| \\
\mathscr{C}(\text{RA}_l;\ c=(\sigma|s,b|\beta,A),G_g) &= \left|\{(k,l',b)\in A_g \mid k\in\sigma\vee k\in\beta\}\cup\{(b,l',k)\in A_g \mid k\in\sigma\}\right| \\
\mathscr{C}(\text{RE};\ c=(\sigma|s,\beta,A),G_g) &= \left|\{(s,l',k)\in A_g \mid k\in\beta\}\right| \\
\mathscr{C}(\text{SH};\ c=(\sigma,b|\beta,A),G_g) &= \left|\{(k,l',b)\in A_g \mid k\in\sigma\}\cup\{(b,l',k)\in A_g \mid k\in\sigma\}\right|
\end{aligned}
$$

Figure 4: Transition costs for the arc-eager transition system with gold tree $G_g = (V_x, A_g)$.

The computation of transition costs is summarized in Figure 4. We can now return to the example in Section 2.2 and analyze the behavior of the static oracle in the presence of erroneous transitions. In the example there, the last two SHIFT transitions predicted by the static oracle each has a cost of 1, because they each lose an arc going into the first word of the buffer. By contrast, the transition LEFT-ARC$_{\text{DET}}$ in place of the first SHIFT has a cost of 0, despite the fact that the arc $(5, \text{DET}, 3)$ is not in the gold tree, because it does not eliminate any gold arc that is still reachable – the cost of the incorrect attachment is already accounted for in the cost of the erroneous SHIFT action.

After defining the concept of a *dynamic oracle* which is correct over the entire configuration space of a transition system and providing a concrete instantiation of it for the arc-eager transition system, we now go on to present one useful application of such an oracle.

4 Training Parsers with the Dynamic Oracle

Greedy transition-based parsers trained with static oracles are known to suffer from error propagation (McDonald and Nivre, 2007). We may hope to mitigate the error propagation problem by letting the parser explore larger portions of the configuration space during training and learn how to behave optimally also after committing previous errors. While this is not possible with the usual static oracles, the dynamic oracle defined above allows us to do just that, as it returns a set of optimal transitions for each possible configuration.

Algorithm 2 is a standard online training algorithm for transition-based dependency parsers using a static oracle. Given a training sentence x with gold tree G_{gold}, it starts in the initial configuration $c_s(x)$ and repeatedly predicts a transition t_p given its current feature weights $\mathbf{w}$ and compares this to the transition t_o predicted by the static oracle. If the model prediction is different from the oracle prediction, the feature weights are updated, but the new configuration is always derived using the oracle transition (line 10), which means that only configurations in the canonical oracle-induced transition sequence are explored.[7]

Algorithm 3 is a modification of the standard algorithm which makes use of the dynamic oracle to explore a larger part of the configuration space. The first difference is in line 7, where the new algorithm, instead of getting the single prediction of the static oracle, gets the set of

[7]Some readers may be more familiar with a two-stage process in which first the oracle is used to create oracle transition sequences from the entire training set, which are then transformed to individual training examples and passed on to an external classifier. This process is equivalent to the one in Algorithm 2 in case the external classifier is a multiclass perceptron (or any other online classifier), with the only difference being that the training examples are generated on-the-fly whenever they are needed. The online formulation is used to facilitate a smooth transition to Algorithm 3.

Algorithm 2 Online training with a static oracle

```
 1: w ← 0
 2: for I = 1 → ITERATIONS do
 3:     for sentence x with gold tree G_gold in corpus do
 4:         c ← c_s(x)
 5:         while c is not terminal do
 6:             t_p ← arg max_t w · φ(c, t)
 7:             t_o ← o(c, G_gold)
 8:             if t_p ≠ t_o then
 9:                 w ← w + φ(c, t_o) − φ(c, t_p)
10:             c ← t_o(c)
11: return w
```

transitions that have zero cost according to the dynamic oracle. The weight update is then performed only if the model prediction does not have zero cost (lines 9–10), which means that updates no longer need to reflect a single canonical transition sequence. Finally, the transition used to update the parser configuration is no longer the single transition predicted by the static oracle, but a transition that is chosen by the function CHOOSE_NEXT, which may be a transition that does not have zero-cost (lines 11-12). In our current implementation, CHOOSE_NEXT is conditioned on the predicted transition t_p, the set of zero cost transitions, and the iteration number. However, more elaborate conditioning schemes are also possible.

We propose two versions of the CHOOSE_NEXT function. In the first version, $\text{CHOOSE\_NEXT}_{\text{AMB}}$, the training algorithm only follows optimal (zero cost) transitions but permits spurious ambiguity by following the model prediction t_p if this has zero cost and a random zero cost transition otherwise. In the second version, $\text{CHOOSE\_NEXT}_{\text{EXP}}$, the training algorithm also explores non-optimal transitions. More precisely, after the first k training iterations, it follows the model prediction t_p regardless of its cost in 100(1-p)% of the cases and falls back on $\text{CHOOSE\_NEXT}_{\text{AMB}}$ in the remaining cases. It is worth noting that Algorithm 3 subsumes Algorithm 2 as a special case if we define ZERO_COST to contain only the prediction t_o of the static oracle, and define CHOOSE_NEXT to always return t_o.

The novel training algorithm presented here is based on perceptron learning.[8] Since the dynamic oracle provides a cost for every transition-configuration pair, it could be used also for cost-sensitive learning. Our preliminary attempts with cost-sensitive learning through the max-loss and prediction-based passive-aggressive algorithms of Crammer et al. (2006) show that the cost-sensitive variants of the algorithms indeed improve upon the non-cost-sensitive variants. However, the best passive-aggressive results were still significantly lower than those obtained using the averaged perceptron. We do not elaborate on cost-sensitive training in this work, and leave this direction for future investigation.

5 Experiments

We present experiments comparing greedy arc-eager transition-based parsers trained (a) using the static oracle (Algorithm 2), (b) using the dynamic oracle with spurious ambiguity (Algorithm 3 with $\text{CHOOSE\_NEXT}_{\text{AMB}}$), and (c) using the dynamic oracle with spurious ambiguity

[8] In practice, we use an averaged perceptron, although this is not reflected in the algorithm descriptions above.

Algorithm 3 Online training with a dynamic oracle

```
1: w ← 0
2: for I = 1 → ITERATIONS do
3:     for sentence x with gold tree G_gold in corpus do
4:         c ← c_s(x)
5:         while c is not terminal do
6:             t_p ← arg max_t w · φ(c, t)
7:             ZERO_COST ← {t | o(t; c, G_gold) = true}
8:             t_o ← arg max_{t∈ZERO_COST} w · φ(c, t)
9:             if t_p ∉ ZERO_COST then
10:                w ← w + φ(c, t_o) − φ(c, t_p)
11:            t_n ← CHOOSE_NEXT(I, t_p, ZERO_COST)
12:            c ← t_n(c)
13: return w
```

```
1: function CHOOSE_NEXT_AMB(I, t, ZERO_COST)
2:     if t ∈ ZERO_COST then
3:         return t
4:     else
5:         return RANDOM_ELEMENT(ZERO_COST)
```

```
1: function CHOOSE_NEXT_EXP(I, t, ZERO_COST)
2:     if I > k and RAND() > p then
3:         return t
4:     else
5:         return CHOOSE_NEXT_AMB(I, t, ZERO_COST)
```

and non-optimal transitions (Algorithm 3 with CHOOSE_NEXT$_{\text{EXP}}$). We evaluate the models on a wide range of English data sets, as well as the data sets from the CoNLL 2007 shared task on multilingual dependency parsing (Nivre et al., 2007).

The parser is a greedy transition-based parser using the arc-eager transition system of Nivre (2003, 2008) with the feature representations defined by Zhang and Nivre (2011). As our primary goal is to compare the training methods, and not to achieve the highest possible score for each data set, we use the exact same feature representations and training parameters across all experiments. Specifically, we train an averaged perceptron model for 15 iterations. When using CHOOSE_NEXT$_{\text{EXP}}$, we set $k = 2$ and $p = 0.1$, meaning that the algorithm allows non-optimal transitions in 90% of the cases, starting from the third training iteration. Note that many of these transitions will nevertheless be correct, as the first training iterations already put the model in a good region of the parameter space.

The English model is trained on Sections 2–21 of the Penn-WSJ Treebank (Marcus et al., 1993), converted to Stanford basic dependencies (de Marneffe et al., 2006), with part-of-speech tags assigned by a structured-perceptron tagger trained on the same corpus with 4-fold jack-knifing. We use Section 22 to tune parameters, and we evaluate on the following data sets, which are also converted to the same dependency scheme, and pos-tagged using the same

	WSJ22	WSJ23	BNC	BRN	FTBL	QTB	ANS	EML	GRPS	REVS	BLGS
Unlabeled Attachment Scores											
Static	90.31	89.88	82.79	85.11	78.85	86.80	78.81	79.23	81.21	80.61	83.40
Dynamic-ambiguity	90.42	90.18	82.98	85.41	79.36	87.29	79.19	79.56	81.18	80.96	83.61
Dynamic-explore	91.24	90.96	84.17	86.22	80.04	87.50	80.21	80.04	82.08	81.81	84.72
Labeled Attachment Scores											
Static	87.88	87.69	78.47	81.15	74.69	73.69	73.60	74.95	77.15	75.87	79.66
Dynamic-ambiguity	87.95	87.83	78.69	81.47	75.05	73.91	73.90	75.29	77.16	76.19	79.91
Dynamic-explore	88.76	88.72	79.75	82.30	75.82	74.36	74.95	75.85	78.11	77.06	81.09

Table 1: Results on the English data sets

structured-perceptron tagger, trained on the entire training set.

- WSJ22: Section 22 of the Penn-WSJ Treebank (development set).
- WSJ23: Section 23 of the Penn-WSJ Treebank (test set).
- BNC: 1,000 manually annotated sentences from the British National Corpus (Foster and van Genabith, 2008).
- BRN: The entire Brown Corpus (Kucera and Francis, 1967).
- FTBL: The entire Football Corpus (Foster et al., 2011).
- QB: The entire QuestionBank (Judge et al., 2006).
- ANS, EML, GRPS, REVS, BLGS: the question-answers, emails, newsgroups, reviews and weblogs portions of the English Web Treebank (Bies et al., 2012; Petrov and McDonald, 2012).

The CoNLL models are trained on the dedicated training set for each of the ten languages and evaluated on the corresponding test set, with gold standard part-of-speech tags in both cases. Since the arc-eager parser can only handle projective dependency trees, all trees in the training set are projectivized before training, using the baseline pseudo-projective transformation in Nivre and Nilsson (2005). However, non-projective trees are kept intact in the test sets for evaluation. We include all ten languages from the CoNLL 2007 shared task:

- ARA: Arabic (Hajič et al., 2004)
- BAS: Basque (Aduriz et al., 2003)
- CAT: Catalan (Martí et al., 2007)
- CHI: Chinese (Chen et al., 2003)
- CZE: Czech (Hajič et al., 2001; Böhmová et al., 2003)
- ENG: English (Marcus et al., 1993)
- GRE: Greek (Prokopidis et al., 2005)
- HUN: Hungarian (Czendes et al., 2005)

	ARA	BAS	CAT	CHI	CZE	ENG	GRE	HUN	ITA	TUR
Unlabeled Attachment Scores										
Static	80.60	74.10	91.21	84.13	78.00	86.24	79.16	77.75	84.11	79.02
Dynamic-ambiguity	80.72	74.90	91.09	83.62	78.38	86.83	79.48	76.17	84.52	78.97
Dynamic-explore	83.06	76.10	92.01	84.65	79.54	88.81	80.66	77.10	84.77	78.84
Labeled Attachment Scores										
Static	71.04	64.42	85.96	79.75	69.49	84.90	70.94	68.10	79.93	68.80
Dynamic-ambiguity	71.06	65.18	85.73	79.24	69.39	85.56	71.88	66.99	80.63	68.58
Dynamic-explore	73.54	66.77	86.60	80.74	71.32	87.60	73.83	68.23	81.02	68.76

Table 2: Results on the CoNLL 2007 data sets

- ITA: Italian (Montemagni et al., 2003)
- TUR: Turkish (Oflazer et al., 2003)

Table 1 gives the results for the English model, while Table 2 presents the multilingual evaluation. In both cases, we present unlabeled and labeled attachment scores excluding punctuation.

For the English data sets, we see that adding spurious ambiguity (Dynamic-ambiguity) generally improves both labeled and unlabeled attachment scores by up to 0.5 percent absolute. The only exception is GRPS, where there is a small decrease in unlabeled attachment score. When the training procedure in addition explores non-optimal transitions (Dynamic-explore), the improvement is even greater, in some cases up to about 1.5 percentage points, which is quite substantial given that our baseline parser already performs at the state-of-the-art level for greedy deterministic transition-based parsers on English Stanford-dependencies.[9]

For the CoNLL data sets, results for the Dynamic-ambiguity condition are mixed causing a drop in accuracy for some data sets, but the Dynamic-explore condition makes up for it and brings substantial improvement in accuracy for all except two languages. We see very substantial improvements in both UAS and LAS for Arabic, Basque, Czech, English and Greek, as well as improvements for Catalan, Chinese and Italian. The average LAS improvement across all the CoNLL datasets is 1.5 LAS points. The only two exceptions are Hungarian and Turkish, where unlabeled attachment scores drop slightly as a result of not using the static oracle, and labeled attachment scores are practically unaffected. More analysis is needed to find out what is going on for these languages, but it is likely that results could be improved with language-specific tuning.[10]

Overall, the experimental results show a considerable improvement in the accuracy of deterministic linear-time classifier-based dependency parsing through training procedures that explore a larger part of the search space than traditional methods based on static oracles.

[9] Note that the web data sets (ANS, EML, GRPS, REVS, BLGS) are annotated according to the Ontonotes corpus guidelines, which are somewhat different than the original Penn Treebank guidelines used in the training corpora. In particular, base-NPs in the web data sets are more nested. Our scores on these data sets are thus artificially lower than they could be. We could get better scores for these data sets for all training conditions by training on the Ontonotes corpora instead, but as our main concern is not in achieving the best scores, we opted for the simpler experimental setup.

[10] Language-specific tuning is likely to improve results for the other languages as well – we did not perform any language-specific tuning, and it is well established that individual languages parsing accuracies can greatly benefit from tuning of the feature set, the transition system being used and the learning parameters (Hall et al., 2007).

6 Related Work

Deterministic classifier-based dependency parsing is an instance of independent sequential classification-based structured prediction. In sequential classification models, such as Maximum-Entropy Markov Models (McCallum et al., 2000), a structured output is produced by repeated application of a locally trained classifier, where each classification decision is conditioned on the structure created by previous decisions. Several methods have been developed to cope with error propagation in sequential classification, including stacked sequential learning (Cohen and Carvalho, 2005), LaSO (Daumé III and Marcu, 2005), Searn (Daumé III et al., 2009) and its followers SMILe (Ross and Bagnell, 2010) and DAgger (Ross et al., 2011).

While stacked learning is well suited for sequence prediction tasks such as tagging and chunking, it is not clear how to apply it to parsing.[11] Searn and the closely related DAgger algorithm are more promising for dealing with the complexity of dependency parsing, but it appears that previous attempts to apply Searn-based learning to dependency parsing have been unsuccessful. A key component in the specification of a Searn learning algorithm is an *optimal policy* mapping states in the search space (such as parser configurations) to optimal outcomes (such as transitions). Attempts to approximate the optimal policy for parsing using static oracles are unlikely to work very well, since a static oracle is only correct for a small subset of the search space. The dynamic oracle introduced in this paper, which is correct for arbitrary parser configurations, can be used to define an optimal policy for Searn-based learning. Both Searn and DAgger require several complete training rounds over the entire data set and take a relatively long time to train. We instead use a simpler online algorithm which can be viewed as a stochastic approximation of the DAgger algorithm, which is itself heavily inspired by the Searn algorithm.

Recent work on beam search and structured prediction for transition-based dependency parsing has shown that parsing accuracy can be improved considerably if models are trained to perform beam search instead of greedy one-best search, and if training is done using a global structured learning objective instead of local learning of individual decisions (Zhang and Clark, 2008; Zhang and Nivre, 2011; Bohnet and Kuhn, 2012; Huang et al., 2012). Like our method, beam search with global structured learning mitigates the effects of error propagation by exploring non-canonical configurations at training time, but the use of a beam reduces parsing speed by a factor that is roughly proportional to the size of the beam, making parsing less efficient. Our method in contrast still trains classifiers to perform local decisions, and thus incurs no efficiency penalty at parsing time, but each local decision is trained to take into account the consequences of previous, possibly erroneous, decisions. Although we may not be able to reach the accuracy level of a beam-search parser, we show that a substantial improvement in accuracy is possible also for a purely deterministic classifier-based parser, making our method suitable for training accurate parsers in situations where maximum efficiency is needed, e.g., when there is a need to process very large corpora. Integrating our dynamic oracle in the training procedure for a transition-based parser with beam search is an interesting question for future work.

The work that probably comes closest to ours is Choi and Palmer (2011), who improve the accuracy of a greedy transition-based dependency parser through an iterative training procedure that they call bootstrapping. They start by training a classifier using a standard static oracle for a hybrid transition system combining elements of the arc-eager system and the algorithm of

[11] Stacked learning has been explored to some extent in the context of parsing for integrating approximate higher order features as well as for combining the predictions of different parsers (Nivre and McDonald, 2008; Martins et al., 2008).

Covington (2001). In a second step, they then use this classifier to parse the training corpus, creating one new training instance for every configuration visited during parsing, using an adapted version of the static oracle to predict the optimal transition for each configuration. They iterate this procedure as long as parsing accuracy improves on a held-out development set and report improvements in parsing accuracy of about 0.5 percent absolute for English and almost 2 percent absolute for Czech. The main difference compared to our approach, except for the fact that they use a different transition system, is that their method for finding the optimal transition after the first training round is heuristic and does not guarantee that the best parse is still reachable.

Finally, Cohen et al. (2012) tackle the problem of spurious ambiguity for static oracles by eliminating ambiguity from the underlying transition system instead of modifying the oracle. They show how this can be achieved for the arc-standard system of Nivre (2004) as well as the non-projective extension by Attardi (2006). It is still an open question whether their technique can also be applied to the arc-eager system targeted in this paper.

7 Conclusion

We have highlighted the shortcoming of traditional static oracles used to train transition-based dependency parsers, and instead proposed the notion of a *dynamic oracle*, which allows more than one correct transition sequence in the case of spurious ambiguity, and which can predict an optimal transition also for non-optimal configurations. We have defined a concrete dynamic oracle for the arc-eager transition system and showed how it can be used in online training of greedy deterministic parsers.

Greedy deterministic transition-based dependency parsers are among the most efficient systems available for syntactic parsing of natural language. In terms of parsing accuracy, they perform near the state-of-the-art level for many languages but tend to suffer from prediction errors and subsequent error propagation. This problem can be mitigated by using our proposed training method. Experimental results for English show consistent improvements in parsing accuracy of up to almost 1.5 percent absolute on a wide range of data sets. Experimental results on the ten languages from the CoNLL 2007 shared task on dependency parsing show significant improvements of up to almost 3 LAS points for some languages, but there are also few cases where we see little or no improvement in parsing accuracy, a phenomenon that requires further investigation. Other topics for future research is the effective use of cost-sensitive learning instead of the perceptron loss used in this paper, the derivation of dynamic oracles for other transition systems, and utilizing the dynamic oracles in non-greedy settings, such as beam-search parsers.

References

Aduriz, I., Aranzabe, M. J., Arriola, J. M., Atutxa, A., Díaz de Ilarraza, A., Garmendia, A., and Oronoz, M. (2003). Construction of a Basque dependency treebank. In *Proceedings of the 2nd Workshop on Treebanks and Linguistic Theories (TLT)*, pages 201–204.

Attardi, G. (2006). Experiments with a multilanguage non-projective dependency parser. In *Proceedings of the 10th Conference on Computational Natural Language Learning (CoNLL)*, pages 166–170.

Bies, A., Mott, J., Warner, C., and Kulick, S. (2012). English web treebank. Linguistic Data Consortium, LDC2012T13.

Böhmová, A., Hajič, J., Hajičová, E., and Hladká, B. (2003). The Prague Dependency Treebank: A three-level annotation scenario. In Abeillé, A., editor, *Treebanks: Building and Using Parsed Corpora*, pages 103–127. Kluwer.

Bohnet, B. and Kuhn, J. (2012). The best of bothworlds – a graph-based completion model for transition-based parsers. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 77–87.

Chen, K., Luo, C., Chang, M., Chen, F., Chen, C., Huang, C., and Gao, Z. (2003). Sinica treebank: Design criteria, representational issues and implementation. In Abeillé, A., editor, *Treebanks*, pages 231–248. Kluwer.

Choi, J. D. and Palmer, M. (2011). Getting the most out of transition-based dependency parsing. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 687–692.

Cohen, S. B., Gómez-Rodríguez, C., and Satta, G. (2012). Elimination of spurious ambiguity in transition-based dependency parsing. Technical report.

Cohen, W. W. and Carvalho, V. R. (2005). Stacked sequential learning. In *Proceedings of the International Joint Conference on Artificial Intelligence*, pages 671–676.

Covington, M. A. (2001). A fundamental algorithm for dependency parsing. In *Proceedings of the 39th Annual ACM Southeast Conference*, pages 95–102.

Crammer, K., Dekel, O., Keshet, J., Shalev-Shwartz, S., and Singer, Y. (2006). Online passive-aggressive algorithms. *Journal of Machine Learning Research*, 7:551–585.

Czendes, D., Csirik, J., Guimóthy, and Kocsor, A. (2005). *The Szeged Treebank*. Springer.

Daumé III, H., Langford, J., and Marcu, D. (2009). Search-based structured prediction. *Machine Learning*, 75:297–325.

Daumé III, H. and Marcu, D. (2005). Learning as search optimization: Approximate large margin methods for structured prediction. In *Proceedings of the 22nd International Conference on Machine Learning*, pages 169–176.

de Marneffe, M.-C., MacCartney, B., and Manning, C. D. (2006). Generating typed dependency parses from phrase structure parses. In *Proceedings of the 5th International Conference on Language Resources and Evaluation (LREC)*.

Foster, J., ÇetinoÄŸlu, O., Wagner, J., and van Genabith, J. (2011). Comparing the use of edited and unedited text in parser self-training. In *Proceedings of the 12th International Conference on Parsing Technologies*, pages 215–219.

Foster, J. and van Genabith, J. (2008). Parser evaluation and the BNC: Evaluating 4 constituency parsers with 3 metrics. In *Proceedings of the 6th International Conference on Language Resources and Evaluation (LREC)*.

Goldberg, Y. and Elhadad, M. (2010). An efficient algorithm for easy-first non-directional dependency parsing. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL HLT)*, pages 742–750.

Hajič, J., Smrž, O., Zemánek, P., Šnaidauf, J., and Beška, E. (2004). Prague Arabic Dependency Treebank: Development in data and tools. In *Proceedings of the NEMLAR International Conference on Arabic Language Resources and Tools*.

Hajič, J., Vidova Hladka, B., Panevová, J., Hajičová, E., Sgall, P., and Pajas, P. (2001). Prague Dependency Treebank 1.0. LDC, 2001T10.

Hall, J., Nilsson, J., Nivre, J., Eryiğit, G., Megyesi, B., Nilsson, M., and Saers, M. (2007). Single malt or blended? A study in multilingual parser optimization. In *Proceedings of the CoNLL Shared Task of EMNLP-CoNLL 2007*, pages 933–939.

Huang, L., Fayong, S., and Guo, Y. (2012). Structured perceptron with inexact search. In *Proceedings of Human Language Technologies: The 2012 Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL HLT)*.

Judge, J., Cahill, A., and van Genabith, J. (2006). Questionbank: Creating a corpus of parse-annotated questions. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th Annual Meeting of the Association for Computational Linguistics*, pages 497–504. Association for Computational Linguistics.

Kucera, H. and Francis, W. N. (1967). *Computational Analysis of Present-Day American English*. Brown University Press.

Marcus, M. P., Santorini, B., and Marcinkiewicz, M. A. (1993). Building a large annotated corpus of English: The Penn Treebank. *Computational Linguistics*, 19:313–330.

Martí, M. A., Taule, M., Màrquez, L., and Bertran, M. (2007). CESS-ECE: A multilingual and multilevel annotated corpus. Available for download from: http://www.lsi.upc.edu/∼mbertran/cess-ece/.

Martins, A. F., Das, D., Smith, N. A., and Xing, E. P. (2008). Stacking dependency parsers. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 157–166.

McCallum, A., Freitag, D., and Pereira, F. (2000). Maximum entropy markov models for information extraction and segmentation. In *Proceedings of the 17th International Conference on Machine Learning*, pages 591–598.

McDonald, R. and Nivre, J. (2007). Characterizing the errors of data-driven dependency parsing models. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 122–131.

Montemagni, S., Barsotti, F., Battista, M., Calzolari, N., Corazzari, O., Lenci, A., Zampolli, A., Fanciulli, F., Massetani, M., Raffaelli, R., Basili, R., Pazienza, M. T., Saracino, D., Zanzotto, F., Nana, N., Pianesi, F., and Delmonte, R. (2003). Building the Italian Syntactic-Semantic Treebank. In Abeillé, A., editor, *Treebanks: Building and Using Parsed Corpora*, pages 189–210. Kluwer.

Nivre, J. (2003). An efficient algorithm for projective dependency parsing. In *Proceedings of the 8th International Workshop on Parsing Technologies (IWPT)*, pages 149–160.

Nivre, J. (2004). Incrementality in deterministic dependency parsing. In *Proceedings of the Workshop on Incremental Parsing: Bringing Engineering and Cognition Together (ACL)*, pages 50–57.

Nivre, J. (2006). *Inductive Dependency Parsing*. Springer.

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

Nivre, J., Hall, J., Kübler, S., McDonald, R., Nilsson, J., Riedel, S., and Yuret, D. (2007). The CoNLL 2007 shared task on dependency parsing. In *Proceedings of the CoNLL Shared Task of EMNLP-CoNLL 2007*, pages 915–932.

Nivre, J., Hall, J., and Nilsson, J. (2004). Memory-based dependency parsing. In *Proceedings of the 8th Conference on Computational Natural Language Learning (CoNLL)*, pages 49–56.

Nivre, J. and McDonald, R. (2008). Integrating graph-based and transition-based dependency parsers. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 950–958.

Nivre, J. and Nilsson, J. (2005). Pseudo-projective dependency parsing. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 99–106.

Oflazer, K., Say, B., Hakkani-Tür, D. Z., and Tür, G. (2003). Building a Turkish treebank. In Abeillé, A., editor, *Treebanks: Building and Using Parsed Corpora*, pages 261–277. Kluwer.

Petrov, S. and McDonald, R. (2012). Overview of the 2012 shared task on parsing the web. In *Notes on the First Workshop on Syntatctic Analysis for Non-Canonical Language*.

Prokopidis, P., Desypri, E., Koutsombogera, M., Papageorgiou, H., and Piperidis, S. (2005). Theoretical and practical issues in the construction of a Greek dependency treebank. In *Proceedings of the 3rd Workshop on Treebanks and Linguistic Theories (TLT)*, pages 149–160.

Ross, S. and Bagnell, J. A. (2010). Efficient reductions for imitation learning. In *Proceedings of the 13th International Conference on Artificial Intelligence and Statistics*.

Ross, S., Gordon, G. J., and Bagnell, J. A. (2011). A reduction of imitation learning and structured prediction to no-regret online learning. In *Proceedings of the 14th International Conference on Artificial Intelligence and Statistics*.

Yamada, H. and Matsumoto, Y. (2003). Statistical dependency analysis with support vector machines. In *Proceedings of the 8th International Workshop on Parsing Technologies (IWPT)*, pages 195–206.

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

Zhang, Y. and Nivre, J. (2011). Transition-based dependency parsing with rich non-local features. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 188–193.

Statistical Mechanical Analysis of Semantic Orientations on Lexical Network

Takuma GOTO[*] *Yoshiyuki KABASHIMA*
Hiroya TAKAMURA
Tokyo Institute of Technology, 4259 Nagatsuta-cho, Midori-ku, Yokohama, Japan
takuma510@gmail.com, kaba@dis.titech.ac.jp, takamura@pi.titech.ac.jp

ABSTRACT
Many of the state-of-the-art methods for constructing a polarity lexicon rely on the propagation of polarity on the lexical network. In one of those methods, where the Ising spin model is employed as a probabilistic model, it is reported that the system exhibits the phase transition in the vicinity of the optimal temperature parameter. We provide an analysis of this phenomenon from the viewpoint of statistical mechanics and clarify the underlying mechanism. On the basis of this analysis, we propose a scheme for improving the extraction performance, i.e., by removing the largest eigenvalue component from the weight matrix. Experimental results show that the scheme significantly improves the accuracy of the extraction of the semantic orientations at negligible additional computational cost, outperforming the state-of-the-art algorithms. We also explore the origin of the high classification performance by analyzing eigenvalues of the weight matrix and a linearized model.

KEYWORDS: sentiment analysis, polarity lexicon, spin model, label propagation.

[*] Takuma Goto now works for ACCESS CO., LTD.

Proceedings of COLING 2012: Technical Papers, pages 977–994,
COLING 2012, Mumbai, December 2012.

1 Introduction

A huge amount of semantic information is constantly being produced and accumulated on the Internet by the activities of individuals through, for example, their blog, Twitter, and Facebook postings. The information tends to focus on personal interests but can include generally useful information such as opinions about fashion and comments about new products. This means that extracting and structuralizing such information can be beneficial for both producers and consumers, which led us to focus on the development of methods for handling semantic information.

In general, each word constituting sentences possesses its specific orientation. For example, we usually receive positive impressions for words such as "good", "excellent" and "enjoyable", while "bad", "poor" and "boring" sound negative. Such word-specific orientation of impression is termed *polarity* (or *semantic orientation*). A polarity lexicon is a list of words and phrases that are labeled by their polarity, and is an important resource in extracting semantic information from natural language data. Accordingly, the construction of such lists under various conditions has been a major focus in sentiment analysis research (Hatzivassiloglou and McKeown, 1997; Choi and Cardie, 2009). Many construction methods have been developed so far (Hatzivassiloglou and McKeown, 1997; Takamura et al., 2005; Turney and Littman, 2003; Velikovich et al., 2010; Kamps et al., 2004; Rao and Ravichandran, 2009).

Among those methods for polarity lexicon construction, the method proposed by Takamura et al. (2005) is distinctive in terms of emphasizing the utility of probabilities. In their method, the construction of a polarity lexicon is mapped to the Ising spin model of magnetism at a finite temperature. This mapping, in conjunction with the formalism of equilibrium statistical mechanics, yields a probabilistic model for assigning a polarity to each word. The optimal assignment of the polarities is determined by approximately evaluating the averages of the spin variables. Its experimental application to a lexical network of 88,015 words demonstrated its utility. Besides this, we would like to draw attention to the following observations:

- The classification performance is optimized at a finite temperature. This is somewhat counterintuitive, since the cost (energy) function is not minimized unless the temperature vanishes. At a finite temperature, the probability mass is split and given to all states, instead of a single one that minimizes the cost function. As a result, most of the spin averages are hardly biased. This suggests that performance can be optimized by using relatively small signals which are offered by the hardly biased spin averages.
- The experimental data indicates that the spin system exhibits a ferromagnetic phase transition in the vicinity of the optimal temperature.

The purpose of this paper is to propose a simple but effective scheme to improve the performance of the original spin-model-based method by clarifying the mechanism underlying the above observations. We show that the rate of correct word classification is significantly increased simply by removing the largest eigenvalue component from the weight matrix of the lexical network, which is directly related to avoidance of the ferromagnetic phase transition. The computational cost for the removal grows only linearly with network size, which means that the improved method is highly practical and useful. We also show that this scheme of removing the largest eigenvalue component also improves the linearized model, which is almost equivalent to the label propagation (Zhu and Ghahramani, 2002).

Although we focus on a specific problem of constructing the polarity lexicon in this paper, the developed methodology can be employed for general purposes of assessing influences of a few representative nodes in a network via local communications. In fact, network-based semi-supervised models are used in a number of tasks in natural language processing, such as word sense disambiguation (Yu et al., 2011), machine translation (Alexandrescu and Kirchhoff, 2009), query classification (Hu et al., 2009; Li et al., 2008).

2 Related Work

There has been much related work on building polarity lexicons.

One of the earliest studies was done by Hatzivassiloglou and McKeown (1997), who focused on conjoined adjectives in the Wall Street Journal corpus (Marcus et al., 1993). They deduced the polarity of adjectives by using pairs of adjectives appearing with a conjunction in the corpus. For example, pairs of adjectives joined with an "and" tend to have the same semantic orientation (e.g., "simple and well-received") while those joined with a "but" tend to have the opposite semantic orientation (e.g., "simplistic but well-received"). Because of the limited applicability of this method, only adjectives can be entried in a polarity lexicon.

Taking a corpus-based approach, Turney and Littman (2003) built a polarity lexicon by using two algorithms. They used a query such as "word NEAR good" and "word NEAR bad," where "good" and "bad" are seed words (their polarities are known), and obtained the number of hits returned by a search engine. The association strength of the target word with positive (negative) seed words was calculated. In their work, 3596 words from the General Inquirer lexicon (Stone et al., 1966) were used for empirical evaluation. General Inquirer is a list of words with polarity labels such as "Positiv" and "Negativ", which we also used for our evaluation.

Kamps et al. (2004) proposed a thesaurus-based method for adjectives that uses a network constructed by connecting each pair of synonymous words provided by WordNet (Fellbaum, 1998) in which the shortest paths to two seed words, "good" and "bad," are used to obtain the polarity of a word. This method is attractive in terms of computational cost, but the shortest paths are sensitive to local disturbances in the network topology.

Similarly, Velikovich et al. (2010) developed a method that aggregates a huge amount of unlabeled corpus data from the Web and constructs a lexical network. They used a graph propagation algorithm, in which the weighted shortest paths from seed words are used.

Kaji and Kitsuregawa (2007) proposed a method for constructing a Japanese polarity lexicon from Web data. They collected positive (negative) sentences from the Web using structural clues such as HTML tags and then extracted polar phrases from them.

The method proposed by Rao and Ravichandran (2009) increases robustness against network disturbances. It treats polarity detection as a semi-supervised label propagation problem in a graph in which higher order correlations of network topology other than the shortest paths are involved. They showed that the *label propagation* algorithm (Zhu and Ghahramani, 2002) leads to a significant improvement in the accuracy of polarity detection for WordNet-based networks compared to various known heuristics. The label propagation is used for word polarity extraction also in the recent literature (Speriosu et al., 2011; Brody and Diakopoulos, 2011).

The linearized model in Section 5.3 can also be interpreted as a graph kernel, which is used in natural language processing (e.g., Komachi et al. (2008)).

3 Ising spin model

3.1 Overview of Ising spin model

For later analysis, we briefly summarize the basic notation and techniques of Ising spin systems. In general, the Ising spin model is composed of N binary variables termed (*Ising*) *spins*: $\boldsymbol{S} = \left(S_1, S_2, \ldots, S_N\right)$, where $S_i \in \{+1, -1\}$ $(i = 1, 2, \ldots, N)$, for which energy function

$$E(\boldsymbol{S}, \beta) = -\beta \sum_{i>j} J_{ij} S_i S_j - \sum_{i=1}^{N} h_i S_i \tag{1}$$

is defined. Here, J_{ij} represents the efficacy of interactions between two spins, S_i and S_j, and h_i stands for the external field added to S_i, and β is called the inverse temperature. The most fundamental assumption of equilibrium statistical mechanics is that, when a system is characterized by energy function $E(\boldsymbol{S})$, the probability that microscopic state $\boldsymbol{S}$ is generated in equilibrium at temperature T $(= \beta^{-1} > 0)$ is provided by the Boltzmann-Gibbs distribution:

$$P(\boldsymbol{S}) = \frac{e^{-E(\boldsymbol{S}, \beta)}}{Z}, \tag{2}$$

where Z is the normalization factor. This equation assigns a larger probability to microscopic state $\boldsymbol{S}$ if it has a lower value of energy (Equation (1)). This tendency is more/less significant when β is set to a higher/lower value. Given this assumption, the main task of statistical mechanics is to evaluate the averages of various quantities using Equation (2), which is unfortunately computationally difficult.

A family of mean field approximations offers a practical solution for resolving this difficulty (Opper and Saad, 2001). The approximations are calculated on the basis of the Kullback-Leibler divergence between Equation (2) and test distribution $Q(\boldsymbol{S})$: $D(Q||P) = \sum_{\boldsymbol{S}} Q(\boldsymbol{S}) \log(Q(\boldsymbol{S})/P(\boldsymbol{S})) = \log(Z) + F[Q, \beta]$, where

$$F[Q, \beta] = \sum_{\boldsymbol{S}} Q(\boldsymbol{S}) E(\boldsymbol{S}, \beta) + \sum_{\boldsymbol{S}} Q(\boldsymbol{S}) \log Q(\boldsymbol{S}) \tag{3}$$

is termed the variational free energy. $D(Q||P)$ is generally non-negative and vanishes if and only if $Q(\cdot) = P(\cdot)$, which means that minimizing $F[Q, \beta]$ with respect to test distribution $Q(\cdot)$ leads to the correct evaluation of the true distribution, $P(\cdot)$.

A naïve approximation is derived by limiting the test distribution to one in factorizable form:

$$Q(\boldsymbol{S}) = \prod_i \frac{1 + m_i S_i}{2}. \tag{4}$$

Here, m_i denotes the mean of spin S_i with respect to the test distribution, which parameterizes the marginal distribution as $Q_i(S_i) = \sum_{\boldsymbol{S} \backslash S_i} Q(\boldsymbol{S}) = (1 + m_i S_i)/2$. Here, $A \backslash x$ generally denotes exclusion of x from set A. Plugging Equation (4) into Equation (3) and minimizing $F[Q, \beta]$ with respect to m_i $(i = 1, 2, \ldots, N)$ yield the *mean field equation*:

$$m_i = \tanh\left(\beta \sum_j J_{ij} m_j + h_i\right). \tag{5}$$

In many cases, Equation (5) can be solved by iterative substitution. Its computational cost is at most $O(N^2)$ per update, which is much lower than that required for the exact evaluation of the spin average, $O(2^N)$.

3.2 Analytical analysis

Although obtaining the exact solution is technically difficult, one can still handle Equation (5) analytically to a certain extent if $h_i = 0$ $(i = 1, 2, \ldots, N)$ is satisfied. For this, we set $h_i = 0$ $(i = 1, 2, \ldots, N)$ and use a Taylor series, $\tanh(x) = x - x^3/3 + \ldots$, on the right hand side of Equation (5), which yields

$$m_i = \sum_j \beta J_{ij} m_j - \frac{1}{3}\left(\sum_j \beta J_{ij} m_j\right)^3 + \ldots. \tag{6}$$

This guarantees that Equation (5) possesses the trivial solution[1] $m_i = 0$ $(i = 1, 2, \ldots, N)$. When T (β) is sufficiently high (low), this solution minimizes Equation (3) under the constraint of Equation (4) since the second term (the negative entropy part) of Equation (3) is dominant and provides an appropriate approximation. Let us decompose (symmetric) matrix $\boldsymbol{J} = (J_{ij})$ as $\boldsymbol{J} = \sum_{\mu=1}^{N} \lambda_\mu \boldsymbol{x}_\mu (\boldsymbol{x}_\mu)^{\mathrm{tr}}$, where $\lambda_1 \geq \lambda_2 \ldots \geq \lambda_N$, and $\boldsymbol{x}_\mu$ $(\mu = 1, 2, \ldots, N)$ are eigenvalues of $\boldsymbol{J}$ and the corresponding unit eigenvectors, respectively, and tr denotes the matrix transpose. As explained in Appendix A, since the eigenvalues of the Hessian of the mean field free energy (Equation (16)) are given as $\beta^{-1} - \lambda_\mu$ $(\mu = 1, 2, \cdots, N)$, the trivial solution becomes unstable in the direction of $\boldsymbol{x}_1$ when

$$1 - \beta\lambda_1 = 0, \tag{7}$$

breaking the stability condition that all eigenvalues of the Hessian are positive. When no solutions other than the trivial one coexist, which is experimentally confirmed in our system shown later, this signals the onset of a phase transition to a non-trivial solution. In particular, when $J_{ij} \geq 0$ holds for all spin pairs, which is the case for models of ferromagnetism, every element of the critical eigenvector $\boldsymbol{x}_1$ has an identical sign. As a consequence, the non-trivial solution is characterized by a non-vanishing value of *magnetization*:

$$m = \frac{1}{N}\sum_i m_i, \tag{8}$$

whose absolute value is kept non-negligible as N becomes large if and only if m_i $(i = 1, 2, \ldots, N)$ mostly have an identical sign being dominated by the components of $\boldsymbol{x}_1$. This is considered to correspond to the emergence of spontaneous magnetization in ferromagnetic materials, which is particularly referred to as the *ferromagnetic phase transition*.

More precisely, the analysis presented above means that, when the trivial solution is perturbed by weak fields $\boldsymbol{h} = (h_i)$ for $\beta < \lambda_1^{-1}$, the spin averages depend linearly on $\boldsymbol{h}$ (to the first order):

$$\boldsymbol{m} \simeq (\boldsymbol{I} - \beta\boldsymbol{J})^{-1}\boldsymbol{h}, \tag{9}$$

where $\boldsymbol{I}$ is the identity matrix. This can be derived from Equation (6) by ignoring the $O(\beta^3)$ terms, adding $\boldsymbol{h}$, and solving $\boldsymbol{m} \simeq \beta\boldsymbol{J}\boldsymbol{m} + \boldsymbol{h}$. An expression of eigenvalue decomposition, $(\boldsymbol{I} - \beta\boldsymbol{J})^{-1} = \sum_{\mu=1}^{N} (1 - \beta\lambda_\mu)^{-1} \boldsymbol{x}_\mu (\boldsymbol{x}_\mu)^{\mathrm{tr}}$, suggests that the phase transition signaled by Equation (7) is brought about by divergence of the sensitivity of the spin averages with respect to the addition of infinitesimal external fields that are proportional to $\boldsymbol{x}_1$.[2]

[1] Trivial solution is called *paramagnetic solution* in statistical mechanics.

[2] $\boldsymbol{x}_\mu (\boldsymbol{x}_\mu)^{\mathrm{tr}}$ works as an operator that projects a vector $\boldsymbol{h}$ to the direction of $\boldsymbol{x}_\mu$ when multiplied as $\boldsymbol{x}_\mu (\boldsymbol{x}_\mu)^{\mathrm{tr}} \boldsymbol{h}$.

4 Original Method

The original method (Takamura et al., 2005) is based on a lexical network constructed from three types of resources: a dictionary, a corpus, and a thesaurus. Although it is not easy to directly infer word polarities from those resources, it is relatively easy to obtain the tendency that two words have the same polarity as described in the next paragraph. One of the simplest probabilistic models that connect such tendency and the polarity assignment is the Ising spin model.

First, two words taken from the dictionary are linked if one of them appears in the gloss of the other. Each link is classified as either *same orientation SL* or *different orientation DL*. Next, synonyms, antonyms, and hypernyms taken from the thesaurus are connected by the link. Only antonym links are categorized as *DL*. Two adjectives are connected if they appear in a conjunctive form in the corpus (Hatzivassiloglou and McKeown, 1997).

After the links are provided, the weight of each link is set:

$$J_{ij} = \begin{cases} \frac{1}{\sqrt{d(i)d(j)}} & (l_{ij} \in SL) \\ -\frac{1}{\sqrt{d(i)d(j)}} & (l_{ij} \in DL), \\ 0 & \text{otherwise} \end{cases} \tag{10}$$

in order to introduce interactions into the spin model. Here, l_{ij} denotes the link between words i and j, and $d(i)$ denotes the degree of word i, i.e., the number of words linked directly to word i.

We assume that a small set of seed words, for which the correct polarities are known, is available. This assumption can be implemented into the spin model by imposing sufficiently large external fields on the spins of the initial word set:

$$E(\boldsymbol{S}, \beta) = -\beta \sum_{i>j} J_{ij} S_i S_j - \alpha \sum_{i \in L} a_i S_i, \tag{11}$$

where L denotes the initial word set (set of seed words), a_i denotes polarity (± 1) of a seed word i, and α denotes the strength of the external field of the seed words.

Substituting Equation (11) into Equation (2) yields the joint probability that all words simultaneously have semantic orientations labeled by $\boldsymbol{S} = (S_i)$ and $P(\boldsymbol{S})$. $P(\boldsymbol{S})$ can be used to evaluate marginal distribution $P(S_i) = \sum_{S \setminus S_i} P(\boldsymbol{S})$, which stands for the probability that word i has a polarity of S_i.

Given the marginal probabilities, the Bayesian framework guarantees that assigning a polarity $\sigma_i = \mathrm{argmax}_{S_i}\{P(S_i)\}$ to word i maximizes the rate of correct classification (Iba, 1999). In practice, this can be approximated as $\sigma_i = \mathrm{sign}(m_i) = m_i/|m_i|$ by solving the mean field equation (Equation (5)).

The performance of the method was evaluated for a lexical network constructed from WordNet (Fellbaum, 1998), the Wall Street Journal and Brown corpora of the Penn Treebank (Marcus et al., 1993), partially using TreeTagger (Schmid, 1994). The network was composed of 88,015 nodes (words) that were sparsely connected as they were characterized by a power-law-type degree distribution with an average of 18.94 (Figure 1). Only 5.63% of the weights were negative, which suggests that the spin system had a tendency to exhibit the

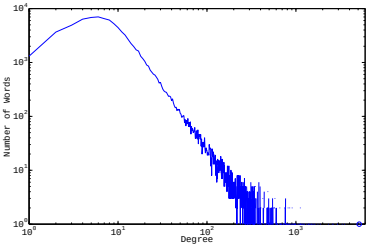


Figure 1: Degree distribution of constructed lexical network (log-log plot). Average degree was 18.94; maximum was 5144 (circle on the horizontal axis).

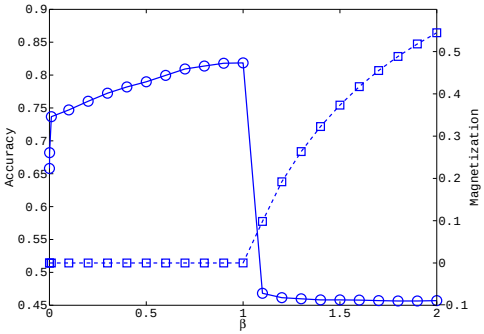


Figure 2: Temperature dependence of classification accuracy (circles) and magnetization (squares) for original method for 14 seed words.

ferromagnetic phase transition described above. The performance was assessed by using the General Inquirer labeled word list (Stone et al., 1966) as the gold standard. Of the 88,015 words in the network, 3596 were included in this list. Of these words, 1616 were positive, and 1980 were negative.

Testing was performed by varying β from 0.1 to 2.0 by 0.1 and setting α to 1000. The number of fixed seed words ranged from 2 to 14: {good, bad}, {good, superior, bad, inferior}, and {good, nice, excellent, positive, fortunate, correct, superior, bad, nasty, poor, negative, unfortunate, wrong, inferior}.

Figure 2 shows how the classification accuracy (rate of classifying words correctly) and magnetization depended on temperature for the case of 14 seed words. Classification accuracy was maximized at $\beta \simeq 1.0$, where non-zero magnetization appeared, signaling the ferromagnetic phase transition, but deteriorated drastically as soon as β was raised to $\beta > 1.0$. Similar behavior was observed for four and two seed words.

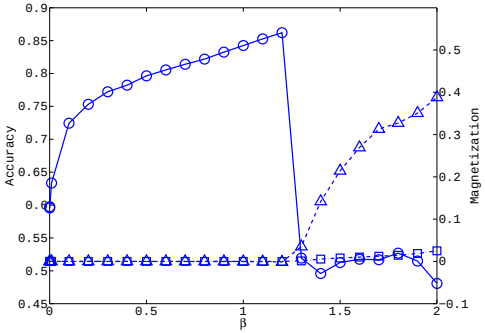


Figure 3: Classification accuracy and magnetization with improved method for 14 seed words. Circles ∘ denote classification accuracy. Squares □ and triangles △ denote magnetization evaluated using all words and only the 3596 labeled words, respectively.

5 Improved Method

5.1 Attention to Largest Eigenvalue

The setting and results described above naturally led to the following considerations.

1. Because external fields are added to only a small number of seed words, a spin system operating at high temperature (small β) can be handled as a perturbed state from the trivial solution.

2. When the temperature is lowered from a sufficiently high value, classification accuracy monotonically improves until non-vanishing magnetization appears. This suggests that the ferromagnetic phase transition is the main cause of the drastic performance deterioration.

The second consideration suggests that the classification accuracy can be improved by preventing the ferromagnetic phase transition. Equations (7) and (9), in conjunction with the first consideration, imply that the phase transition is caused by divergence of the sensitivity matrix, $(\boldsymbol{I}-\beta\boldsymbol{J})^{-1}=\sum_{\mu=1}^{N}(1-\beta\lambda_\mu)^{-1}\boldsymbol{x}_\mu(\boldsymbol{x}_\mu)^{\mathrm{tr}}$, in the direction of $\boldsymbol{x}_1$. This means that a possible way to prevent this transition is to simply expurgate the λ_1 component from weight matrix $\boldsymbol{J}=(J_{ij})$:

$$\boldsymbol{J}'=\boldsymbol{J}-\lambda_1\boldsymbol{x}_1(\boldsymbol{x}_1)^{\mathrm{tr}}. \tag{12}$$

Figure 3 shows the profiles of the classification accuracy and magnetization versus β for the modified weight (Equation (12)) for 14 seed words. Similar profiles were obtained for the other two cases. Note that solving the mean field equation for $\boldsymbol{J}'$ does not increase the computational cost significantly; $\boldsymbol{x}_1$ can be obtained using the power method, for which the computational cost is similar to that of solving the original mean field equation. The $\lambda_1\boldsymbol{x}_1(\boldsymbol{x}_1)^{\mathrm{tr}}$ term requires only $O(N)$ computations per update in the iterative substitution scheme.

Seeds	SP	LP	Original	Improved	Improved (II)
2	70.0	74.8(0.6)	75.2(0.8)	84.5(1.2)	84.4(1.2)
4	70.0	74.2(0.7)	74.4(0.6)	83.5(1.2)	83.7(1.1)
14	74.2	81.6(0.9)	81.9(1.0)	86.2(1.2)	86.2(1.2)

Table 1: Optimal classification accuracy (%) for 2, 4, 14 seed words, and the cross-validation setting. "SP" corresponds to the method based on shortest path. "LP" corresponds to label propagation. "Original" and "Improved" correspond to original method (Takamura et al., 2005) and one based on Equation (12). "Improved (II)" is same as "Improved" except that second largest eigenvalue component, λ_2, is also removed from Equation (12). Values in parentheses are β value at which accuracy was optimized.

As we speculated, the ferromagnetic phase transition was prevented, as shown by the magnetization for "all words" (squares) in Figure 3. As a consequence, classification accuracy was improved beyond $\beta \simeq 1.0$. Classification performance was evaluated on General Inquirer as in the previous work (e.g., (Turney and Littman, 2003)). Table 1 shows the optimal classification accuracy achieved for the original method (Original) and two improved methods, together with two existing state-of-the-art algorithms (SP and LP). SP is the method based on the shortest-path from seed words on the network (Velikovich et al., 2010). LP is the label propagation (Rao and Ravichandran, 2009). Note that, since we are interested in the impact made by the choice of algorithms, both of SP and LP were test on the lexical network that we used for our method, except that the edges with negative weights are removed because SP and LP cannot work properly with negative weights.[3] Also, although the label propagation by Rao and Ravichandran (2009) did not have the parameter β, we introduced it to LP for the purpose of fair comparison. Its value is optimized on the test set.

The performance of Original was improved in all cases by using the proposed scheme (Improved). All the differences were statistically significant in the sign test with significance level of 1%. The result also shows that the improved methods are significantly better than the shortest-path based method and the label propagation.[4] Note that the increase in accuracy compared with the values previously reported in some other papers (e.g., 82.8% (Turney and Littman, 2003) and 82.2% (Esuli and Sebastiani, 2005)) is substantial.

The results with a few seed words (e.g., 2, 4, and 14) are more important since semi-supervised methods should be applied to resource-scarce languages or new domains where creating a large amount of seed words is not very practical. However, in order to examine the behavior of our method when we have many seed words, we employed 10-fold cross validation (i.e., approximately 3,200 seed words). The accuracy of 91.5% (Original) was slightly improved to 91.8 (Improved) and 91.9 (Improved (II)).

5.2 Removal of more eigenvalue components

Figure 3 also shows that the performance still deteriorated at a higher β value (1.3). To clarify the reason for this, we also plotted "selected magnetization" (triangles), which was evaluated

[3] The label propagation is not guaranteed to converge in the presence of negative weights.

[4] Yu et al. (2011) used the shortest-path based method on an extended lexical network, and compared it with the label propagation on a non-extended lexical network, resulting in a better performance of the former method. The combination of their extended network with our algorithm would be a promising piece of future work.

using only the 3596 labeled words for checking a possibility that a certain phase transition relevant to only the labeled words brings about the deterioration. The plot indicates that the selected magnetization bifurcates to a finite value for $\beta \simeq 1.3$. This indicates that another phase transition occurred due to the second largest eigenvalue component, λ_2.

However, this indication is only partially correct. Figure 4 plots the profiles of the first and second eigenvectors. While the components of the first eigenvector (top) are evenly spread across almost all sites, those of the second one (bottom) are significantly localized at several sites. This "localization" feature is actually quite common for the eigenvectors of many other relatively large eigenvalues. Figure 5 plots the 30 largest eigenvalues and the inverse participation ratio (IPR), which is defined as

$$\mathrm{IPR} = \frac{\sum_i v_i^4}{(\sum_i v_i^2)^2}, \tag{13}$$

for a real vector, $\boldsymbol{v} = (v_i)$, of their eigenvectors (Biroli and Monasson, 1999). This quantity IPR takes a value between 0 and 1. In particular, as the dimensionality tends to infinity, it remains positive for localized vectors but vanishes for spread ones, so this quantity is widely used as a standard measure for characterizing the localization property of high dimensional vectors.

The plots in Figure 5 show that, although λ_1 is isolated, many other eigenvalues are distributed in a rather degenerated manner and are accompanied by localized eigenvectors. The localized eigenvectors have non-negligible values only for a few elements. Therefore, removal of one of such components does not provide significant effects for most spins. This suggests that the bifurcation of the selected magnetization is caused by not only the λ_2 component but also by many other components that are simultaneously excited at $\beta \simeq 1.3$. Accordingly, little improvement in the classification accuracy can be gained by removing the components of the second largest eigenvalue component from the weight matrix, as shown in the rightmost column of Table 1. The performance was not improved significantly by further removing the third and the fourth eigenvalue components.

Note that most of the edges in the current network have positive weights, i.e., very biased to the same sign. This is the reason why the first eigenvector is not localized and the first eigenvalue is much larger than the others. Also, the localization of eigenvectors with large eigenvalues, which is observed except for the first eigenvalue in the current network, often happens in *complex network*. The theoretical background can be found in the paper by Kabashima and Takahashi (2012). Therefore, the characteristics of the network is not completely accidental.

5.3 What is Essential?

The analysis presented so far indicates that high classification performance is achieved for a relatively small β, at which no magnetization appears unless external fields are imposed. Figure 6 plots the average of spins obtained using the improved method for $\beta = 1.2$ and 1.3 for 14 seed words. Most of the absolute values of the spin averages were rather small when a high classification accuracy was gained ($\beta = 1.2$, top) and became much larger after the performance deteriorated due to a phase transition ($\beta = 1.3$, bottom). When the spin averages have small absolute values, one can handle the system as a state produced by slightly perturbing the trivial solution by external fields imposed on only the seed word spins. This suggests a possibility that the linear response for the external fields plays a key role for the

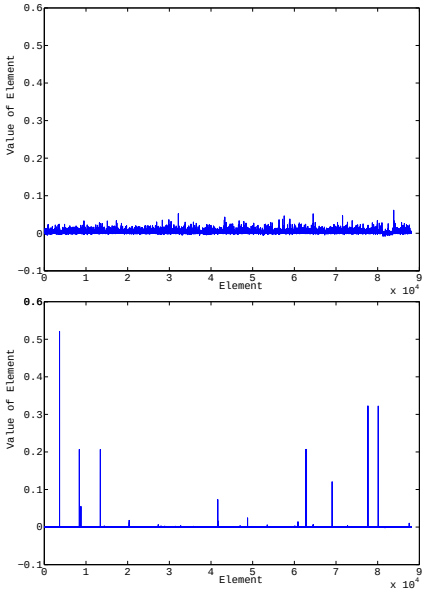


Figure 4: Eigenvectors of λ_1 (top) and λ_2 (bottom) of weight matrix $\boldsymbol{J}$.

Seeds	14	4	2
Accuracy	81.6(0.9)	74.2(0.7)	74.8(0.7)

Table 2: Optimal classification accuracy (%) of linearized model. Values in parentheses are β at which accuracy was optimized.

high classification accuracy. For checking this possibility, we examined the performance of a simplified model defined by the linear approximation of Equation (9):

$$\boldsymbol{m} = (\boldsymbol{I} - \beta \boldsymbol{J}')^{-1} \boldsymbol{h}^0, \tag{14}$$

where $\boldsymbol{h}^0 = (h_i^0)$ is provided as $h_i^0 = \alpha a_i$ if i is included in the set of seed words and vanishes otherwise. A power series expression, $(\boldsymbol{I} - \beta \boldsymbol{J}')^{-1} = \sum_{n=0}^{\infty} (\beta \boldsymbol{J}')^n$, indicates that this can be practically assessed by iterating the recursive equations:

$$\boldsymbol{m}^{t+1} = \boldsymbol{m}^t + \boldsymbol{u}^t \quad \text{and} \quad \boldsymbol{u}^{t+1} = (\beta \boldsymbol{J}') \boldsymbol{u}^t, \tag{15}$$

a sufficient number of times by setting the initial conditions as $\boldsymbol{m}^0 = \boldsymbol{0}$ and $\boldsymbol{u}^0 = \boldsymbol{h}^0$. This can be carried out with a computational cost similar to that of the naive mean field method.

Table 2 shows the optimal classification accuracy obtained with the linearized model. These results, in conjunction with those in Table 1, indicate that the performance of the linearized model was worse than that of the improved method, but similar to those of the original

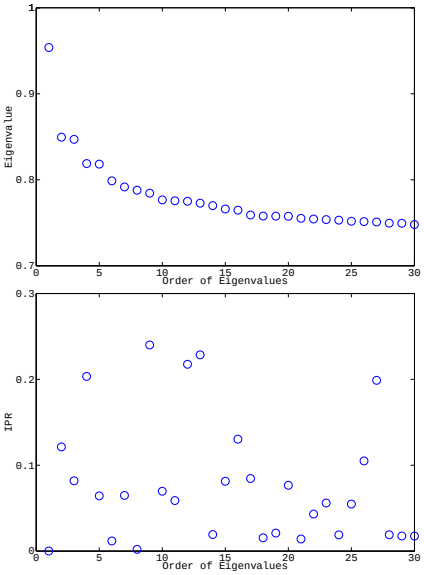


Figure 5: 30 largest eigenvalues of $\boldsymbol{J}$ (top) and inverse participation ratio (IPR) for corresponding eigenvectors (bottom). Larger IPR suggests that the corresponding eigenvector is more localized.

one (Takamura et al., 2005). This suggests that much of the information needed to correctly classify words is included in the linear response of the spin averages to the polarities of the seed words at high temperature.

As shown in Figure 5, the eigenvectors of large eigenvalues, which are emphasized at low temperature, are mostly localized and could contain relevant classification information only for a few words that corresponds to non-negligible values of elements. Therefore, they are individually insufficient for correctly classifying most other words corresponding to negligible elements. This means that, assigning polarities at high temperature so that information for all spin alignments is summed up with moderate probabilities is essential in the current spin-model-based method. This is achieved by assessing the linear response of the trivial solution to external fields representing the polarities of the seed words in the simplified scheme of Equation (14), and employment of the mean field equation (Equation (5)) offers a further gain under favor of the nonlinearity effect of tanh(·). This is in contrast to other approaches in which a single state that (approximately) optimizes a certain cost function is used for determining polarities (Blum and Chawla, 2001; Blum et al., 2004; Choi and Cardie, 2009).

A question that arises would be whether or not the removal of the largest eigenvalue component improves the linearized model. To answer this question, we conducted more experiments with the linearized model with the largest eigenvalue component being removed. The result

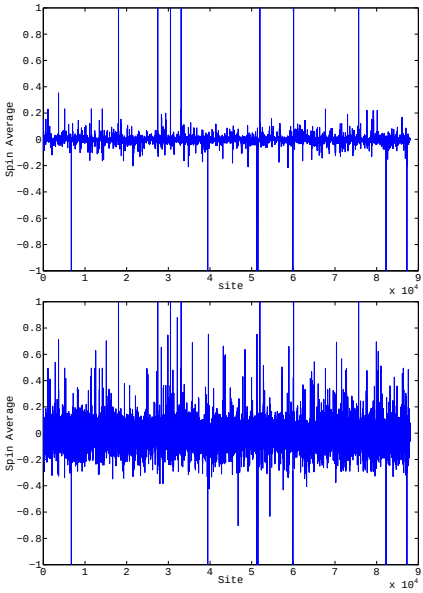


Figure 6: Spin averages m_i for $\beta = 1.2$ and 1.3 for 14 seed words. The mean of the absolute value of $|m_i|$ increased more than ten times, from 0.0028 ($\beta = 1.2$, top) to 0.0310 ($\beta = 1.3$, bottom).

is shown in Table 3. The comparison with the result in Table 2 suggests that the removal of the largest eigenvector also improves the linearized model. We note that the linearized model is almost equivalent to the label propagation (see the Taylor series in (Equation (6)); the difference is simply the presence of the normalization step. In fact, the result of the linearized model (Table 2) is almost the same as that of the label propagation (Table 1). The experimental result in Table 3 suggests that an idea similar to the removal of eigen components might also improve the label propagation, although we need to overcome the difficulty that the label propagation is not guaranteed to work properly if some edge weights are negative.

Conclusion

We have provided an analytical analysis of the behavior of a previously proposed spin-model-based method for constructing a polarity lexicon from the viewpoint of statistical mechanics.

Seeds	14	4	2
Accuracy	85.3(1.1)	83.8(1.1)	83.8(1.1)

Table 3: Optimal classification accuracy (%) of linearized model with the largest eigenvector being removed. Values in parentheses are β at which accuracy was optimized.

On the basis of this analysis, we proposed a scheme for improving the performance of polarity lexicon extraction, i.e., removing the largest eigenvalue component from the weight matrix of the lexical network, the result is quite significant. For example, classification accuracy was increased from 75.2 to 84.5% for the case of two seed words without significantly increasing computational cost. This scheme also improves the linearized model.

We also examined the possibility of improving the performance further by removing more eigenvalue components. However, the resulting degeneracy of the eigenvalues in the weight matrix, which is accompanied by eigenvector localization, minimizes the gain improvement. In addition, we investigated a linearized model to characterize the classification performance and found that the linear response to the polarities of the seed words at high temperature contains essential information. While many methods have been proposed for binary classification, apparently most of them are based on optimization of a certain cost function or on achievement of the low-temperature state of the Boltzmann-Gibbs distribution. In general, high-temperature states are technically easier to handle than low-temperature ones because a greater variety of perturbative techniques can be used. The utility of the (linear) response in the high-temperature state shown here offers a novel promising approach to generic classification when labels are provided for a small fraction of representative instances.

The developed methodology can be employed for general purposes of assessing influences of a few representative nodes in a network via local communications. The Ising spin model or similar models including its linearized model are used in a number of tasks in natural language processing.

Future work includes more use of language data and development of applications using this polarity lexicon construction method as well as use of other approximation schemes such as advanced Markov chain Monte Carlo methods in which equilibration is significantly accelerated by using extended ensembles (Iba, 1999).

Acklowdedgments

This work was partially supported by JSPS KAKENHI No. 22300003 and Mitsubishi Foundation (YK).

A Stability of Trivial Solution

Inserting Equation (4) into Equation (3) and setting h_i to zero ($i = 1, 2, \ldots, N$) yields an expression of the *mean field free energy*:

$$F_{\mathrm{MF}}(\boldsymbol{m}) \quad = \quad -\beta \sum_{i>j} J_{ij} m_i m_j + \sum_{i=1}^{N} \sum_{S_i=\pm 1} \frac{(1+m_i S_i)}{2} \log \frac{(1+m_i S_i)}{2}. \tag{16}$$

To examine the local stability of the paramagnetic solution, $m_i = 0$ ($i = 1, 2, \ldots, N$), we evaluated the Hessian of $F_{\mathrm{MF}}(\boldsymbol{m})$:

$$\boldsymbol{H} = \left(\left. \frac{\partial^2 F_{\mathrm{MF}}(\boldsymbol{m})}{\partial m_i \partial m_j} \right|_{\boldsymbol{m}=\boldsymbol{0}} \right) = -\beta \boldsymbol{J} + \boldsymbol{I}. \tag{17}$$

The solution is locally stable if and only if $\boldsymbol{H}$ has no negative eigenvalues. Equation (17) indicates that the eigenvalues of $\boldsymbol{H}$ are given as $\beta^{-1} - \lambda_\mu$ ($\mu = 1, 2, \ldots, N$) using the eigenvalues of $\boldsymbol{J}$, $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_N$. Thus, as β increases from a very low value, the stability condition that all eigenvalues are positive is broken when $\beta^{-1} - \lambda_1 = 0$ holds, i.e., Equation (7).

References

Alexandrescu, A. and Kirchhoff, K. (2009). Graph-based learning for statistical machine translation. In *Proceedings of Human Language Technologies and the Annual Conference of the North American Chapter of the Association for Computational Linguistics (HLT-NAACL2009*, NAACL '09, pages 119–127, Stroudsburg, PA, USA. Association for Computational Linguistics.

Biroli, G. and Monasson, R. (1999). A single defect approximation for localized states on random lattices. *Journal of Physics A: Mathematical and General*, 32(24): pages L255–L261.

Blum, A. and Chawla, S. (2001). Learning from labeled and unlabeled data using graph mincuts. In Brodley, C. and Danyluk, A., editors, *Proceedings of 18th International Conference on Machine Learning (ICML-2001)*, pages 19–26, Williams College, US. Morgan Kaufmann Publishers, San Francisco, US.

Blum, A., Lafferty, J., Rwebangira, M. R., and Reddy, R. (2004). Semi-supervised learning using randomized mincuts. In *Proceedings of the 21st international conference on Machine learning (ICML '04)*, pages 97–104, New York, NY, USA. ACM.

Brody, S. and Diakopoulos, N. (2011). Cooooooooooooooollllllllllll!!!!!!!!!!!!!! using word lengthening to detect sentiment in microblogs. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP'11)*, pages 562–570, Edinburgh, Scotland, UK. Association for Computational Linguistics.

Choi, Y. and Cardie, C. (2009). Adapting a polarity lexicon using integer linear programming for dom ain-specific sentiment classification. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 590–598.

Esuli, A. and Sebastiani, F. (2005). Determining the semantic orientation of terms through gloss analysis. In *Proceedings of the 14th ACM International Conference on Information and Knowledge Management (CIKM-05)*, pages 617–624.

Fellbaum, C. (1998). *WordNet: An Electronic Lexical Database, Language, Speech, and Communication Series*. MIT Press.

Hatzivassiloglou, V. and McKeown, K. R. (1997). Predicting the semantic orientation of adjectives. In *Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics and the 8th Conference of the European Chapter of the Association for Computational Linguistics*, pages 174–181.

Hu, J., Wang, G., Lochovsky, F., tao Sun, J., and Chen, Z. (2009). Understanding user's query intent with wikipedia. In *Proceedings of the 18th international conference on World wide web*, WWW '09, pages 471–480, New York, NY, USA. ACM.

Iba, Y. (1999). The Nishimori line and Bayesian statistics. *Journal of Physics A: Mathematical and General*, 32(21): pages 3875–3888.

Kabashima, Y. and Takahashi, H. (2012). First eigenvalue/eigenvector in sparse random symmetric matrices: influences of degree fluctuation. *Journal of Physics A: Mathematical and Theoretical*, 45(32): 325001 (19pp).

Kaji, N. and Kitsuregawa, M. (2007). Building lexicon for sentiment analysis from massive collection of html documents. In *Proceedings of the Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 1075–1083.

Kamps, J., Marx, M., Mokken, R. J., and de Rijke, M. (2004). Using wordnet to measure semantic orientation of adjectives. In *Proceedings of the 4th International Conference on Language Resources and Evaluation (LREC 2004), volume IV*, pages 1115–1118.

Komachi, M., Kudo, T., Shimbo, M., and Matsumoto, Y. (2008). Graph-based analysis of semantic drift in espresso-like bootstrapping algorithms. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 1011–1020.

Li, X., Wang, Y.-Y., and Acero, A. (2008). Learning query intent from regularized click graphs. In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '08, pages 339–346, New York, NY, USA. ACM.

Marcus, M. P., Santorini, B., and Marcinkiewicz, M. A. (1993). Building a large annotated corpus of english: The Penn Treebank. *Computational Linguistics*, 19(2):313–330.

Opper, M. and Saad, D. (2001). *Advanced Mean Field Methods: Theory and Practice*. MIT Press.

Rao, D. and Ravichandran, D. (2009). Semi-supervised polarity lexicon induction. In *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics (EACL'09)*, pages 685–682.

Schmid, H. (1994). Probabilistic part-of-speech tagging using decision trees. In *Proceedings of International Conference on New Methods in Language Processing*, pages 44–49.

Speriosu, M., Sudan, N., Upadhyay, S., and Baldridge, J. (2011). Twitter polarity classification with label propagation over lexical links and the follower graph. In *Proceedings of the EMNLP 2011 Workshop on Unsupervised Learning in NLP*, pages 53–63.

Stone, P. J., Dunphy, D. C., Smith, M. S., and Ogilvie, D. M. (1966). *The General Inquirer: A Computer Approach to Content Analysis*. The MIT Press.

Takamura, H., Inui, T., and Okumura, M. (2005). Extracting semantic orientations of words using spin model. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 133–140.

Turney, P. D. and Littman, M. L. (2003). Measuring praise and criticism: Inference of semantic orientation from association. *ACM Transactions on Information Systems*, 21(4):315–346.

Velikovich, L., Blair-Goldensohn, S., Hannan, K., and McDonald, R. (2010). The viability of web-derived polarity lexicons. In *Proceedings of Human Language Technology and the Annual Conference of the North American Chapter of the Association for Computational Linguistic (HLT-NAACL 2010)*, pages 777–785.

Yu, M., Wang, S., Zhu, C., and Zhao, T. (2011). Semi-supervised learning for word sense disambiguation using parallel corpora. In *Proceedings of the 8th International Conference on Fuzzy Systems and Knowledge Discovery (FSKD)*, pages 1490–1494.

Zhu, X. and Ghahramani, Z. (2002). Learning from labeled and unlabeled data with label propagation. Technical Report CMU-CALD-02-107, Carnegie Mellon University.

Finding Thoughtful Comments from Social Media

Swapna Gottipati, *Jing Jiang*
School of Information Systems, Singapore Management University, Singapore
swapnag.2010@smu.edu.sg, jingjiang@smu.edu.sg

ABSTRACT
Online user comments contain valuable user opinions. Comments vary greatly in quality and detecting high quality comments is a subtask of opinion mining and summarization research. Finding attentive comments that provide some reasoning is highly valuable in understanding the user's opinion particularly in sociopolitical opinion mining and aids policy makers, social organizations or government sectors in decision making. In this paper we study the problem of detecting thoughtful comments. We empirically study various textual features, discourse relations and relevance features to predict thoughtful comments. We use logistic regression model and test on the datasets related to sociopolitical content. We found that the most useful features include the discourse relations and relevance features along with basic textual features to predict the comment quality in terms of thoughtfulness. In our experiments on two different datasets, we could achieve a prediction score of 79.37% and 73.47% in terms of F-measure on the two data sets, respectively.

KEYWORDS: Opinion mining, Information Extraction, Text Classification.

Proceedings of COLING 2012: Technical Papers, pages 995–1010,
COLING 2012, Mumbai, December 2012.

1 Introduction

In recent years sentiment analysis and opinion mining has been extensively studied in natural language processing (Pang and Lee, 2008), largely because of the availability of a huge amount of opinionated text in online product reviews, blogs, social networking sites, forums, etc. Most work on opinion mining is about mining reviews of products and services, and the focus of these studies has been on a few important subtasks such as sentiment classification (Pang et al., 2002; Cui et al., 2006; Jiang et al., 2011) and opinion extraction (Popescu and Etzioni, 2005; Choi et al., 2006; Wu et al., 2009).

When we go beyond product review mining and consider the general problem of opinion mining from social media, many other subtasks and challenges arise. One of them is how to assess the quality of online comments and select high quality ones for further analysis and summarization. Consider the problem of mining the comments found in online social media towards a political speech such as Obama's State of the Union address. By restricting the search space to politically active blogs and forums and by using queries such as "*State of the Union*," likely we are able to retrieve highly relevant comments to the speech. However, not every comment contains valuable insight into the public's opinions regarding the sociopolitical issues addressed in the speech. Comments such as "*innovate and innovation appeared 10 times*" and "*To him..investment = more deficit spending*" are subjective but lack thoughtful explanations to support their claims. In comparison, comments like "*You want to really drive innovation, job growth and entrepreneurs? Make education, health care and retirement less of a burden on the average family, adopt more socialist policies like Norway (paid for by higher taxes, especially on the rich), and watch our standard of living rise at last!*" provide much more insightful reasoning that government policy makers may find highly valuable in understanding the general public's sentiment. Hence, we define thoughtfulness as insightful reasoning with relevance to the issues discussed in the article. So, detection of the comments with reasoning or justification is the focus of our task. Thoughtfulness is accessed only for relevant comments. This problem of finding *thoughtful* comments from social media is what we study in this paper. Formally, a *thoughtful comment* is relevant to the target document and has a justification or an argument to the issue(s) in the target document. It is particularly important for sociopolitical opinion mining because of the complexity of sociopolitical issues.

Intuitively, finding thoughtful comments is related to measuring text quality. There has been a large body of previous work on text quality prediction, but the methods are usually applied to student essays (Attali and Burstein, 2006) and news articles (Tang et al., 2003), (TREC novelty track 2003 and 2004). In social media mining, there have also been a number of studies on finding high quality reviews (e.g. (Kim et al., 2006; Agichtein et al., 2008)), but the focus in not on finding thoughtful comments, which requires us to look for reasoning in text. Presumably, a thoughtful comment should be logically well organized and coherent. We therefore hypothesize that discourse relations such as comparison, expansion and contingency will play an important role in finding thoughtful comments. A well organized comment is not always thoughtful. Comments such as, "*He is a great speaker as he writes the speech by himself and also delivers it very confidently*" are justified but are not relevant to the issues discussed in the article. Hence we hypothesize that relevance factors play an important role in detecting thoughtful comments.

We adopt a supervised learning approach and consider a diverse set of factors ranging from lexical usage to discourse relations, all derived from the textual content of comments. Many

of the factors we consider are based on the study by (Pitler and Nenkova, 2008). In addition, we also consider a relevance feature because of the nature of our problem. We construct two data sets to evaluate the various factors, one based on Singapore Prime Minister's National Day Rally speech[1] and the other on US President's State of the Union address[2].

Empirical evaluation reveals that discourse relations and relevance scores together with the standard textual features aid in better prediction of thoughtful comments. We could achieve a prediction score of 79.37% and 73.47% in terms of F-measure on the two data sets, respectively. We further tested our model across data collections. Our test result shows that the model with combined textual, discourse and relevance features still performs better than textual features alone.

The rest of the paper is organized as follows. We present the related work in Section 2. In Section 3, we formally define our problem and give an overview of our solution. We present the various features we consider in Section 4. The data set details are presented in Section 6. Evaluation and results are presented in Section 7. Finally we conclude and discuss some future work.

2 Related work

Our work is related to a large body of literature on measuring text quality in NLP, but our problem has some essential differences. The main difference is that in traditional sense, high-quality text should be grammatical, coherent and readable. For online comments, we focus more on the insightfulness or thoughtfulness of comments.

Many recent studies examined the challenges on the quality of comments. (Kim et al., 2006) studied how to predict the helpfulness of product reviews. They found that a helpful review should describe the features of the products and the pros/cons of the features. A more elaborative review that provides the complete details of the product is more likely to be considered high quality. Our problem is more general and the comments are not necessarily about products. Moreover, comments on sociopolitical articles need not elaborate on all the issues in the article. Therefore products and their features are not relevant to our problem. Another study by (Ghose and Ipeirotis, 2011) on review helpfulness looked into factors related to the reviewer, such as reviewer characteristics and reviewer history. In our work, we focus on features observed from the text only. Other social factors such as a commenter's profile or past behavior are complementary to our method.

Our work is also related to opinion retrieval (Zhang and Ye, 2008; Huang and Croft, 2009; Macdonald et al., 2009), which aims at automatically finding attitudes or opinions about specific targets, such as named entities, consumer products or public events. In most existing work on opinion retrieval, only relevance and subjectivity are considered, whereas we propose that quality in terms of thoughtfulness is also an important factor.

Work on measuring quality of social media content considers not only the quality of the content itself but also its authority in the social network through the author's authority, its popularity, etc. (Hsu et al., 2009). Several researchers explored the social network together with the content of the reviews to predict the review quality. (Bian et al., 2009) proposed a mutual reinforcement learning framework to simultaneously predict content quality and user reputation, whereas (Lu et al., 2010) proposed a linear regression model with various social contexts

[1] http://www.pmo.gov.sg/
[2] http://www.whitehouse.gov/

for review quality prediction. They combined textual and social context information to evaluate the quality of individual reviewers and to assess the quality of the reviews. We do not consider these factors as we want to focus on textual cues first. These additional features can be factored in as an independent step. Similar line of work can be seen by (Chen et al., 2011; Liu et al., 2007; Bian et al., 2009).

Our work is similar to (Amgoud et al., 2011) where they introduced argument analysis together with opinion. In their task, properties of a person or product (honesty, rigor, friendliness, etc.) are treated as arguments. The task is oriented towards aggregating features related to the product and supporting arguments to detect polarity. The task we address in this paper is quite different from their work in two main aspects. Firstly, for sociopolitical issues, the policy makers look for insightful reasoning text to understand the public sentiment in which case, properties are insufficient. Secondly, we study the attentiveness of the comments but not the polarity. Polarity orientation is a separate task which can be studied individually.

3 Problem Definition and Overview of Solution

We assume the following general definition of the task of finding thoughtful comments: Given a comment c made with respect to a *target* document d, we would like to determine whether c is a thoughtful comment. We will explain in Section 6 how we instruct the human annotators to label thoughtful comments. Generally speaking, a thoughtful comment is relevant to the target document and has a justification or an argument to the issue(s) in the target document.

While the task defined above is certainly not trivial, and theoretically speaking one would need a deep understanding of both the target document and the comment as well as relevant world knowledge to be able to judge whether a comment is thoughtful. Here we take an empirical approach and test whether features defined at lexical, syntactic, discourse levels and relevance factors have correlations with the thoughtfulness of comments and whether they can be used to achieve decent prediction accuracy. A large portion of the linguistic features we consider are inspired by existing work on measuring text quality. Indeed, at first glance our problem may appear to be the same as measuring text quality. News articles and student essays are formal and usually lengthier, whereas online comments are usually much shorter and less formal. In traditional sense, high-quality text should be grammatical, coherent and readable. Our problem seems to be text quality assessment which is defined as above, but we are not looking for grammar and readability. Instead we look for insightful reasoning with relevance to the article, as mostly user comments are not formal in social media.

We adopt a supervised learning approach to our problem. Specifically, we assume that we have a set of N training examples $\{(d_i, c_i, y_i)\}_{i=1}^{N}$, where d_i is a target document, c_i is a comment on d_i, and y_i is a binary label indicating whether c_i is a thoughtful comment with respect to d_i. With a set of feature functions, we can represent (d_i, c_i) by a feature vector $\mathbf{x}(d_i, c_i)$ (which we refer to as $\mathbf{x}_i$). We can then use standard classification algorithms to learn a classifier from $\{(\mathbf{x}_i, y_i)\}_{i=1}^{N}$. This classifier can be used to predict y for any unseen pair of d and c. In the following sections, we will explain in detail the features we consider and the classification algorithm we use.

4 Features

There have been many studies on measuring text quality and many features have been proposed to capture text quality. As mentioned previously our methodology is based on existing work on this topic. In particular, we follow the work by (Pitler and Nenkova, 2008). They

conducted a systematic study on text quality using various linguistic features and Wall Street Journal articles. Based on the major findings of their study, we take the following features as our starting point.

4.1 Structural Feature

Structural features are generated from the comment structure. (Pitler and Nenkova, 2008) tested various structural features including the average number of characters per word, the average number of words per sentence, the maximum number of words per sentence, and article length. According to their findings, article length was the only significant factor with good correlation with text quality. Hence, we define our first feature F_1 as the number of words in the comment.

4.2 Lexical Feature

Lexical features aim to capture the lexical usage of a piece of text compared to some reference corpus. (Pitler and Nenkova, 2008) used a lexical feature based on unigram language models, which provide a principled way to statistically model text. Specifically, it is assumed that there is a reference corpus that represents high quality text, e.g. a corpus of Wall Street Journal articles. A unigram language model, denoted as θ_r, can be estimated from this reference corpus. The lexical feature is defined as the log likelihood of the comment based on θ_r, calculated as:

$$\sum_w n(w,c) \log P(w|\theta_r), \tag{1}$$

where $P(w|\theta_r)$ is the probability of word type w according to θ_r, and $n(w,c)$ is the number of times word type w appears in comment c. We call this feature F_2.

4.3 Syntactic Feature

(Pitler and Nenkova, 2008) examined various syntactic features including the average parse tree height, the average number of noun phrases per sentence, the average number of verb phrases per sentence and the average number of subordinate clauses per sentence. They found that the average number of verb phrases per sentence was a useful feature with high correlation with text quality. So, the third feature F_3 we use for our study is the average number of verbs per comment.

We also experimented with other syntactic features like average number of noun phrases and noun to verb ratio calculated from the user's comments. We found that the average number of verbs per comment had the highest correlation with comment quality, and therefore we do not consider these other syntactic features in our experiments.

4.4 Discourse Features

Previous study by (Pitler and Nenkova, 2008) found that discourse relations were also correlated with text quality. Discourse relations aim to capture textual structures such as comparison, elaboration, cause-effect explanations and examples. They are considered key for the ability to properly interpret or produce discourse. For the problem of finding thoughtful comments, we hypothesize that discourse relations may play an even larger role because a logical argument will likely rely on coherently connecting textual units through discourse relations.

Discourse relations are divided into four major semantic classes (Prasad et al., 2008):

Expansion covers those relations where the second argument expands the discourse of the first argument or move its narrative forward.

Comparison relations highlight prominent differences between the two arguments of a relation.

Contingency is marked when one of the situations described in an argument causally influences the other argument.

Temporal relations are marked when the situations described in the arguments are related temporally, either synchronously or sequentially.

It has been found that a large portion of discourse relations can be detected through connectives, i.e. cue words and phrases (Pitler et al., 2008). We use a list of such connectives compiled by (Prasad et al., 2008) and study the statistics of our corpus to discover the discourse relations. Table 1 shows that the statistics of discourse relations in our dataset.

DR Class	Singapore	US
COMPARISON	44.50%	45.87%
EXPANSION	16.75%	15.47%
CONTINGENCY	38.75%	38.66%
TEMPORAL	5.70%	5.72%

Table 1: Discourse relations statistics in our corpus

Table 1 shows that the frequency of temporal relations is low in our corpus. It is not surprising because for many online comments the arguments are not temporal. Hence, we ignore the temporal class for the rest of our paper, and restrict our attention to only the other three major classes, namely, expansion, comparison and contingency. At the same time, many relations are explicit and can be discovered using the connectives/words as used in other applications (Saito et al., 2006).

The full list of the phrases for each class are shown in Table 2. This list is collated from (Prasad et al., 2008). We observed that some words are ambiguous: 'if', 'and', 'but', 'as' etc. In our study, such words are counted only once while combining the classes for the feature generation.

In the earlier work by (Pitler and Nenkova, 2008), the Penn Discourse Treebank was used for computing the discourse features. For us, we take a simpler approach and count the number of discourse relations in a comment. This becomes the F_4 in our experiments.

4.5 Relevance Feature

One of the important differences between our problem and standard text quality assessment is that the quality of a comment also relies on its relevance to the target of the comment. In our problem definition, the target is also a piece of text. For example, consider comments made to Obama's State of the Union speech. A comment such as "We are very lucky to live in the USA. I always have and always will support our president" is not directly related to any issue addressed by Obama in his speech, and therefore is not considered to be a thoughtful comment. Hence, for thoughtful comment prediction, we also consider a relevance feature

Class	Phrases
COMPARISON	although, as though, but, by comparison, even if, even though, however, nevertheless, on the other hand, still, then, though, while, yet, and, meanwhile, in turn, next, ultimately, meantime, also, as if, even as, even still, even then, regardless, when, by contrast, conversely, if, in contrast, instead, nor, or, rather, whereas, while, yet, even after, by contrast, nevertheless, besides, much as, as much as, whereas, neither, nonetheless, even when, on the one hand indeed, finally, in fact, separately, in the end, on the contrary, while
EXPANSION	accordingly, additionally, after, also, although, and, as, as it, as if besides, but, by comparison, finally, first, for example, for one thing, however, in addition, in fact, in other words, in particular, in response, in sum, in the end, in turn, incidentally, indeed, instead, likewise, meanwhile, nevertheless, on the one hand, on the whole, overall, plus, separately, much as, whereas, ultimately, as though, rather, at the same time, or, then, if, in turn, furthermore, in short, turns out, while, yet, that is, so, what's more as a matter of fact, further, in return, moreover, similarly, specifically,
CONTINGENCY	and, when, typically, as long as, especially if, even if, even when, if, so, when if only, lest,once, only if, only when, particularly if, at least partly because, especially as, especially because, especially since, in large part because, just because, largely because, merely because, not because, not only because, particularly as, particularly because, particularly since, partly because, because, simply because, since, then, after, one day after, reportedly after, consequently, mainly because, for, thus, apparently, in the end, in turn, primarily because, largely as a result, as, because, therefore, only because, particularly, when, so that, thereby, presumably, hence, as a result, if and when, unless, until, in part because, now that, perhaps because, only after, accordingly,

Table 2: Discourse relations

in addition to text quality features. There are many ways to measure relevance, and here we choose KL-divergence score, a principled measure for relevance commonly used in information retrieval tasks.

The KL-divergence score between a comment c and a target document d is defined as the KL-divergence between the unigram language models θ_c and θ_d estimated from c and d, respectively:

$$Div(\theta_c||\theta_d) = \sum_{w \in \mathcal{V}} p(w|\theta_c) \log \frac{p(w|\theta_c)}{p(w|\theta_d)}, \tag{2}$$

where $\mathcal{V}$ is the vocabulary.

KL-divergence using only nouns: We hypothesize that the topical relevance between a comment and its target relies more on the overlap of nouns in the two pieces of text. Hence, we consider another KL-divergence measure using only nouns in c and d. Specifically, we use the unigram language models that are defined over nouns only. Let θ_c^N and θ_d^N denote the two language models. We have

$$Div(\theta_c^N||\theta_d^N) = \sum_{w \in \mathcal{V}} p(w|\theta_c^N) \log \frac{p(w|\theta_c^N)}{p(w|\theta_d^N)}. \tag{3}$$

We define the fifth feature that uses only nouns, F_5=-$Div(\theta_c^N||\theta_d^N)$ and the sixth feature which is based on all words, F_6=-$Div(\theta_c||\theta_d)$.

KL-Divergence between comment and average comment: (Lu et al., 2010) proposed conformity features in which the comment c is compared with other comments by looking at the KL-divergence between the unigram model of the comment c, and unigram model of an "average" comment that contains the text of all comments for an article. We did a preliminary analysis to study the impact of conformity on the quality. We found that this KL-divergence score has low correlation with comment quality on our data sets, and therefore we do not consider it in the rest of this paper.

5 Logistic Regression

So far we have introduced six features, which are summarized in Table 3.

Feature Set	Description
F_1	Comment length
F_2	Comment likelihood
F_3	Average number of verbs
F_4	Number of discourse relations
F_5	Relevance score using nouns only
F_6	Relevance score using all words

Table 3: Full feature set for comment representation

Once features are defined, we can use a classification algorithm to learn a model from the training data and apply the model to unseen data for thoughtful comment prediction. In this paper we use logistic regression as our classification algorithm.

As we have pointed out earlier, a comment c together with its target document d can be represented by a feature vector $\mathbf{x}$. A logistic regression classifier models the probability of observing a discrete label y for a given $\mathbf{x}$ as follows:

$$p(y|\mathbf{x};\mathbf{w}) = \frac{1}{Z(\mathbf{x},\mathbf{w})}\exp(\mathbf{w}_y^T\mathbf{x}), \tag{4}$$

where

$$Z(\mathbf{x},\mathbf{w}) = \sum_{y\in\mathscr{Y}}\exp(\mathbf{w}_y^T\mathbf{x}).$$

Here $\mathbf{w}$ is a weight matrix and $\mathbf{w}_y$ is the weight vector corresponding to class y, and $\mathscr{Y}$ is the set of class labels.

Given training data $\{\mathbf{x}_i, y_i\}_{i=1}^N$, we learn a weight matrix by minimizing the following objective function:

$$\hat{\mathbf{w}} = \arg\min_{\mathbf{w}}\left[\lambda\|\mathbf{w}\|^2 - \frac{1}{N}\sum_{i=1}^{N}\log p(y_i|\mathbf{x}_i;\mathbf{w})\right], \tag{5}$$

where $\|\mathbf{w}\|^2 = \sum_{y\in\mathscr{Y}}\|\mathbf{w}_y\|^2$ and λ is a regularization parameter that is empirically set.

6 Data Set

6.1 Data Collection

Our objective is to study how the thoughtfulness of a comment is reflected in the various linguistic factors including discourse relations and relevance factors to the article. As we have mentioned earlier, measuring the thoughtfulness of a comment is especially important for sociopolitical opinion mining. We therefore collected two data sets in this domain for our evaluation. We first acquired the following two political speeches: (1) Singapore Prime Minister's National Day Rally Speech in 2010. (2) US President's State of the Union address in 2011. We further broke down each speech into several segments based on topical boundaries. The topics of each speech are listed in Table 4.

Speech	**Topics**
Singapore	Economy & Productivity, Immigration, Congestion, Housing, Education, National Service (NS), Singapore Spirit, Founding Fathers, Youth Olympics
US	Economy, Innovation & Research, Education, Rebuild, Spending & Taxes, Debts, Military

Table 4: Topics in the two speeches.

To collect an unbiased sample of comments for each speech, we use two search queries ("national day rally speech 2010" and "president state union address 2011") and Google API to obtain a list of top 50 URLs. We further manually selected URLs from online forums and blogs. We cleaned the data and removed short comments with no more than two words.

For F_2, the lexical feature, we need a suitable reference corpus. For the Singapore data set, we collected 1200 news articles from AisaOne.com to form our reference corpus. For the US data set, we used a set of 1358 New York Times articles to form the reference corpus.

6.2 Annotation

We engaged two human annotators to judge the comments we had collected. The annotators were asked to judge (1) whether a comment was relevant to each segment of its corresponding speech, and if so, (2) whether the comment was a thoughtful one. In other words, we treat each segment of a speech as a target document. For each pair of a comment c and a target document (i.e., a speech segment) d, we obtained two binary labels: a label z that indicates whether c is relevant to d, and a label y that indicates whether c is a thoughtful comment with respect to d.

To judge whether a comment was thoughtful, the annotators were asked to use the following criteria:

1. Is the comment a mere repetition or a rephrase of the speech text? For example, "PM says that we should stay open for the foreigners" is a repetition of the text from the article in passive voice. Such comments are relevant to the article, but not insightful.

2. Does the comment contain opinions of the commenter? For example, "Eliminating the deficit-Im sure this makes Mitch happy" is about the topic "Spending and Taxes" but without any insightful opinion. Such comments are relevant but not insightful.

3. Does the commenter provide argument to support her opinion? For example, "All this deficit crap reminds me of when Reagan ran for president ; how the deficit was terrible etc, etc, and after he got elected he ran up the biggest deficit ever. This was mostly due to spending on the military and tax cuts for the rich. This was even after he slashed domestic spending. If the US would wake up to the fact that we can't afford the wars, we might be able to move forward."

In total, the human judges annotated 1350 pairs of issue-comments for PM speech and 1150 pairs of issue-comments for Obama Speech. Since the annotation is still subjective, we calculated the inter-annotator agreement level using Cohen's Kappa coefficient. Cohen's kappa on quality is 0.8965 for all comments. On relevance the kappa is 0.7355 for all comments. We use the judgment from the judge who is stricter as our ground truth. The statistics of the labeled data are shown in Table 5.

Comment type	**Singapore**	**US**
Thoughtless	63.35%	68.25%
Thoughtful	36.65%	31.75%

Table 5: Comment statistics for both articles

7 Experiments

To check whether the features we have defined correlate with the thoughtfulness of comments based on human judgement, we first compute the Pearson correlation coefficients for all the features summarized in Table 3. The results are shown in Table 6. We observe that all features are positively correlated with the thoughtfulness of comments.

Feature	**Singapore**	**US**
F_1	0.3744	0.3594
F_2	0.3782	0.3755
F_3	0.3639	0.3911
F_4	0.3913	0.3554
F_5	0.1606	0.2437
F_6	0.1191	0.2146

Table 6: Pearson correlation coefficients between the features and the thoughtfulness of comments.

In the remaining of this section, we show our experimental results that answer the following questions:

RQ1: Does the KL-divergence relevance score based on nouns work better than the KL-divergence score based on all words?

RQ2: Which discourse relations have bigger impact on the performance?

RQ3: Which combination of various features gives the best prediction of thoughtfulness?

For all the experiments below, we use the standard precision, recall and F-score as our performance measures.

7.1 Relevance Model

To answer our RQ1, we first tested the performance on finding relevant comments on the Singapore dataset for both KL methods discussed in Section 4.5 . For this evaluation, we used only the labels z from the human judgment, i.e. the relevance judgment. We tested both relevance models: KL-divergence using all words and KL-divergence using only nouns. The results are shown in Table 7. We used the F-measure to evaluate the results. If score is greater than $\tau >$ -2.2 (set empirically), the comment is relevant to the topic. We can see that using nouns to compute the KL-divergence score works better. So, for the succeeding experiments we use F_5 which is the feature based on KL-divergence score between a comment and a target speech segment using nouns only.

Feature	Model	F-1
F_5	KL-Divergence using nouns only	**0.634**
F_6	KL-Divergence using all words	0.611

Table 7: Comparison between the two KL-divergence scores on the Singapore dataset.

7.2 Discourse Relations

To answer our RQ2, we studied the influence of various discourse relations on the F-measure of the comment thoughtfulness using the logistic regression model. For this evaluation, we used the labels y from the human judgment, i.e. the thoughtful comment. Table 8 shows the comparison of all three classes of discourse relations (Comparison, Expansion and Contingency) on comment quality. We can see that for both data sets, when *comparison* relations are used, the accuracy is the highest for both data sets. For the subsequent experiments, we use only the *comparison* relations to form our discourse feature, i.e. F_4 is set to be the number of *comparison* relations in a comment.

DR-Level	Singapore	US
All	0.6186	0.6464
Comparison	**0.6313**	**0.6538**
Expansion	0.5824	0.6111
Contingency	0.6213	0.6309

Table 8: Comparison of different classes of discourse relations using F-measure.

7.3 Thoughtful Comment Study

To answer RQ3, we conducted a detailed analysis on all the feature combinations we summarized in Table 3. We tested the thoughtfulness of the comments for a given article using the logistic regression model. The results of our experiments are shown in Table 9 for Singapore and in Table 10 for US. For all our experiments we performed 5-fold cross validation and with all the combinations of the features. For better analysis, we show only the most important combinations in the results.

For the Singapore data set, using linguistic features alone (F_1+F_2+F_3) leads to a F-score of 73.33%. Our hypothesis is that discourse relations play important role in detecting thoughtful comments. The results confirm that using discourse relations together with linguistic features

Feature Set	Recall	Precision	F-1
F_1+F_2+F_3	0.7097	0.7586	0.7333
F_1+F_2+F_4	0.8065	0.7143	0.7576
F_1+F_3+F_4	0.8387	0.6667	0.7429
F_2+F_3+F_4	0.8065	0.6944	0.7463
F_1+F_2+F_3+F_4	0.7742	0.7273	0.7500
F_1+F_2+F_3+F_5	0.7419	0.7667	0.7541
F_1+F_2+F_4+F_5	0.8065	0.7813	**0.7937**
F_1+F_3+F_4+F_4	0.7419	0.7931	0.7667
F_2+F_3+F_4+F_5	0.7742	0.7500	0.7619
F_1+F_2+F_3+F_4+F_5	0.7742	0.8000	0.7869

Table 9: Prediction results of thoughtful comments for Singapore using various feature combinations.

yields (F_1+F_2+F_3+F_4) a 75% F-score. But the model performs slightly better without syntactic features (F_1+F_2+F_4)) with 75.76%, which is a 0.76% increase over combined features and 2.43% higher than the linguistic features. Our second hypothesis is that relevance factors play an important role in detecting thoughtful comments. The results confirm that using relevance scores together with linguistic features and discourse relations (F_1+F_2+F_3+F_4+F_5) leads to 78.69% F-score, which is a 3.69% increase compared to linguistic together with discourse relations. Here again, we notice that the model has better performance without syntactic features (F_1+F_2+F_4+F_5) with F-score of 79.37, which is a 4.37% increase compared to linguistic together with discourse relations and 6.04% higher than linguistic features alone.

Feature Set	Recall	Precision	F-1
F_1+F_2+F_3	0.6522	0.6818	0.6667
F_1+F_2+F_4	0.7826	0.6429	0.7059
F_1+F_3+F_4	0.7391	0.6296	0.6800
F_2+F_3+F_4	0.7391	0.5862	0.6538
F_1+F_2+F_3+F_4	0.7826	0.6207	0.6923
F_1+F_2+F_3+F_5	0.7391	0.6296	0.6800
F_1+F_2+F_4+F_5	0.7826	0.6923	**0.7347**
F_1+F_3+F_4+F_4	0.7826	0.6429	0.7059
F_2+F_3+F_4+F_5	0.7391	0.6538	0.6939
F_1+F_2+F_3+F_4+F_5	0.7826	0.6667	0.7200

Table 10: Prediction results of thoughtful comments for US using various feature combinations.

For experiments on US data set, using linguistic features (F_1+F_2+F_3) alone leads to the F-measure of 66.67%. Discourse relations together with linguistic features (F_1+F_2+F_3+F_4) yields 69.23% F-measure which is a 2.56% increase over features without discourse relations. Here, we also notice that the model performs slightly better without syntactic features (F_1+F_2+F_4) with F-score of 70.59% which is 1.36% increase over combined features. Using relevance scores together with linguistic features and discourse relations (F_1+F_2+F_3+F_4+F_5) leads to 72% F-measure which is 2.77% increase compared to linguistic together with discourse relations. We also notice that the model has better performance with out syntactic

features (F_1+F_2+F_4+F_5) with F-score of 73.47, which is 4.24% increase compared to linguistic together with discourse relations.

During our analysis, we observed that the US data is less verbose compare to Singapore data. Another thing we also noticed is that the US data users focus more on the speech delivery rather than the actual speech issues. At the same time, they tend to discuss mostly one issue in each comment where as, Singapore users tend to combine many issues in their comments. So, even if one issue is justified in the comment, the comment is treated as thoughtful comment. This explains the performance differences in the two data sets. It will be interesting to study more fine grained opinion analysis at the comment level and we leave it for our future work.

7.4 Cross Collection Experiments

We further perform cross data collection experiments to test the performance of our model. We tested Singapore using USs' 5-feature model and viceversa. To compare the cross data collection results with original results, we depict Table 11 which shows the F-measure prediction on thoughtful comments. Singapore performed with a quality prediction F-measure lowered by 4.49%, whereas the US performance decreased by 5.11% compared to actual model.

		train	
		Singapore	**US**
test	**Singapore**	0.7937	0.7488
	US	0.6836	0.7347

Table 11: Cross data collection comparison. F-Measure for thoughtful comments.

We show some sample comments in Table 12 for both datasets. Due to space constraints we show 2 sample thoughtful and thoughtless comments for two issues in each articles.

Topic	**Quality**	**User Comment**
Innovation	Thoughtless	I love these plans on energy, but alas, the energy secretary appears to be asleep.
	Thoughtful	You want to really drive innovation, job growth and entrepreneurs? Make education, health care and retirement less of a burden on the average family, adopt more socialist policies like Norway
Spending & Taxes	Thoughtless	..Oh, so Obama "compromised" on the tax cuts for the wealthy
	Thoughtful	Low taxes aren't helping the vast middle and working class and aren't creating more jobs, it's a policy that only benefits the rich.
Housing	Thoughtless	By the way did anybody count the no of flags on a HDB flat. believe me 95% of the time u will take less than 10 sec to do it
	Thoughtful	I am glad that to hear more HDB houses to be built. But do I got a taste of this pie? What about those who are genuine to upgrade their existing 3 room flat but not 1st timer?..
National Service	Thoughtless	i'm still waiting before the budget and erection. otherwise i'll vote oppo. 9k is ?
	Thoughtful	Just 9000 for NSman. Those foreign scholar in NUS NTU got tution non-subsidize fee alone is 20000 one year. That even exclude lodging and return ticket fully paid by PAP.

Table 12: Sample comments: First two topics are from US and last two are from Singapore

7.5 Parameter Sensitivity

The regularization parameter λ in Equation 5 is set empirically. We study the optimal value and tuned it by regular cross validation. Figure 1 shows our experiments for both Singapore and US datasets. We get optimum results when we set λ to 0.1 or 1. We choose 0.1 for both datasets as it generates higher prediction performance in general.

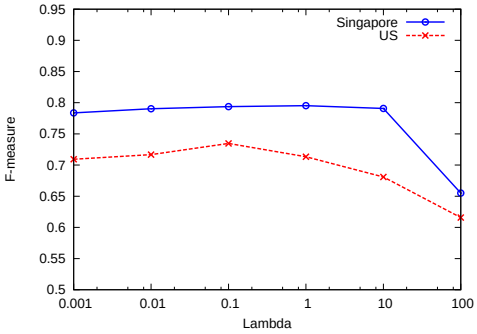


Figure 1: Regularization parameter sensitivity study

Conclusion and perspectives

Detecting thoughtful opinions is an important subtask of opinion mining and summarization. We perform an empirical study using syntactic, vocabulary, discourse and relevance features for prediction and combination of all is substantally better than the baseline surface features. Moreover, through our cross data collection experiments, we show that prediction using our approach achieves competitive performance.

Currently, we use KL divergence to compute the similarity but users tend to use abbreviations for some words and this impacts the performance of KL-divergence scores. We want to try other similarity techniques based on topic modeling and enhance the relevance performance. Extending the problem to identify the sentiment orientation is another useful subtask of opinion mining which we want to try next. In the future, we would like to extend our work to application base, and investigate the usage of thoughtful opinions in opinion summarization.

Acknowledgments

This research/project is supported by the Singapore National Research Foundation under its International Research Centre@Singapore Funding Initiative and administered by the IDM Programme Office.

References

Agichtein, E., Castillo, C., Donato, D., Gionis, A., and Mishne, G. (2008). Finding high-quality content in social media. In *WSDM '08: Proceedings of the international conference on Web search and web data mining*, pages 183–194, New York, NY, USA. ACM.

Amgoud, L., Bannay, F., Costedoat, C., Saint-Dizier, P., and Albert, C. (2011). Introducing argumention in opinion analysis: Language and reasoning challenges. In *Sentiment Analysis*

where AI meets Psychology, pages 28–34, Chiang Mai, Thailand. Asian Federation of Natural Language Processing.

Attali, Y. and Burstein, J. (2006). Automated essay scoring with e-rater v.2. *The Journal of Technology, Learning and Assessment (JTLA)*, 4(3).

Bian, J., Liu, Y., Zhou, D., Agichtein, E., and Zha, H. (2009). Learning to recognize reliable users and content in social media with coupled mutual reinforcement. In Quemada, J., Leon, G., Maarek, Y. S., and Nejdl, W., editors, *WWW*, pages 51–60. ACM.

Chen, B.-C., 0002, J. G., Tseng, B. L., and 0015, J. Y. (2011). User reputation in a comment rating environment. In Apte, C., Ghosh, J., and Smyth, P., editors, *KDD*, pages 159–167. ACM.

Choi, Y., Breck, E., and Cardie, C. (2006). Joint extraction of entities and relations for opinion recognition. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, pages 431–439.

Cui, H., Mittal, V., and Datar, M. (2006). Comparative experiments on sentiment classification for online product reviews. In *Proceedings of the 21st AAAI Conference on Artificial Intelligence*, pages 1265–1270.

Ghose, A. and Ipeirotis, P. G. (2011). Estimating the helpfulness and economic impact of product reviews: Mining text and reviewer characteristics. *IEEE Trans. Knowl. Data Eng.*, 23(10):1498–1512.

Hsu, C.-F., Khabiri, E., and Caverlee, J. (2009). Ranking comments on the social web. In *CSE (4)*, pages 90–97. IEEE Computer Society.

Huang, X. and Croft, W. B. (2009). A unified relevance model for opinion retrieval. In *Proceedings of the 18th ACM conference on Information and knowledge management*, CIKM '09, pages 947–956, New York, NY, USA. ACM.

Jiang, L., Yu, M., Zhou, M., Liu, X., and Zhao, T. (2011). Target-dependent Twitter sentiment classification. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics*, pages 151–160.

Kim, S.-M., Pantel, P., Chklovski, T., and Pennacchiotti, M. (2006). Automatically assessing review helpfulness. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 423–430, Sydney, Australia. Association for Computational Linguistics.

Liu, J., Cao, Y., Lin, C.-Y., Huang, Y., and Zhou, M. (2007). Low-quality product review detection in opinion summarization. In *Proceedings of the Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 334–342. Poster paper.

Lu, Y., Tsaparas, P., Ntoulas, A., and Polanyi, L. (2010). Exploiting social context for review quality prediction. In Rappa, M., Jones, P., Freire, J., and Chakrabarti, S., editors, *WWW*, pages 691–700. ACM.

Macdonald, C., Ounis, I., and Soboroff, I. (2009). Overview of the trec 2009 blog track. In *TREC*.

Pang, B. and Lee, L. (2008). Opinion mining and sentiment analysis. *Foundations and Trends in Information Retrieval*, 2(1-2):1–135.

Pang, B., Lee, L., and Vaithyanathan, S. (2002). Thumbs up? Sentiment classification using machine learning techniques. In *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing*, pages 79–86.

Pitler, E. and Nenkova, A. (2008). Revisiting readability: A unified framework for predicting text quality. In *EMNLP*, pages 186–195. ACL.

Pitler, E., Raghupathy, M., Mehta, H., Nenkova, A., Lee, A., and Joshi, A. K. (2008). Easily identifiable discourse relations. In Scott, D. and Uszkoreit, H., editors, *COLING (Posters)*, pages 87–90.

Popescu, A.-M. and Etzioni, O. (2005). Extracting product features and opinions from reviews. In *Proceedings of the Human Language Technology Conference and the Conference on Empirical Methods in Natural Language Processing*.

Prasad, R., Dinesh, N., Lee, A., Miltsakaki, E., Robaldo, L., Joshi, A. K., and Webber, B. L. (2008). The penn discourse treebank 2.0. In *LREC*. European Language Resources Association.

Saito, M., Yamamoto, K., and Sekine, S. (2006). Using phrasal patterns to identify discourse relations. In *Proceedings of the Human Language Technology Conference of the NAACL, Companion Volume: Short Papers*, pages 133–136, New York City, USA. Association for Computational Linguistics.

Tang, R., Ng, K. B., Strzalkowski, T., and Kantor, P. B. (2003). Automatically predicting information quality in news documents. In *HLT-NAACL*.

Wu, Y., Zhang, Q., Huang, X., and Wu, L. (2009). Phrase dependency parsing for opinion mining. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 1533–1541.

Zhang, M. and Ye, X. (2008). A generation model to unify topic relevance and lexicon-based sentiment for opinion retrieval. In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '08, pages 411–418, New York, NY, USA. ACM.

A Distributed Platform for Sanskrit Processing

Pawan Goyal[1] Gérard Huet[1]
Amba Kulkarni[2] Peter Scharf[3] Ralph Bunker[4]
(1) Inria Paris-Rocquencourt
(2) Department of Sanskrit Studies, University of Hyderabad
(3) Blaise Pascal Chair, Université Paris 7, and The Sanskrit Library
(4) Maharishi University of Management, Fairfield, Iowa
Pawan.Goyal@inria.fr, Gerard.Huet@inria.fr, apksh@uohyd.ernet.in,
Peter.Scharf@inria.fr, rbunker@lisco.com

Abstract

Sanskrit, the classical language of India, presents specific challenges for computational linguistics: exact phonetic transcription in writing that obscures word boundaries, rich morphology and an enormous corpus, among others. Recent international cooperation has developed innovative solutions to these problems and significant resources for linguistic research. Solutions include efficient segmenting and tagging algorithms and dependency parsers based on constraint programming. The integration of lexical resources, text archives and linguistic software is achieved by distributed interoperable Web services. Resources include a morphological tagger and tagged corpus.

Keywords: Indian language technology, Resources and annotation, Morphology and POS tagging, Parsing.

Proceedings of COLING 2012: Technical Papers, pages 1011–1028,
COLING 2012, Mumbai, December 2012.

1 Introduction

Formal and computational linguistics was dominated by English at its inception and developed in subsequent decades primarily in the environment of European languages. More recently there has been a concerted effort to undertake formal linguistic analysis of a wide variety of languages, with particular interest in those with dramatically different features, and to enrich linguistic theory to account for linguistic variety. In spite of this effort, analytic structures and procedures utilized in formal linguistics remain dominated by those invented for, and most suitable for, English and other European languages. Linguistic theory remains unduly weighted in favor of European languages even as their extension to the variety of the world's languages involves undue complication thereby revealing their inadequacy in representing language universally. The inadequacy of contemporary computational methods is vividly apparent in the analysis of Sanskrit. Recent worldwide collaboration to overcome the challenges to conducting computational linguistic research on Sanskrit offers insights into methods and procedures that may be useful generally for languages that differ markedly from western European languages.

1.1 Sanskrit

Sanskrit is the primary culture-bearing language of India, with a continuous production of literature in all fields of human endeavor over the course of four millennia. Preceded by a strong oral tradition of knowledge transmission, records of written Sanskrit remain in the form of inscriptions dating back to the first century B.C.E. Extant manuscripts in Sanskrit number over 30 million - one hundred times those in Greek and Latin combined - constituting the largest cultural heritage that any civilization has produced prior to the invention of the printing press. Sanskrit works include extensive epics, subtle and intricate philosophical, mathematical, and scientific treatises, and imaginative and rich literary, poetic, and dramatic texts. The primary language of the Vedic civilization, Sanskrit developed constrained by a strong grammatical tradition stemming from the fairly complete grammar composed by Pāṇini by the fourth century B.C.E. In addition to serving as an object of study in academic institutions, the Sanskrit language persists in the recitation of hymns in daily worship and ceremonies, as the medium of instruction in centers of traditional learning, as the medium of communication in selected academic and literary journals, academic fora, and broadcasts, and as the primary language of a revivalist community near Bangalore. The language is one of the twenty-two official languages of India in which nearly fifty thousand speakers claimed fluency in the 1991 Indian census.

India developed an extraordinarily rich linguistic tradition over more than three millennia that remains under-appreciated and under-investigated. A cursory glance at the long tradition of discussion and argumentation within and between Indian sciences of phonetics (śikṣā), grammar (vyākaraṇa), logic (nyāya), ritual exegesis (karmamīmāṃsā), and literary theory (alaṃkāraśāstra) reveals that Indian linguistic traditions have much to offer contemporary linguistic theory in the areas of phonetics, morphology, syntax, and semantics.

1.2 Computational linguistic processing challenges

Since computational linguistics developed primarily in the environment of western European languages, its methods were structured and remain suited to those languages. Prior to undertaking any kind of computational analysis of Sanskrit text, one must deal with several challenges presented by features of the language that differ markedly from those of modern western European languages. In addition to the complexity introduced by lexical complements, which is relevant in these languages

as well, Sanskrit has orthographic, prosodical, and inflectional complexities not encountered in western European languages. The rich inflectional and derivational morphology of the language permits relationships that are shown positionally in western European languages to be made known by the morphology instead. As a result, the word order is much less constrained by governance structure and is free to intimate discourse structure and performative aspects of language. For the analysis of Sanskrit syntax, therefore, positional grammars, and constituency parsers which are based on them, are not very relevant, and dependency parsers are more suitable.

1.2.1 Prosody and orthography

In English, where the heritage is received in writing, and standardized spelling and printing were introduced several hundred years ago, a given morpheme is represented with a single orthography despite the fact that it has different surface phonetic representations in different contexts. For example, the past tense suffix -⟨ed⟩ is so written despite three distinct phonetic realizations in three clearly defined contexts:

/t/ e.g. dip /dɪp/, dipped /dɪpt/
/d/ e.g. boom /bu:m/, boomed /bu:md/
/id/ e.g. loot /lu:t/, looted /lu:tid/

In contrast, in Sanskrit, where oral tradition dominated the sphere of learning and an advanced discipline of phonetics explicitly described prosodic changes, these prosodic changes, well known by the term *sandhi*, are represented in writing. Hence the past passive participle suffix *-ta* variously realized as *ta* or *dha* depending solely upon the phonetic context, is written as follows:

/ta/ e.g. from *su* 'press', *suta* 'pressed'
/d^ha/ e.g. from *budh* 'awake', *buddha* 'awakened'

Moreover, the prosodic changes obscure word boundaries in speech, and these word boundaries are correspondingly eliminated in writing as well. For example, *vasati* 'dwells' followed by *atra* 'here' becomes वसत्यत्र (*vasatyatra*) in continuous speech. The semisyllabic Indic scripts such as Devanāgarī forestall word separation here obliterating the word boundary. Some prosodic changes, like this one, can be separated in alphabetic Roman transcription despite the sound alteration, viz. *vasaty atra*. Other prosodic changes, however, preclude word separation even in alphabetic transcription because the final sound of the preceding word and the initial sound of the following word merge in a single sound. Thus *vidyā* 'knowledge' *āpyate* 'is attained' becomes *vidyāpyate*; the single sound *ā* belongs to both words. The most difficult task in parsing a Sanskrit sentence is determining the word boundaries. Solutions to the problem have valuable ramifications for speech analysis where a similar problem is encountered in virtually all languages.

1.2.2 Inflectional morphology

In English, inflectional morphology is minimal. A present active verbal paradigm contains six slots: three for first, second and third person times two for singular and plural number. Yet the forms that fill these slots number just two, for example, *go* and *goes* for the verb 'to go'. Description of the abstract grammatical structure requires mentioning five items while description of the forms directly requires mentioning only two. Grammatical description is therefore more prolix than listing. It is nearly as efficient to describe English morphology with reference to individual forms as it is to describe it in abstract grammatical structures. (Karp et al., 1992) create a hash table of just 317,477 forms from 90,196 lexical entries, a ratio of 3.5:1. The reverse is true for Sanskrit. In Sanskrit full

verb paradigms number hundreds of unique forms in as many as sixteen hundred slots. The modest full-form lexicon created by (Scharf and Hyman, 2009a) from a lexicon of 170,000 entries numbers more than eleven million forms, a ratio of 64.7:1. It is by far more economical to describe such forms in abstract grammatical categories than it is to list them. The implication of the brevity of grammatical description of Sanskrit in comparison to listing forms is that it is misguided to attempt to describe Sanskrit grammar with reference to individual word forms. This fact was explicitly recognized by Patañjali in his massive commentary *Mahābhāśya* on Pāṇini's concise grammatical description of Sanskrit in the *Aṣṭādhyāyī*.

1.2.3 Lexical complements

In statistical analysis of a corpus of sentences, slots for lexical complements are combined. In computational linguistic processing of English it is recognized that the forms *is* and *are* belong to the same lexical unit as the forms *was* and *were*. Likewise in Sanskrit grammar, forms derived from the root *bhū* are recognized as complements of forms derived from the root *as*.[1] Because it is concerned only with individual word forms, computational linguistic processing of English treats *is* and *are* as lexical complements of each other and *was* and *were* as lexical complements of each other in the same manner as it considers the first pair as lexical complements of the second pair. Such processing treats *a* and *an* as lexical complements in the same manner. Historical linguists differ from computational linguists in their treatment of these forms. They view *is* and *are* as inflectional varieties derived from one common root, and *was* and *were* as inflectional varieties derived from another root. They consider the two roots to be lexical complements. They recognize that *a* is a phonetic variant of *an*, both derived from the word *one*. Sanskrit grammarians agree with historical linguistics here. Only semantically related roots are treated as lexical complements. Variants such as *a/an* are treated as phonetic variants, and variants such as *is/are* are treated as inflectional variants.

1.2.4 Syntax

In languages such as English where word order is strongly associated with roles, it may be reasonable to define positions in relation to roles as is done in positional grammars. Hence in an active sentence, the first position is called the subject position, the second the verb position and the third the object position. In free-word order languages such as Sanskrit, however, position does not determine role. Although certain patterns are common — such as subject, object, verb — even unmarked word order leaves some roles in indeterminate position. Alteration of the word order does not change the roles, and the position is highly influenced by discourse structure and emphasis. Phrase-structure grammars are unsuitable to describe governance structure which is more accurately described by dependency grammars.

2 Birth of a discipline

A number of projects began accumulating digitized texts in the late 1980s. The largest collection was made by the TITUS[2], which accumulated more than eighty digital Sanskrit texts within a decade. The next decades witnessed the growth of other large collections including GRETIL, which serves as a central registry of digitized Indic texts.

In the meantime a couple of projects developed digital dictionaries of Sanskrit. The Cologne Digital Sanskrit Lexicon project began by digitizing Monier Williams' A Sanskrit-English Dictionary (MW)

[1] *Aṣṭādhyāyī* 2.4.52 *aster bhū*

[2] `http://titus.uni-frankfurt.de/`

between 1994 and 1996, and followed by digitizing several other major bilingual Sanskrit dictionaries. The Digital Dictionaries of South Asia project at the University of Chicago included Apte's and MacDonell's Sanskrit-English dictionaries among its digitized Indian language dictionaries.

There were a few early isolated attempts to process Sanskrit text mechanically, such as Pushpak Bhattacharya's Sanskrit parser included as part of his M.Tech. thesis at IIT Kanpur in 1987 and Pr. Lakshmitatachar's verbal cognition generator for Bhandarkar's Sanskrit primer developed at the Academy of Sanskrit Research in Melkote in the early 1990s (Rāmapriya and Saumyanārāyaṇa, 2001).

The Indian Ministry of Communications and Information Technology provided a strong impetus for computational processing of Indian languages beginning at the turn of the century with its Technology Development for Indian Languages program (TDIL). Several periodic conferences were launched to foster research in computational linguistics, such as the International Conference on Natural Language Processing (ICON), and the Language Engineering Conference (LEC). The Akshar Bharati group developed "Anusāraka", a language accessor, for accessing texts in other languages that employed techniques inspired by Pāṇini's *Aṣṭādhyāyī* (Bharati et al., 1995). K. V. Ramakrishnacaryulu introduced natural language processing programs specifically for Sanskrit at the Rashtriya Sanskrit Vidyapeetha, Tirupati, such as "Śābdabodha Systems and Language Technology" in 2005.

In 2002, under the guidance of Amba Kulkarni, the toy morphological analyser developed at Melkote was enriched with the MW lexicon and the Dhātu-ratnākara database resulting in a wide coverage morphological analyser. Amba Kulkarni developed prototypes of several other analytic tools for Sanksrit when she began teaching specialized courses in the subject at Tirupati. Her appointment as the head of the newly formed Department of Sanskrit Studies at the University of Hyderabad, and Girish Nath Jha's appointment to the Special Center for Sanskrit Studies, J.N.U., Delhi beginning in 2002 allowed the systematic training of students in Sanskrit computational linguistics.

In 1998-1999, Peter Scharf and Ralph Bunker developed a Web-based Sanskrit reader program at Brown University called *Kramapāṭha* equiped in 2001 by Hyman with an index program and audio feature. The index program allowed Peter Scharf's *Rāmopākhyāna* to be searched by lexical and inflectional categories as well as by verbal roots, nominal stems, inflected forms, text ranges, or combinations thereof. In 2003-2004, Hyman and Peter Scharf collaborated to produce a digital edition of Whitney's roots (Whitney, 1997), that served as the source of verbal stems for their inflectional generation software. Between 2006 and 2009 Peter Scharf led the International Digital Sanskrit Library Integration project in the Classics Department at Brown University. The project created a digital Sanskrit library by integrating the texts provided by the TITUS with the digital MW dictionary of the Cologne Digital Sanskrit Lexicon project at Universität zu Köln (CDSL). The dictionary was upgraded with the assistance of Jim Funderburk and R. Chandrashekar by converting character code markers to explicit XML tags and systematically classifying and tagging additional information.

Separating linguistic processing from issues of input and display simplifies linguistic processing and also permits precise processing and display of accented dialects. Peter Scharf and Hyman designed the Sanskrit Library Phonetic encodings (SLP), described in (Scharf and Hyman, 2009b), after a thorough investigation of ancient Indian linguistic treatises, that allows all sounds represented in Vedic texts to be represented digitally. After an investigation of Sanskrit paleography, Peter Scharf initiated worldwide collaboration to extend the Unicode Standard to include 68 additional

characters required for the proper display of the ancient Vedic heritage texts of India[3]. The Unicode Standard version 5.2 incorporated the characters in two code blocks, Devanagari Extended and Vedic Extensions under South Asian Scripts on the Unicode Character Code Charts page[4]. SLP serves as the basis of a suite of transcoders that convert between standard Sanskrit Romanization of Sanskrit in Unicode, several popular Roman meta-encodings, and the Unicode pages of the major Indic scripts. Users are permitted to select their preferences for input and display at the Sanskrit Library site.

Around 2000, Gérard Huet started to develop a Sanskrit Heritage platform (SH), centered around an electronic version of the Sanskrit-French Sanskrit Heritage dictionary[5]. The dictionary was structured from the start to serve both as a computerized lexical database for morphology generation, and as a human-readable hypertext encyclopedia on Classical India (Huet, 2001, 2004). It is internally consistent in that each lexical entry is provided with hypertext links to its generating components, and it prepares the ground for syntax analysis by systematically formalizing information about complements (ākāṅkṣā).

Various tools for morpho-phonemic computation, as well as efficient structures for lexicon representation, were adapted to Sanskrit from a general computational linguistics toolkit called the Zen library, implementing general finite state transducers in functional programming style, as instances of a new relational computing paradigm called effective Eilenberg machines (Huet, 2002; Huet and Razet, 2006, 2008; Razet, 2009). The global architecture of this platform is that of interconnected Web services allowing interaction with digital libraries and other external resources. The main tool is a Sanskrit Reader, allowing segmentation (sandhi analysis), tagging, and parsing (Huet, 2003, 2005, 2006, 2007, 2009; Goyal and Huet, 2013).

In 2008 the Indian Government funded a major consortium project to develop various tools for analysis of Sanskrit text and a Sanskrit-Hindi Machine Translation System. Sanskrit scholars and computational linguists collaborated to develop the prototype of an interactive reader consisting of the 100-verse Saṅkṣepa Rāmāyaṇa. They also developed elaborate guidelines for annotating sandhi, compounds (*samāsa*) and syntactico-semantic roles (*kāraka*), and an annotated 800K corpus. Compound analysis is essential to Sanskrit parsing because 15-20% of the words in a random text are compounds and compounding is productive. A modular compound processor (Kumar, 2012) was developed that segments a given linear string into morphologically valid components (Kumar et al., 2010), determines the underlying constituency structure (Kulkarni and Kumar, 2011), and identifies the compound type (Kulkarni and Kumar, 2013). A paraphrase of the compound is then produced from the labeled constituency tree (Kumar et al., 2009). The morphological analyser previously developed by Amba Kulkarni was enhanced further by employing the head words of MW, and supplying additional derived forms before generating a full-form lexicon of 140 million words. (Kulkarni and Shukl, 2009). The constraint-based parser developed by her employing this analyser is described in section 6 below. All these tools for analysis of Sanskrit texts were used in a Sanskrit-Hindi language accessor (*anusāraka*) and a machine translation system. Comparative study of the divergences between Sanskrit and Hindi were taken up to improve the translation quality (Shukla et al., 2010) of the translator. The morphological analyser, generator, sandhi joiner and splitter, full-fledged parser, and Sanskrit-Hindi Machine Translation system were assembled in what is called *Saṁsādhanī*[6].

[3] http://www.sanskritlibrary.org
[4] http://www.unicode.org/charts
[5] http://sanskrit.inria.fr
[6] http://tdil-dc.in/san

These various efforts started coordinating themselves around 2006, with the creation of a joint team in Sanskrit computational linguistics between INRIA and Department of Sanskrit Studies, University of Hyderabad. In October 2007 the First International Sanskrit Computational Linguistics Symposium (Huet et al., 2009), organized by Gérard Huet at INRIA, allowed the presentation of the various teams and tools, and the development of cooperative software and resources. It was soon followed by the Second Symposium, organized at Brown University by Peter Scharf in May 2008, the Third one organized at University of Hyderabad by Amba Kulkarni in January 2009 (Kulkarni and Huet, 2009), the Fourth one organized at J.N.U. Delhi by Girish Nath Jha in December 2010. The Fifth one is scheduled for January 2013, organized at I.I.T. Bombay by Malhar Kulkarni.

In 2010–2011 the Sanskrit Library linked its texts to the Sanskrit Heritage reader. Each sentence in which sandhi has not been analysed is dynamically linked to the SH parser. The parser analyses the sentence using various syntactic criteria and a full-form lexicon of 700,000 forms derived from the SH lexicon of about 25,000 words. Unpenalized solutions are selected and displayed. The site allows one to examine penalized solutions and to reedit the sentence and resubmit it for further analysis. The Sanskrit Heritage site additionally allows one to submit analysed sentences for syntactic analysis by Amba Kulkarni's dependency tree parser.

The year 2012 saw a significant progress in the integration of the various tools. Pawan Goyal integrated an HTML version of the Monier-Williams dictionary as an alternative plug-in component to the Sanskrit Reader. The simultaneous invitation of Peter Scharf and Ralph Bunker to Paris eased the synchronization of tagging schemes and development of more robust protocols for interoperable Web services. In close collaboration with the other authors, Ralph Bunker developed a software-assisted human interface for morphological tagging currently being used by human annotators to prepare a tagged corpus.

3 Basic Architecture

The basic architecture of the collaborative platform is based on interactions between various Web-services. The main idea is not to have a monolithic system but various platforms, where selected components can inter-operate. These components can be software or linguistic resources. The glue between various platforms is interoperable Web-services via user interfaces and remote procedure calls.

This way of doing distributed computing has several advantages from a software engineering point of view. Firstly, Web technology gives a universal standard user interface with XHTML. Conformant HTML pages permit a uniform viewing by the various browsers offered by the various operating systems of personal computers and workstations. The technology offers automatic adaptation to the display medium, thus accommodating tablets, personal assistants, and smartphones. Furthermore, Unicode allows display in all the scripts of all the human languages. For Sanskrit, this means it is easy to display in Devanāgari script, as well as in the standard Indological romanised script with diacritics, as well as in the various transliteration schemes in use. Actually our joint distributed platform recognizes four such transliteration schemes, permitting equally easy access to scholars trained in using one scheme or another.

Secondly, developing separate components at the various sites does not commit us to any specific programming environment. The various teams at the various sites use different programming languages. There is no need for linking the executables of these various services. Finally, versioning is distributed, there is no need of synchronisation of new versions of the various services, once clear interfaces for data interchange are agreed upon (that is, we only have to agree on the XML abstract

structures defining the marshalling of interchanged data at the interfaces).

We have not felt the need to have a sophisticated orchestration of these various services. The main ingredient is remote procedure call (so-called CGI in the Web jargon). We remedy the poverty of the memory-less protocol (HTTP) by transmitting parameters summarizing the interaction history in the current session.

One common feature of our various developments is the use of UNIX platforms (Linux or MacOS) for development and server deployment. Users of the software, for instance annotators, may of course use any client operating system.

The configuration of the various platforms allows a choice between using a service as a remote process using network communication with the proper server, or alternatively to the same service run locally on the client station. This is easily achieved in Unix clients, where Web servers such as Apache or Tomcat are easy to install. Sometimes, one service is available locally as a plug-in to the server site. Thus, the Sanskrit Heritage segmenter is available as a plug-in to the University of Hyderabad Sanskrit computational platform, in order to undo sandhi in sandhied corpus. Conversely, the University of Hyderabad Sanskrit parser, using sophisticated constraint technology, may be used in the Sanskrit Heritage platform as a filter to its segmenter-tagger, superior in precision to its original crude dependency analyser.

4 Lexicon

Independent projects had previously used different lexicons as the basis for generating inflected forms used in linguistic software. The task of coordinating those lexicons with each other and with other available lexical resources presents a challenge. A current project jointly funded by the NEH and the Deutsche Forschungsgemeinschaft extends the Sanskrit Library's multidictionary interface by integrating supplements to the major bilingual dictionaries already included, and by adding specialized dictionaries, indigenous Indian monolingual dictionaries, traditional thesauri, and traditional linguistic analyses. Even after data-entry of the various lexical sources, the task of integrating them is complicated by the different conventions used by their original compilers. Compilers differ in the scripts they use, conventions of sandhi, selection of stem versus an inflected form, determination of the base form, etc. The project examines the conventions used in each lexical source, and determines ways of mapping the differences to each other.

The Sanskrit Heritage platform uses a Sanskrit-French dictionary. The desire arose to express the output of the various tools of the platform multilingually. The first step was to project the headwords of the SH dictionary onto those of the digitalised Monier-Williams dictionary. To this end, Pawan Goyal engaged in using data-mining techniques and automated translation tools to develop a protocol for the non-trivial task of mutually linking lexical resources, as described in the remainder of this section.

Let us consider the stem, *aṅga* in Sanskrit. This stem appears in two SH entries:

aṅga$_1$: membre; partie du corps; le corps en entier; la personne, la forme
En: member, part of the body the whole body, the person, the form

aṅga$_2$: affirme, confirme, ou exprime le désir ou l'impatience bien, d'accord; certes, vraiment; s'il vous plaît; vite
En: affirms, confirms, or express a desire or impatience, okay, sure, really, please, quickly

MW also has this stem listed in two different entries as:

aṅga$_1$: a particle implying attention, assent or desire, and sometimes impatience, it may be rendered, by well

aṅga$_2$: a limb of the body

So, in this example, *aṅga*$_1$ in SH should link to *aṅga*$_2$ in MW. Clearly, the matching can not be done simply based upon the name of the stem, but the concepts involved in the corresponding entry also need to be used. Thus, the headword linking problem is not trivial because of 1). **Homophony indexes:** SH and MW have their own systems of giving homophony indexes to the entries. Thus *aṅga*$_1$ in SH may correspond to either *aṅga*$_1$ or *aṅga*$_2$ in MW, and 2). **Cross-lingual resources:** While SH is a dictionary from Sanskrit to French, MW is a dictionary from Sanskrit to English. Thus, it is difficult to match the direct meanings as obtained after extracting the meaning text from both the lexicons.

4.1 Labeling MW with the lexical information

Pawan Goyal converted the XML file of the MW dictionary described in section 2 to strict XHTML by XSLT (Extensible Stylesheet Language Transformations). Each entry in the XHTML dataset was labeled with its lexical category information. This avoids the homophony problem across the lexical categories. For an example, the Sanskrit word *bhū* can be used as a noun, meaning 'earth' as well as a root meaning 'to become' and is given the homophony *bhū*$_2$ and *bhū*$_1$ in the MW XML dataset for the lexical categories corresponding to noun and root respectively. In the XHTML file, the nouns were labeled with a suffix '-*pr*[7]' and the roots were labeled with a suffix '-*dh*[8].C_n', where C_n denotes the corresponding class number of the verb (*gaṇa*) and varies from 1 to 10. In the case where a root entry has more than one *gaṇa*, multi-labels were given to that entry. Nominal verbs and verbs with preverb sequences were labeled with the suffix '-*dh*.Nom[9]' and '-*cpvb.*(ps,dh). *cpvb.*(ps,dh) denotes the verb with a preverb sequence 'ps' and the root 'dh'. Thus a verb '*ānī*', consisting of preverb '*ā*' and root '*nī*' was labeled as *ānī*-*cpvb.*(*ā*,*nī*).

4.2 Matching Dictionary Headwords

Once the MW was labeled with the lexical information, the headwords from the SH dictionary were matched with the MW headwords based on their lexical categories, which is explicit in the SH dictionary. As a rough estimate, there are approximately 16000 nouns, 600 roots, 110 nominal verbs and 1200 verbs with preverb sequences in the SH dictionary.

From the MW XHTML pages, the labels corresponding to each entry were extracted (called MW_{ent} henceforth), which, as discussed above, were marked with the lexical information. For each lexical category, the entry in the SH dictionary was looked up in the MW_{ent} and the search results were categorized in one of the following categories for further treatment: 'one to one' mapping of headword, 'many to one' mapping, 'one to many' mapping, 'many to many' mapping and 'not found'. For instance, 'one to many' mapping implies that a single headword in SH maps to many different headwords in MW and requires further disambiguation to select the desired match among the many possible matches.

While one to one mapping indicates that an entry in SH matches with one and only one entry in MW and the match results were used as it is, the cases of many to one mappings were very rare and were

[7] '*pr*' stands for *prātipadika*, the substantival base.
[8] '*dh*' stands for *dhātu*, the verbal base
[9] *dh*.Nom stands for the nominal verbs

dealt with in the same way as that of one to one mapping. The reasoning behind this design decision was the fact that MW has a wider coverage of lexicon entries. The next two cases, 'one to many' and 'many to one' mappings were problematic because these require further disambiguation to select exactly which of the many headwords in MW corresponds to the SH headword. To solve this problem, an approach based on matching the word concepts in the two dictionaries was employed.

4.3 Matching dictionary headwords using concept matching

A dictionary headword can be considered to be a concept-node in the particular ontology expressed by the dictionary and thus, the problem of matching dictionary headwords can be seen as the problem of ontology mapping. In the particular approach adopted by the authors, the problem of headword matching was translated to the problem of matching the concepts expressed by the particular concept-nodes, these headwords represent in different ontologies.

Matching the headword concepts was not so trivial because of the fact that while the SH dictionary stores the word concepts in French, the MW dictionary contains word concepts in English. To overcome this problem, the concepts from the SH dictionary were extracted and translated into English using Google Translate[10]. This ensured that we had the concepts for each entry in SH in the same language as that in MW. For a given word in SH, the concepts from the corresponding MW headwords were extracted. The concepts were preprocessed to remove stopwords such as {on, for, to, and, by} etc. These concepts were then considered similar to the Wordnet notion of 'synset'. Representing these concepts as a 'bag-of-words', the matching between the concept vectors X and Y was performed using the following function:

$$match(X, Y) = \sum_i \sum_j sim(X_i, Y_j)$$

where $sim(X_i, Y_j)$ denotes the similarity function between the strings X_i and Y_j, belonging to the concept vectors X and Y respectively. The similarity function accounted for the fact that even if the two strings have different suffixes, they represent the same concept. For example, 'rained' and 'raining' represent the same concept. The similarity function was directly proportional to the intersecting letters between X_i and Y_j and was inversely proportional to the maximum number of letters in X_i or Y_j. A threshold value was also given such that the similarity function only contributes to the matching score if its value is greater than the threshold.

Once an SH word was matched to all the possible MW headwords, the matching values were sorted and the headword with the highest match was marked as the suitable match. However, if the top two headwords have the same matching score (this also includes the case, where all the headwords obtained a score of 0.0), the SH headword along with all the MW headwords and their meanings were dumped in a text file interface, where the exact match was decided manually.

Note that the problem of solving 'one to many' mapping in the case of verbs with preverb sequences can be expressed in terms of matching the structure (ps,dh) between SH and MW and assuming that the roots have been mapped from SH to MW by following the procedure outlined above, this would not require the matching of headword meanings again. For example, consider the verb *saṃmā*$_1$ in SH, which is analysed as consisting of preverb *sam* and the verb *mā*$_1$. Once this *mā*$_1$ has been mapped to *mā*$_3$ in SH, this information can be utilized to match *saṃmā*$_1$ in SH with *saṃmā-cpvb*(*sam*,*mā*$_3$) in MW.

[10] http://translate.google.com/

5 Segmentation and Tagging

The first computational problem attacked in the framework of the Sanskrit Heritage platform was segmentation, i.e. *sandhi-viccheda*. Sandhi occurs in Sanskrit in several places. In generative morphology, it occurs internally for stem formation and affixes glueing, for instance for declension and conjugation. This so-called internal sandhi is complex, since it gives rise to long-distance retroflexion, not easily invertible by finite-state methods. On the other hand, junction of words within the sentence, as well as compound formation, uses a simpler notion of external sandhi, that may be modeled as a rational relation over words, invertible by finite-state techniques. It was thus decided to divide the task into generation of non-compound word forms in pre-processed databanks, and analysis of sentences in terms of these elementary forms.

A general toolkit for computational linguistics in functional programming style, called Zen (Huet, 2002), was first implemented. It uses in a systematic manner a notion of *decorated tries*, usable both as efficient data structures for lexicon representation, and applicative representation of automata and transducers. A new general framework for relational programming, called effective Eilenberg machines (Huet and Razet, 2006, 2008; Razet, 2009), was designed as a restriction to partial recursive relations of a mathematical model of automata theory due to Samuel Eilenberg (Eilenberg, 1974). This framework, allowing reactive programming over streams of data, proved adequate to the efficient solution of the segmentation problem in Sanskrit (Huet, 2005).

Since segmentation is directed by the inflected forms databases, presented as lemmatized segments, each segmentation solution gives rise to a canonical tagging, where each segment is tagged with the set of combinations of lexemes and morphological parameters used to generate it. The main problem to be faced was the enormous number of potential solutions of even moderately long sentences. In order to control this complexity, several devices were introduced. Firstly, the notion of binary compound was generalized into a notion of multi-segment pre-compounds, replacing a potentially exponential number of binary trees into a single linear pre-compound. Secondly, a dependency graph analysis was performed, on a restricted subset of semantic roles (mostly agent and patient). Each analysis gives rise to some penalties issued from graph-matching. Restricting the segmentation solutions to the minimal penalty ones yields a shallow parser with more manageable output. These developments have been well documented in publications (Huet, 2003, 2005, 2006, 2007, 2009), and thus will not be detailed further here.

The first effective collaboration effort between INRIA and Department of Sanskrit Studies, University of Hyderabad consisted in making the Saṁsādhanī dependency parser available as a further filtering on the Heritage engine segmenter, making it more precise, and allowing the visualization of the dependency graph, labeled with semantic roles. Conversely, the Heritage segmenter was made available in the Saṁsādhanī system as a plug-in, allowing the processing of sandhied corpus.

One important concern is that of the correctness of the computational processes used in the morpho-phonetics routines of the Heritage engine with respect to the Pāṇinian grammatical tradition, seen as a gold standard. To this effect, Pawan Goyal and Gérard Huet strived to give correspondences between the computation routines, and sequences of rewrite rules (*sūtra*s) from Pāṇini's grammar. The current state of this correspondence will appear as (Goyal and Huet, 2013).

The next concern was to start the development of a Sanskrit dependency treebank, a necessary development to benefit from statistical methods and obtain more precise analysers. To this effect, a cooperation between the Sanskrit Library effort and the Sanskrit Heritage platform was started, in view of using the platform for semi-automatic annotation of corpus by Sanskrit specialists. This more recent development is giving rise to a new interface to the segmenting tool, allowing the

synthetic visualisation of all segmentation solutions. The annotator may select any segment, and an automatic tool trims away from the forest of all solutions the ones that are inconsistent with the choice. This allows an exponential saving, and fast focusing on the intended interpretation.

6 Dependency Parsing

The parse of positional languages such as English are well expressed by constituency structure while languages like Sanskrit which are morphologically rich and to a large extent free word order are better represented by a dependency tree where the nodes represent the prātipadikas or dhātus and the edges between the nodes represent the relations between them expressed through the suffixes. Unlike other languages such as English where special efforts were put in as described in PARC (King et al., 2003), Stanford dependency manual (M. Marneffe and Manning, 2006) etc. for defining the set of relations, we are fortunate to have a well defined set of relations for Sanskrit described in traditional grammar books. All these relations have been compiled and classified under the two broad headings viz. inter sentential and intra sentential relations (Ramakrishnamacharyulu, 2009). This work provided a starting point for developing guidelines for annotation of Sanskrit texts at kāraka level and also for the development of an automatic parser for Sanskrit. These tags were further examined from the granularity point of view and a subset of 31 tags was chosen for annotation as well as for developing the parser (Kulkarni and Ramakrishnamacharyulu, 2013). The criterion used for deciding the granularity is simple. If one can tell one relation from the other purely on the basis of syntax or morphology, then the two relations were treated as distinct.

A generative grammar of any language provides rules of generation. For analysis, we require a mechanism by which we can reverse these rules. The reversal in general may not always be deterministic. This problem of non-determinism was well recognised by the mīmāṁsakas (exegesists) who proposed three conditions viz. ākāṅkṣā (expectancy), yogyatā (mutual compatibility), and sannidhi (proximity) as necessary for proper verbal cognition. The ākāṅkṣā is the syntactic expectancy a word has in order to co-relate to the other. This expectancy may be either mutual or one-way. Yogyatā helps in ruling out solutions which satisfy syntactic expectancy but which are not meaning-compatible. Sannidhi is defined as an utterance of words without any gap. The words with mutual expectancy should not be separated by other words. The condition of not allowing separation is only a necessary condition in the process of śābdabodha 'verbal cognition'. We have implemented a parser that uses two constraints viz. ākāṅkṣā and sannidhi. Implementing yogyatā requires a semantically rich lexicon. A study was undertaken to understand the structure of a Sanskrit thesaurus "Amarakośa" and comparison of its synonyms with those of Sanskrit Wordnet (Nair and Kulkarni, 2010; Nair, 2011). The implementation of yogyatā is postponed till a reasonable size of semantically rich lexicon is available.

The problem of parsing is modelled as finding a directed Tree from a Graph where the nodes correspond to the words in a sentence and the edges correspond to the relations between them. Ākāṅkṣā postulates the possible relations. Together with sannidhi it also imposes certain constraints. These constraints are solved using a generic constraint solver Minion [11]. The parses are ranked associating costs to various relations. Detailed description of the parser is available in the earlier publication (Kulkarni et al., 2010). After getting a parse, the sharing of arguments, clausal relations and the anaphora resolution indices are marked.

Let us now describe the communication between Saṁsādhanī and Heritage engine. Both platforms provide a segmenter as well as a parser. The segmenter of Saṁsādhanī (Kumar et al., 2010) works

[11] `http://minion.sourceforge.net`

in two stages. In the first stage it generates all possible segments following the sandhi rules, and in the second stage it validates the splits by a morphological analyser, throwing out almost 90% of the splits it has generated in the first stage. This results in slowing down the process. Heritage splitter on the other hand splits only if the split is morphologically valid, and thus is an efficient implementation. The parser of Saṁsādhanī is a full fledged one which handles various kinds of relations among words, sharing of arguments, and also anaphora resolution to some extent. The shallow parser of Heritage (Huet, 2007) uses mostly minimum information of transitivity of a verb as a sub-categorisation frame and models it as a graph-matching algorithm. In order to benefit from each other's work, we worked towards plugging-in these modules in each other's engine. Though the two systems were developed using different programming environments, their communication through UNIX pipes made their composition transparent. We just had to agree on the input and output specifications for the modules. Here we faced the linguistic challenges. These linguistic challenges owe to different systems being followed for the morphological analysis. Saṁsādhanī follows the Pāṇinian system while Heritage precompiles certain derivations into paradigm tables in the Western manner. This leads to differences in stems of the words in certain cases. For example, Heritage takes *aham* as the stem for the first person pronoun, while for Saṁsādhanī the stem is *asmad*. Similarly, in case of adjectives, Saṁsādhanī treats the feminine, neuter and masculine stems as different, whereas Heritage derives all the forms from the same stem. Another problem was mapping the verbal roots, since there are several classifications of verbs (*dhātupāṭha*s), and there are various views concerning verbal forms in *-yati* (roots of class 10 vs denominative verbs vs causative conjugations). The sole dhātupāṭha available in an exploitable electronic form is the *Mādhavīya dhātuvṛtti*[12]. An effort is on to link various *dhātuvṛttis* through the canonical index of verbal roots and canonical meanings (Shailaja and Kulkarni, 2013). Meanwhile, the number of primary verbal roots being a closed set, these roots were mapped manually based on their meanings. Then there is a problem of homonymy index. Saṁsādhanī uses Apte's Practical Sanskrit-Hindi dictionary. So there is a need to match the head entries of the Heritage Sanskrit-French dictionary with those of Apte's Sanskrit Hindi dictionary. In the current parser, since it does not attempt to disambiguate the words, the homonymy index is just ignored. The effort described in section 4 above may be repeated with Apte's dictionary to map the homonymy indices.

7 Annotation Tools

Syntactic research on Sanskrit is hindered by the fact that there does not exist a morphologically and syntactically tagged corpus of Sanskrit texts. Despite the large number of digitized texts now available at various websites, and the significant number that have been partially or fully sandhi-analysed, only relatively small portions of a small number of texts have been morphologically tagged. In June, Ralph Bunker, Gérard Huet, and Peter Scharf collaborated to create an interface that allows machine-assisted human-validated tagging. Sentences in digital texts in the Sanskrit Library (SL) are fed to the SH parser. The results of possible solutions are summarized in a user-friendly single-page interface that allows a Sanskrit scholar to select among presented words, stems, and morphological tags. As elements are validated, competing solutions are deprecated in the solutions summary and unique tags are automatically copied to the candidate solution. The interface also allows the scholar to edit and resubmit the sentence for re-analysis by the SH-parser, to edit, add, or delete words, stems, and tags, or to tag the sentence manually. Inflectional morphology tagsets designed independently by Peter Scharf and by Gérard Huet in categories familiar to Europeans and by Amba Kulkarni in Pāṇinian terms were mutually mapped and rendered convertible. A convenient dialogue box for tag construction ensures ease and validity of tagging. Results are saved in XML

[12] `http://sanskrit1.ccv.brown.edu/Sanskrit/Vyakarana/Dhatupatha/index2.html`

files that can be reviewed with the same interface. The project contracted IIT Bombay to engage two post-doctoral Sanskrit researchers to utilize the interface to tag digital texts.

We built the webpage for each sentence by parsing the HTML output of the SH parser and converting it to the SL format. This procedure permitted immediate integration of the SH and SL resources while work began to develop the next version of the SH parser with summarization.

The summary mode greatly improves the robustness of the SL/SH interface, specially for long complicated sentences, where the large number of potential solutions could possibly choke the server. Both the SH and SL servers are installed locally on machines running Ubuntu Linux or Mac OSX. The SH parser uses a locally installed Apache server. The SL webpages are served using a Tomcat server installed on the assistant's machine. The SL sentence webpage performs most of its work in Javascript in order to enhance responsiveness of the page. Installing the servers locally allows the assistants to tag the sentences independent of Internet access.

In the meantime, Amba Kulkarni has designed XML output of her parser for integration with the next version of the SL tagging interface, and Pawan Goyal and Gérard Huet have designed a new interactive HTML interface that summarizes the union of all solutions returned by the SH parser. This new interface presents a summary of possible sentence segmentations with each possible word positioned at the point where it begins beneath the sentence. As users validate particular words, inconsistent segmentations are discarded. When the number of parsing solutions is sufficiently low, the user can switch to explicit listing of solutions, allowing the semi-automatic selection of ambiguous morphological features.

8 Conclusions and Future Work

Decades of independent digitization of Sanskrit texts and lexical resources and development of Sanskrit linguistic resources have culminated in the collaborative efforts to develop the automated processing of Sanskrit text described in this paper. The emphasis of this collaboration over the past several months has been to build an annotation toolkit to help linguists create a morphologically tagged corpus. Due to the paucity of resources for the Sanskrit language, creating a large-scale annotated corpus is a prerequisite to the use of statistical methods for developing high-performance and robust Sanskrit text analysers. Since it is expensive to produce annotated corpora by hand, our efforts are directed towards reducing the annotation labor by building tools to permit semi-automated annotation. As discussed, the platform produced by the collaboration includes a state-of-the-art user interface with interactions between digital libraries and various text analysers.

The annotated corpus will help us explore the use of statistical methods to enhance our existing models for text analysis. Oliver Hellwig (Hellwig, 2009b,a) has already demonstrated promising results by using various statistics from inflected form n-grams to build a POS tagger. The morphologically tagged corpus under construction will allow the extended use of statistics on more abstract linguistic features. Since the corpus used as the source for annotation consists of complete texts that preserve the context of sentences within their discourse structures, the tagged corpus will be potentially helpful to pursue research towards discourse-level dependency parsing, including anaphora resolution and ellipsis determination.

References

Bharati, A., Chaitanya, V., and Sangal, R. (1995). *Natural Language Processing. A Paninian Perspective*. Prentice-Hall of India, New Delhi.

Cardona, G. (1988). *Pāṇini: his work and its traditions*. Motilal Barnasidass.

Eilenberg, S. (1974). *Automata, Languages, and Machines, volume A*. Academic Press.

Gillon, B. S. (1995). Autonomy of word formation: evidence from Classical Sanskrit. *Indian Linguistics, 56 (1-4)*, pages 15–52.

Gillon, B. S. (2009). Tagging classical Sanskrit compounds. In Kulkarni, A. and Huet, G., editors, *Sanskrit Computational Linguistics 3*, pages 98–105. Springer-Verlag LNAI 5406.

Goyal, P. and Huet, G. (2013). Completeness analysis of a Sanskrit reader. In *Proceedings, 5th International Symposium on Sanskrit Computational Linguistics*. D. K. Printworld(P) Ltd.

Hellwig, O. (2009a). Extracting dependency trees from Sanskrit texts. In Kulkarni, A. and Huet, G., editors, *Sanskrit Computational Linguistics 3*, pages 106–115. Springer-Verlag LNAI 5406.

Hellwig, O. (2009b). SanskritTagger, a stochastic lexical and POS tagger for Sanskrit. In Huet, G., Kulkarni, A., and Scharf, P., editors, *Sanskrit Computational Linguistics 1 & 2*, pages 266–277. Springer-Verlag LNAI 5402.

Huet, G. (2001). From an informal textual lexicon to a well-structured lexical database: An experiment in data reverse engineering. In *Working Conference on Reverse Engineering (WCRE'2001)*, pages 127–135. IEEE.

Huet, G. (2002). The Zen computational linguistics toolkit: Lexicon structures and morphology computations using a modular functional programming language. In *Tutorial, Language Engineering Conference LEC'2002*.

Huet, G. (2003). Towards computational processing of Sanskrit. In *International Conference on Natural Language Processing (ICON)*.

Huet, G. (2004). Design of a lexical database for Sanskrit. In *Workshop on Enhancing and Using Electronic Dictionaries, COLING 2004*. International Conference on Computational Linguistics.

Huet, G. (2005). A functional toolkit for morphological and phonological processing, application to a Sanskrit tagger. *J. Functional Programming*, 15,4:573–614.

Huet, G. (2006). *Themes and Tasks in Old and Middle Indo-Aryan Linguistics, Eds. Bertil Tikkanen and Heinrich Hettrich*, chapter Lexicon-directed Segmentation and Tagging of Sanskrit, pages 307–325. Motilal Banarsidass, Delhi.

Huet, G. (2007). Shallow syntax analysis in Sanskrit guided by semantic nets constraints. In *Proceedings of the 2006 International Workshop on Research Issues in Digital Libraries*, New York, NY, USA. ACM.

Huet, G. (2009). Formal structure of Sanskrit text: Requirements analysis for a mechanical Sanskrit processor. In Huet, G., Kulkarni, A., and Scharf, P., editors, *Sanskrit Computational Linguistics 1 & 2*. Springer-Verlag LNAI 5402.

Huet, G., Kulkarni, A., and Scharf, P., editors (2009). *Sanskrit Computational Linguistics 1 & 2*. Springer-Verlag LNAI 5402.

Huet, G. and Razet, B. (2006). The reactive engine for modular transducers. In Futatsugi, K., Jouannaud, J.-P., and Meseguer, J., editors, *Algebra, Meaning and Computation, Essays Dedicated to Joseph A. Goguen on the Occasion of His 65th Birthday*, pages 355–374. Springer-Verlag LNCS vol. 4060.

Huet, G. and Razet, B. (2008). Computing with relational machines. *ICON'2008 tutorial,* `yquem.inria.fr/~huet/PUBLIC/Pune_tutorial.pdf`.

Joshi, S., Roodbergen, J., and Akādemī, S. (2004). *The Aṣṭādhyāyī of Pāṇini with Translation and Explanatory Notes*. Number v. 11 in The Aṣṭādhyāyī of Pāṇini. Sahitya Akademi.

Karp, D., Schabes, Y., Zaidel, M., and Egedi, D. (1992). A freely available wide coverage morphological analyser for english. In *Proceedings of Coling-92, Nantes, August 23–28, 1992*, pages 950–955.

King, T. H., Crouch, R., Riezler, S., Dalrymple, M., and Kaplan, R. (2003). The PARC 700 dependency bank.

Kiparsky, P. (2009). On the architecture of Pāṇini's grammar. In Huet, G., Kulkarni, A., and Scharf, P., editors, *Sanskrit Computational Linguistics 1 & 2*. Springer-Verlag LNAI 5402.

Kulkarni, A. and Huet, G., editors (2009). *Sanskrit Computational Linguistics 3*. Springer-Verlag LNAI 5406.

Kulkarni, A. and Kumar, A. (2011). Statistical constituency parser for Sanskrit compounds. In *Proceedings of ICON 2011*. Macmillan Advanced Research Series, Macmillan Publishers India Ltd.

Kulkarni, A. and Kumar, A. (2013). Clues from Aṣṭādhyāyī for compound type identification. In Kulkarni, M., editor, *Proceedings of the International Sanskrit Computational Linguistics Symposium*. D. K. Printworld(P) Ltd.

Kulkarni, A., Pokar, S., and Shukl, D. (2010). Designing a constraint based parser for Sanskrit. In Jha, G. N., editor, *Proceedings of the 4th International Sanskrit Computational Linguistics Symposium*. Springer-Verlag LNAI 6465.

Kulkarni, A. and Ramakrishnamacharyulu, K. V. (2013). Parsing Sanskrit texts: Some relation specific issues. In Kulkarni, M., editor, *Proceedings of the 5th International Sanskrit Computational Linguistics Symposium*. D. K. Printworld(P) Ltd.

Kulkarni, A. and Shukl, D. (2009). Sanskrit morphological analyser: Some issues. *Indian Linguistics*, 70(1-4):169–177.

Kumar, A. (2012). *An automatic Sanskrit Compound Processing*. PhD thesis, University of Hyderabad, Hyderabad.

Kumar, A., Mittal, V., and Kulkarni, A. (2010). Sanskrit compound processor. In Jha, G. N., editor, *Proceedings of the 4th International Sanskrit Computational Linguistics Symposium*. Springer-Verlag LNAI 6465.

Kumar, A., SheebaSudheer, V., and Kulkarni, A. (2009). Sanskrit compound paraphrase generator. In *Proceedings of ICON 2009*.

M. Marneffe, B. M. and Manning, C. D. (2006). Generating typed dependency parses from phrase structure parses. In *The fifth international conference on Language Resources and Evaluation, LREC 2006, Italy*.

Mittal, V. (2010). Automatic sanskrit segmentizer using finite state transducers. In *Proceedings of the ACL 2010 Student Research Workshop*, pages 85–90, Uppsala, Sweden. Association for Computational Linguistics.

Nair, S. (2011). *The Knowledge Structure in Amarakośa*. PhD thesis, University of Hyderabad, Hyderabad.

Nair, S. and Kulkarni, A. (2010). The knowledge structure in Amarakośa. In Jha, G. N., editor, *Proceedings of the 4th International Sanskrit Computational Linguistics Symposium*. Springer-Verlag LNAI 6465.

Ramakrishnamacharyulu, K. V. (2009). Annotating the Sanskrit texts based on the śābdabodha systems. In *Proceedings of 3rd International Sanskrit Computational Symposium*. Springer-Verlag LNAI-5406.

Rāmapriya, B. V. and Saumyanārāyaṇa, V. (2001). Saṅgaṇakayantre nyāyaśāstrīyaśābdabodhaḥ. *Journal of Foundation Research*, VI(1–2):61–68.

Razet, B. (2009). *Machines d'Eilenberg Effectives*. PhD thesis, Université Denis Diderot (Paris 7).

Scharf, P. (2009). Levels in *Pāṇini*'s *Aṣṭādyāyī*. In Kulkarni, A. and Huet, G., editors, *Proceedings, Third International Symposium on Sanskrit Computational Linguistics*, volume LNAI 5406, pages 66–77. Springer.

Scharf, P. and Hyman, M. (2009a). Enhancing access to primary cultural heritage materials of india. In Govindaraju, V. and Setlur, S., editors, *Guide to OCR for Indic Scripts: Document Recognition and Retrieval*, pages 237–247, London; Dordrecht; Heidelberg; New York. Springer-Verlag.

Scharf, P. and Hyman, M. (2009b). *Linguistic Issues in Encoding Sanskrit*. Motilal Banarsidass, Delhi.

Shailaja, N. and Kulkarni, A. (2013). Comparative study of pāṇinīya dhātuvṛttis. In Kulkarni, M., editor, *Proceedings of the 5th International Sanskrit Computational Linguistics Symposium*. D. K. Printworld.

Sharma, A., Deshpande, K., and Padhye, D. (2008). *Kāśikā: A Commentary on Pāṇini's Grammar*. Sanskrit Academy series. Sanskrit Academy, Osmania Universtiy.

Shukla, P., Shukl, D., and Kulkarni, A. (2010). Vibhakti divergence between Sanskrit and Hindi. In Jha, G. N., editor, *Proceedings of the 4th International Sanskrit Computational Linguistics Symposium*. Springer-Verlag LNAI 6465.

Whitney, W. D. (1997). *Roots, Verb-forms and Primary Derivatives of the Sanskrit Language*. Motilal Banarsidass, Delhi. (1st edition 1885).

Understanding the Performance of Statistical MT Systems: A Linear Regression Framework

Francisco Guzman *Stephan Vogel*
Qatar Computing Research Institute
Qatar Foundation
fguzman@qf.org.qa svogel@qf.org.qa

ABSTRACT
We present a framework for the analysis of Machine Translation performance. We use multivariate linear models to determine the impact of a wide range of features on translation performance. Our assumption is that variables that most contribute to predict translation performance are the key to understand the differences between good and bad translations. During training, we learn the regression parameters that better predict translation quality using a wide range of input features based on the translation model and the first-best translation hypotheses. We use a linear regression with regularization. Our results indicate that with regularized linear regression, we can achieve higher levels of correlation between our predicted values and the actual values of the quality metrics. Our analysis shows that the performance for in-domain data is largely dependent on the characteristics of the translation model. On the other hand, out-of domain data can benefit from better reordering strategies.

TITLE AND ABSTRACT IN ANOTHER LANGUAGE

Modelos Lineales para el Análisis del Desempeño de la Traducción Automática

En este documento presentamos una metodología para el análisis del desempeño de los sistemas de traducción automática. Utilizamos modelos lineales multivariados para determinar el impacto que diversas variables tienen en la calidad de las traducciones. En este estudio se asume que las variables que más contribuyen a predecir la calidad de las traducciones, son determinantes para entender las diferencias entre buenas y malas traducciones. Nuestros resultados demuestran que usando regresión lineal penalizada, se pueden obtener altos índices de predicción de calidad de traducción. Un análisis detallado revela que el desempeño de los sistemas de traducción frente a datos *in-domain* dependen en gran medida de las características de nuestros modelos de traducción. En contraste, la traducción de documentos *out-of-domain* está fuertemente ligada a las estrategias de reodenamiento que se utilicen.

KEYWORDS: Statistical machine translation, translation quality prediction, system performance analysis.

KEYWORDS IN L_2: Traducción automática estadística, calidad de traducción, Análisis de desempeño.

Proceedings of COLING 2012: Technical Papers, pages 1029–1044,
COLING 2012, Mumbai, December 2012.

1 Introduction

In their strive to improve machine translation, researchers constantly introduce new models and features; new training and decoding algorithms, or experiment with variations and combination of existing approaches. This is often done based on experience and intuition. The success or failure of an algorithm is based on trial and error, guided by the end-to-end automatic translation quality metrics as a measure of performance. However, little attention is paid to an equally important task: understanding how the different components in the complex SMT systems affect performance. Such knowledge could allow researchers and engineers to address specific weaknesses of their systems. For example, knowing that coverage is an issue, a team might decide to increase the amount of in-domain training data to match their specific needs. In this paper, we develop a methodology to perform this type of analysis. We propose a framework that uses linear models to identify the variables that most contribute to predict the performance of a translation system when dealing with a specific translation task. Our assumption is that variables that most contribute to predicting translation performance are the key to improving performance. Detecting them will provide leverage to design better translation systems.

In our paper, we employ linear models to predict translation quality. Linear regression is a widely known and applied technique. It models the response variable y as a weighted linear combination of a feature vector X. For instance:

$$y = \theta^T X + \varepsilon \tag{1}$$

where θ represents the parameter vector for the regression model and ε is the model error. On a multivariate regression model, Equation 1 represents an hyperplane that minimizes the error ε.

In this paper, we analyze the output of several translation systems that use the same decoder, but differ in the alignment models that they use to build their respective translation model. We use different characteristics of their corresponding translation hypotheses and translation models as input features X to predict their translation performance y in terms of three popular automatic translation quality metrics: BLEU (Papineni et al., 2002), METEOR (Denkowski and Lavie, 2011) and TER (Snover et al., 2006). We use a regularized regression model to estimate the parameters of our prediction model. We use Spearman's rank correlation, Pearson's correlation and RMSE to evaluate the fitness of the regression models estimated for two different domains (News, Proceedings) and a mixed-domain, general model.
Our results indicate that using a regularized linear regression, we can achieve high levels of correlation between our predicted values and the actual values of the quality metrics. We take a closer look at the most important features according to the regression coefficients and discuss the results. We find that many features are shared as the most important predictors across the different objective functions (BLEU, Meteor, TER). Our analysis shows that the performance for in-domain data is largely dependent on the characteristics of the translation model. On the other hand, performance in out-of-domain tasks relies on characteristics such as reordering and alignment distortion. Note, however that the results are dependent on the specific datasets analyzed as well as the features included in the model. Our goal is not to provide a one-hat-fits-all set of recommendations that would address every possible scenario, but rather to provide an analytical framework that researchers can apply to their own systems.

2 Related work

The work presented in this paper is related to previous analysis done in the past few years. For instance, the correlation between characteristics of the translation model and the automatic quality metrics has previously been addressed. Lopez and Resnik (2006) make a study of different phrase-based translation model (TM) features and their impact translation quality. They also analyze variations in the translation search space of the decoder by having alignments of gradually degraded quality. On the other hand, Birch et al. (2008) study different language-pair characteristics and use them as predictors of BLEU translation quality using linear regression. Furthermore, Pado et al. (2009) use linear models to build a higher-level translation quality metric that uses features from other established metrics (e.g. BLEU, METEOR, TER) as well as Textual Entailment features (Dagan et al., 2006) and that achieves higher correlation with human judgements. However, multivariate regression, has not been used as a tool to predict translation quality based on the characteristics of the translation model.

Others have focused on identifying characteristics of the word alignments upon which these models have been built. Fraser and Marcu (2007) study how alignment quality (AER) is related to its translation quality relative BLEU. As a result, they proposed a modified version of AER to increase the correlation between alignment quality and translation performance. Lambert et al. (2009, 2010) analyze how alignment characteristics correlate with translation quality. They analyze the effect of the number of links of different types of alignments including its repercussions on the size of phrase tables and the ambiguity of the translation model. They also propose new structural metrics for alignments such as link length, distortion and crossings. In this study, we also include alignment features to characterize our translation models.

A closely related topic to this study is the task of Quality Estimation for Machine translation (Specia et al., 2009; Specia, 2011), where sentence-level prediction models are used to estimate quality of Machine Translation output. In that task, researchers have many sources of information available (typically: model scores, automatic metric scores, post-editing effort scores, etc.) and the goal is to provide a model that reliably is able to distinguish good from bad translations. The proposed work differs from Quality Estimation in two aspects: First, here we are interested in contrasting the output from several translation models, to be able to learn their shared features that help predict better quality scores. Second, we are not interested in performing a local estimation (at a sentence level) but at a document level.

Finally, recent work by Devlin and Matsoukas (2012), focuses in using variation in traits, or hypothesis characteristics to generate alternative hypotheses that are later used for system combination. In their work, they use null words, reordering, ngram-frequency, hypothesis length, among other features. In our view, the current study is complementary to that work, given that our framework allows to detect important features of "traits" which could serve as input a trait-based hypothesis selection system.

Summarizing, in this paper we propose a framework for the analysis of Machine Translation performance in terms of characteristics of the translation models and translation hypotheses. The main difference to previous research relies on the use of multivariate linear models to determine the impact of a wide range of features in translation performance. While we present results only for a phrase-based approach, this type of analysis can be applied to different approaches and language pairs. The systematic identification of important features, as proposed in this study, can help to focus development efforts in in critical areas which will help to improve translation performance.

3 Problem formulation

Our task is to find the parameter vector θ that minimizes the squared error of the fitted function $\theta^T X$. For instance, if we define the error as function of the parameter vector:

$$\varepsilon(\theta) = y - \theta^T X \tag{2}$$

then our learning task is to find $\hat{\theta}$ that minimizes the squared error:

$$\hat{\theta} = \arg\min_{\theta_i} \frac{1}{m} \varepsilon(\theta_i)^T \varepsilon(\theta_i) \tag{3}$$

where m represents the number of training examples in our training set. In other words, we want to find the weights for each of the input features X such that their weighted sum minimizes the error across the training set. This serves as a predictor function for our dependent variable (y).

In our scenario, the dependent variable is automatic metric based translation quality, in terms of BLEU, METEOR or TER. The independent variables are features extracted from the translation hypotheses and their corresponding translation models. Below, we describe each of the components of this learning problem. First, we describe the dependent and independent variables. Then, we introduce the algorithms employed for estimating the parameters.

3.1 Features and objective functions

In linear regression, we are interested in building a function $f(\theta, X) \rightarrow y$ to predict y. In our specific application, we are interested in predicting translation quality. Thus y represents a translation quality score. Since most evaluation metrics operate at a document level (i.e. they aggregate statistics of several translations) we need to adapt our formulation to be able to handle this data.

Let (s_i, t_i^M, r_i) be a source document (s_i), its MT translation (t_i^M) given a model M and its human reference translations (r_i).

We define $y_i = g(t_i^M, r_i) \rightarrow \mathbb{R}$, where $g(\cdot,\cdot)$ represents an automatic translation quality metric that takes as input a translated document t_i^M and its corresponding reference translation r_i .

Similarly, we need to use aggregated features over the full document set. As such, our features take into account (s_i, t_i^M, M). We divide these into two types of features: the translation hypothesis features (i.e. they take into account only t_i) and the translation model features (that take into account s_i and M to compute their value. Below, we introduce each group briefly.

3.1.1 Translation model features

In this study, we are interested in the characteristics of the part translation model M that is visible to the decoder when translating. Therefore, for each input document s_i we extract features from a submodel M_i s.t. $M_i \subset M$. In other words, we only keep entries of M that had a match in the corresponding sub-document s_i. In practice, this is achieved by filtering the phrase-table to the specific document s_i. The features measured are the following:

Model entropy For each of the translation features (inverse and direct, phrasal and lexical translation probabilities) in the baseline phrase-based models, we used a variation of

the conditional entropy, assuming a uniform distribution over x (i.e. $p(x) = 1/|X|$), For instance, the entropy for the inverse phrasal probability $p(f|e)$ is:

$$H_p(F_i|E_i) = 1/|E_i| \sum_{e \in E_i} \sum_{f \in F_i} p(f|e) \log p(f|e) \quad (4)$$

Translation model size For each phrase-table, we measure the number of entries (log), as well as the number of source and target singletons.

Alignment density variables We use the per-phrase pair number of links (Ayan and Dorr, 2006), source and target gaps (Guzman et al., 2009), averaged over the phrase-table.

Alignment distortion variables We use the per-phrase pair number of link-crossings (Lambert et al., 2009), relative link distortion, and a new distortion feature we call diagonality, which is the absolute value of Pearson's correlation (from 0 to 1) of the positions in the source and target words of an alignment.

3.1.2 Translation hypothesis features

These types of features include the translation cost for each of the features used in the Moses phrase-based decoder (Koehn et al., 2007). These include:

Translation feature costs The per-phrase cost for each of the translation probability features in the translation model averaged over the translation set t_i. We used the baseline translation features in the phrase-based model (Koehn et al., 2003): inverse and direct phrasal translation probabilities and inverse and direct lexical probabilities.

Lexicalized reordering costs The per phrase cost for the distance-based reordering feature and each of the three different orientations (mono, swap, discontinuous) in a bidirectional setting averaged over the translation set t_i.

Language model cost The per-word language model cost for each translation hypothesis averaged over the translation set t_i.

Additionally, we include word-alignment based features:

Word aligment variables Similarly to the translation model features, we used aligment density and alignment distortion variables, averaged over the number of phrases used.

3.2 Learning algorithm

There are several algorithms to estimate the multivariate regression parameters (regression coefficients). Here, we use a numerical optimization method with $L2$ regularization, which allow for the inclusion of many types of variables, regardless of collinearity. For the linear regression problem, the regularized cost function is defined as follows:

$$J(\theta) = \frac{1}{m} \varepsilon(\theta)^T \varepsilon(\theta) - \lambda \theta^T \theta \quad (5)$$

There are several minimization methods available for solving this problem. In our experiments, we used a Polack-Ribiere based conjugate gradient minimization routine by Carl Rasmussen[1]. We optimized the λ regularization constant to minimize cost on a cross-validation set.

3.3 Framework architecture

In order to implement this analysis framework, these are the steps that need to be followed.

1. **Translation model generation**
   The first step consists in training the MT system (or systems) of interest, using a standard training pipeline (e.g. Moses). Additionally, use a standard tuning metric (e.g. MERT) to assign optimal parameter weights for each system.

2. **Dataset selection**
   The next step consists in partitioning a dataset (or multiple datasets) for analysis into equally sized subdocuments (source and target parts). For a more diverse sample, we recommend using subdocuments extracted from different datasets.

3. **Hypothesis generation**
   Translate each of the translated subdocuments using the MT systems of interest. Additionally, generate the first-best feature-cost information (e.g. with the `-n-best-list` parameter in Moses).

4. **Feature extraction**
   Extract the relevant features corresponding to each subdocument. For the translation model features, filter each of phrase-tables of interest to each of the input subdocuments and extract the phrase-table level features. For the hypothesis features, use the extracted feature costs from the n-best list. For the translation quality, use the generated hypotheses and evaluate them against the given references.

5. **Regression**
   Use the generated data to fit the translation quality regularized regression model with an off-the-shelf ML package. Using zero-mean normalization of features is highly recommended, because it facilitates the interpretation of the results. Cross-validation is recommended to set the optimal regularization penalty.

6. **Analysis**
   Finally, obtain the regression weights and analyze them. To facilitate this task, we recommend using only the top-k largest (in magnitude) coefficients.

4 Experimental setup

For our experiments, we used a phrase-based system (Moses). For training, we used the Spanish-English portion of Europarl v5, the United Nations, and the news-commentary datasets as provided for the WMT2010 competition. The statistics of this data are shown in table 1.

[1] available at `http://www.gaussianprocess.org/gpml/code/matlab/doc/`

	RAW			PP		
Set	Lines	Tok	Voc	Lines	Tok	Voc
Spanish						
EU	1.7M	43.1M	393.1K	1.4M	35.1M	140.0K
NC	98.6K	2.5M	123.1K	90.0K	2.3M	59.2K
UN	6.2M	190.6M	1.4M	4.9M	129.8M	330.1K
total	8.0M	236.2M	1.6M	6.4M	167.2M	387.9K

Table 1: Statistics for Raw and preprocessed data for Europarl (EU), News Commentary (NC) and UN training data. We present the total number of training examples (lines), number of tokens (tok) and the vocabulary size (voc).

4.1 Translation model training

The data was lowercased and tokenized with the standard preprocessing toolkit available in Moses. To introduce variation in our translation models, we used different types of alignments. The aligners used for these systems were a discriminative aligner (DWA) (Niehues and Vogel, 2008) with different density thresholds (0.4, 0.5, 0.6, 0.7) to have a variety of dense and sparse alignments. The DWA aligner was trained using hand aligned data from the EPPS (Lambert et al., 2006) dataset. Additionally, we used the symmetrized GIZA++ alignments using the heuristics grow-diag, grow-diag-final and grow-diag-final-and. While these variations in alignments might seem minor, in reality, as previously observed by (Guzman et al., 2009; Niehues et al., 2010) they can have a large impact on the characteristics of the translation model. In total we experimented with 7 different translation models.

Each of the systems was tuned using MERT on the WMT news2008 set.

4.2 Feature generation

For our regression training, we translated and analyzed the quality of different documents. We used a variety of different test-sets publicly available for the Spanish-English translation task for the WMT competitions[2]. The description of the different datasets is presented in Table 2.

4.2.1 Sub-document sampling

To better appreciate the effect of a translation model into translation quality, we split each dataset into used several sub-documents long enough to provide accurate translation statistics (e.g. ngram counts for BLEU), but short enough to allow us to appreciate the differences between different translation models. Sub-document splitting is a known technique that has been used previously for confidence interval estimation (Koehn, 2004).
In our study, we chose a slightly more conservative sub-document size of 100 translation sentences to get smoother results. For our experiments we used only 4 subdocuments (one hundred sentences each) from each of the 9 datasets presented in Table 2. We restricted to 4 samples to ensure that each dataset was equally represented (some datasets are shorter than others). We obtained translations for each of the 7 different translation systems. This resulted in a total set of (9x7x4) 252 different training instances for our regression models (for the cross domain set).

[2]Data can be obtained directly from `http://www.statmt.org/wmt11/`

Set id	Description	Domain	Sentences	Coverage(%)
AC	Acquis Communautaire	Legal	4107	95.68
NC07	News Commentary	News commentary	2007	98.12
NC08	News Commentary	News commentary	2028	98.20
SC09	News System Combination	News/Other	502	96.59
NW09	News Test 2009	News	2525	92.83
NW10	News Test 2010	News	2489	92.49
WMT06	Europarl Test 2006	Proceedings	2000	98.60
WMT07	Europarl Test 2007	Proceedings	2000	98.83
WMT08	Europarl Test 2008	Proceedings	2000	98.66

Table 2: Description of the different datasets used in this study. We present the number of the original sentences as well as the percentage of coverage w.r.t. to our training corpus described in Table 1

We also performed a domain-specific (news, proceedings) training, and validation of our linear models to discover which variables were most important in describing performance for each of those genres.

4.3 Regression training

We divided our experiments into three different tasks: learn predictive models for each of the objective functions (BLEU, METEOR and TER). Additionally, to showcase how this technique can be applied to a variety of different scenarios, and to get a better insight of which variables affect different translation tasks, we computed three types of domain-dependent models for each objective function. The first type is a cross-domain or general model, which predicts translation performance regardless of the translation task. For this prediction task, we used all available data from the cross-domain training samples. The second type of prediction model is for out-of-domain news-based data. For training this type of model, we only used the portion of documents that are news related (NW09, NW10). Finally we also present the results for in-domain proceedings data. For these models we used the parliamentary proceedings (WMT06, WMT07, WMT08).

4.4 Measuring regression performance

One caveat of regression, is that given enough features, it can find the appropriate weights to fit the training set. Regularization helps in part to alleviate over-fitting. However, we performed several tests to ensure that over-fitting was not a problem. Each evaluates a different aspect of our predictions. First, we obtain the root mean square error (RMSE), which gives us an idea of the distance between our predictions and the real value of the translation quality metrics we are approximating. RMSE is given by:

$$RMSE = \sqrt{\sum_{j}^{m}(y_j - \theta^T X_j)^2} \tag{6}$$

The other two metrics we used for comparison are Spearman's rho, also known as rank correlation and Pearsons's correlation. These metrics allows us to measure to which degree the

	PROC			NEWS			GENERAL		
	BLEU	MET	TER	BLEU	MET	TER	BLEU	MET	TER
Spearman	0.71	0.72	0.70	0.80	0.80	0.78	0.84	0.82	0.77
Pearson	0.70	0.74	0.65	0.79	0.89	0.84	0.84	0.83	0.78
RMSE	1.38	0.71	1.91	1.05	0.69	1.51	3.04	2.06	3.90

Table 3: Leave-one-out crossvalidation results for the regression algorithm. We present three different performance metrics: Sperman's rank correlation, Pearson correlation and RMSE. We trained different regression models to fit BLEU, Meteor and TER across three different domains (general, news and proceedings.)

values and rank (order) of our predicted variables are correlated to the values and ranks of the actual values. In other words, how monotonic is the relationship between the predicted and real values. The formula for the correlation is:

$$\rho = \frac{\sum_j^m (x_j - \bar{x})(y_j - \bar{y})}{\sqrt{\sum_j^m (y_j - \bar{y})^2 \sum_j^m (x_j - \bar{x})^2}} \tag{7}$$

where each x_j and y_j are values (for Pearson's) or ranks (for Spearman's).

4.4.1 Cross-validation

Given the reduced amount of data we had (252 training instances), we opted to use leave-one-out cross validation instead of having separate training and test sets. Thus, we trained our models in all-but-one instances, and used the remaining one for testing purposes. We repeated this process for each of our training instances. For our parameter analysis, used the average parameter values of all iterations of our cross-validation.

In Table 3 we observe the results for each of the document sets (general, news, proceedings) and for the each of the three translation quality metrics used.

First, note that correlation (both rank and real-valued) we observe large correlation values (> 0.65). This indicates that our regression algorithm is doing a good job in predicting translation quality using the available information. However, notice that we have a larger correlation values for the general set. This hints that our algorithm can benefit from having additional training data for making more accurate estimations. This is despite the fact that the general set has the highest variance, which naturally results in a higher RMSE.

5 Analysis

By using linear regression, we find the parameter vector θ that minimizes the error between our prediction and the actual values of the metrics. By looking at the values of each specific θ_i we can deduce the relative importance of the relationship between a certain feature X_i and the quality metric y.

To simplify the analysis, we only look at the five most important features in the feature vector (according to their absolute magnitude). Given that the features were originally normalized (to

Keys	Description	BLEU	MET	TER
General				
Const	Constant term	28.57	33.81	54.15
PT4*	Translation model direct lexical entropy $lex(e\|f)$	-6.28	-3.99	7.78
FPSL*	Average length of source phrases used in first-best	5.88	5.06	-8.09
PNE*	Translation model size (options)	5.54	2.52	-5.87
PSL*	Translation model average length of source phrases	-4.42	-2.91	5.55
FPTL	Average length of target phrases used in first-best		-2.37	3.90
Proceedings				
Const	Constant term	28.60	33.83	55.47
FD0*	Distance-based reordering cost	0.89	0.55	-0.86
FTG*	Target Gaps in the alignment of first-best	-0.69	-0.30	1.11
PT4	Translation model direct lexical entropy $lex(e\|f)$	-1.20		0.74
FD4	Right-monotone lexicalized reordering cost		-0.40	0.63
News				
Const	Constant term	21.22	29.54	60.50
FT2*	Direct lexical cost $lex(f\|e)$	1.19	0.92	-1.82
FCR*	Average number of alignment crossings of first-best	-1.64	-1.10	2.41
FD3*	Left-discontinuous lexicalized reordering cost	1.05	0.76	-1.99
PCR*	Translation model average number of alignment crossings	-0.87	-0.80	1.66
FSG	Source Gaps in the alignment of first-best		-0.73	1.16

Table 4: Most important features for the regression models for BLEU, Meteor and TER. For simplicity, only those features shared across two or more objective functions are shown. Features that are important to all metrics are marked with a $(*)$. Notice that for TER (lower is better) the weights have opposite sign than for BLEU and Meteor (higher is better)

have an average of 0 and variance of 1), the magnitude of the scores allows us to compare the strength of the relationship across features.

In Table 4 we present the top five variables for our models, for each of the different scenarios: general, proceedings and news. To further highlight the agreement of the most important features across BLEU, Meteor and TER, we only present those features which are most important to at least two of these quality metrics. We do so to facilitate the analysis.

Looking at the table, we find the most important features and their coefficients. For completeness, we also include the constant term. Simplifying, this term could be interpreted as a base or default translation quality value for any of our systems. Then, each of the feature coefficients indicates how changes in a specific feature value (assuming everything else remains unchanged) could affect performance. A positive coefficient indicates that the larger the feature value, the more gain in performance is obtained. Conversely, with a negative coefficient, the results indicate that the larger the feature value is, the larger the loss in performance. Note that BLEU and Meteor are positively defined, while TER is not. In BLEU, a higher value is considered better, while for TER, the lower the better. This is reflected by opposing signs in the corresponding feature coefficients.

5.1 General set

For the general models, one of the most important features is the direct lexical entropy of the translation model (PT4). This feature has a negative weight for translation quality which indicates that the more lexical entropy we have in our model, the worse translations we will have. In other words, given a specific source phrase f, there should be little ambiguity of which is the best translation for it.

The next important feature is the length of the source phrases used to construct our translation (FPSL). This feature indicates that the fewer phrases we need to translate a sentence, the more reliable our translations will be.

Another feature of importance is size of the translation model (PNE). This variable indicates that the more translation options we have in our translation model, the better results we will be able to obtain across metrics. This can be interpreted as the importance of coverage. Not only to reduce the number of unseen words, but also to have a good inventory of phrases.

Finally, another feature that is present is the average length of the source phrases in the translation model (PSL). In this case, the feature has a negative weight, which indicates that the longer the phrases in the translation model, the worse our translations are. This result is opposite to that of FPSL and could be interpreted in two ways: First, that this is a shortcoming of performing linear regression with two correlated features, (overshooting the weight for one variable can be compensating by setting the complementary with a negative weight). The other interpretation, which we favor, is that usually longer phrases also allows for more noise within boundaries (e.g. more source gaps in the alignment), which are an indication of a poorer model estimation.

In summary, when dealing with an unknown test set, we should aim for larger models (coverage), that are of the best quality possible (low entropy, good estimation).

5.2 News set

One interesting result comes from the news set, where the coverage is considerably lower. These results, however, need to be taken with a grain of salt, as due to the fewer number of training instances regression results (even if regularized) can be unstable.
For the news set, the direct lexical cost (FT2) is an indicator of better translations. This result is rather counter-intuitive, given that we would expect that better translations are formed using phrases with higher probability. One possible interpretation for this is that in the news set better translations require words, which are less frequently seen and which therefore are penalized with low lexical scores.

Additionally we have three juxtaposing features that regard distortion and reordering. On one hand, the number of alignment crossings at the phrase-table level (PCR) and at hypothesis level (FCR) indicate that the use of phrases that carry local reordering (more alignment crossings) has a negative effect on translation quality. On the other hand, we have a preference for hypotheses that have discontinuous reordering.
Altogether, these findings may suggest that for the news set, the phrase boundaries are not sufficient to capture the correct reordering necessary to generate good translations. We require more reordering monotonicity inside the phrases, yet to be able to move phrases as a unit for longer distances. In summary, we need to allow for more rare words to be chosen and to allow longer distance reorderings.

5.3 Proceedings set

For this type of documents, where coverage is not an issue, translations benefit from having longer distance reorderings (FD0). Also, the negative weight of variable (FTG) indicate that phrases which lack alignment support (have more target gaps in the alignment), trend to degrade the quality of the translation. Ill-defined, high entropy direct translation model lexical features will hurt performance (PT4). These results are more in line with the expected behavior of a translation system. Furthermore, as we will present later, they will allow us to perform quick fixes that will result in a gain in translation quality.

5.4 Discussion

Notice that a constant in our models is the lack of a language model component as an important predictor for translation quality. This is a result of our translations being generated using a single language model but different phrase-tables. This does not mean, however, that language models do not matter in translation, but it is merely a result of our experimental design.

As we observed, some of the results rather counter-intuitive and the interpretation is difficult, especially the case for the news test. More data needs to be analyzed in order to make more reliable estimations. However, we should note that there is a consistent agreement between features that are important for BLEU, Meteor and TER.

It is well known that the behavior of translation systems depend on many factors, not only on the techniques used for generating translation models or the choices of a specific aligner or symmetrization heuristic. While the results in our study might not be applicable to every possible scenario or language pair, our purpose is not to attain the highest level of generalization, but to encourage the use of analytic methodologies for the design of targeted systems.

6 Practical considerations

While the focus of this paper is the analysis of the most important features for translation quality, which in itself is an important goal, the application of the acquired knowledge is essential for the improvement of translation systems. Therefore, below we provide a set of pointers on how to address specific issues. Furthermore, we provide one case scenario where using this information gives positive results for the in-domain translation task.

6.1 Addressing target gaps: a practical case study

From our models for in-domain data, we observed that translation hypotheses with more target gaps have lower translation quality. Inspired by that fact, and to give the decoder more control over the gaps in the translation hypotheses we introduce a new decoding feature that takes into account the alignment gaps information and uses it dynamically at decoding time. This might be regarded as a low hanging fruit, but it showcases the potential of our methodology. As we will observe later, using such a feature enables the decoder to turn an originally liability (more gaps meant less translation quality) into an asset. The target gap decoding feature (h_{ftg}) is defined as follows:

$$h_{ftg}(e_1^I, f_1^J) = J - \sum_{j \in J} \prod_{i \in I} (1 - l(i, j)) \tag{8}$$

In our experiments, we use the weights for the new features tuned by the Minimum Error Rate optimization on the newstest2008 dataset. For comparison, we used a set of different translation

models, which were generated from the alignments we had analyzed previously (DWA-4 to DWA-7, grow-diag, grow-diag-final, grow-diag-final-and). The considered documents were both in-domain Europal test-sets (WMT06, WMT07, WMT08), the also limited domain Acquis corpus (AC) and the News test sets (NW09, NW10, SC09). The translation results are shown in Figure 1. In the image, we see the gains for each of the systems (circles) with respect to their baseline.

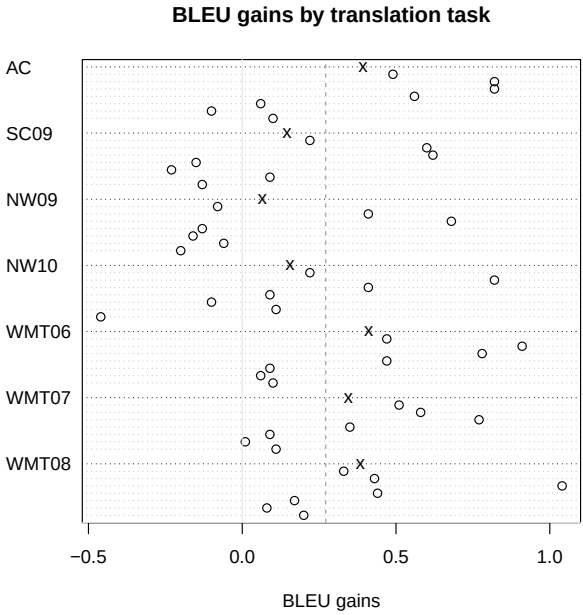


Figure 1: BLEU gains of systems using the target gap feature segmented by test-document. On the left axis we present the different document groups. On the horizontal axis, the gains of BLEU $\Delta BLEU = BLEU_{ftg} - BLEU_{bl}$. For each document set, the overall average gain is marked with an x. The general average is represented by the vertical dashed line.

Notice how for each in-domain document set, the results are consistent with what was expected, and we get significant improvements by using the feature. For instance, for the WMT06 document set, we get an improvement of 1.04 blue-points (BP) for the best-case, while the worst case increases 0.08 BP, with an average of 0.38 BP gain. For the out-of domain sets the results are mixed. For instance, for NW10, the best-case system gets an improvement of 0.8 BP, while the worse, presents a loss of 0.46 BP, with an average gain of 0.15 BP which is barely significant. On the positive side, the general average gain is 0.27 BP.

Notice also, how there is a marked difference between clusters of points. For WMT06, we observe three points with almost no gain, while other points have stronger gains. Upon analysis, we discovered that the no-gain group consists on translation models based on heuristics (grow-diag, etc.), which allow for less gaps in the target phrases than the discriminative type of alignments.

6.2 Addressing other issues

Lexical entropy has arisen as an important factor in several of our models. More entropy can lead to modeling errors. One possible alternative to address this problem is to use phrase-table filtering. Filtering techniques have already been proven effective to shrink the footprint of translation models. The challenge however is to preserve the balance between high coverage and low entropy.

Another problem is how to favor longer, yet well defined, phrases. One possibility would be to include the source entropy information for each phrase pair, as a translation feature. In this way, the decoder would be able do counterbalance length vs. entropy for the phrases.

There are many other fixes that could be suggested. Unfortunately, addressing each of them would be lengthy, and out of the scope of this paper.

7 Conclusions and future work

In this paper, we presented a framework to analyze the differences in translation quality between several translation systems based on characteristics of their translation models and hypotheses. We use multivariate linear regression to predict different translation quality metrics using a wide range of features. We measured its performance in different scenarios and with different objective functions. We analyzed the results of our regression models emphasizing on the most important features that contribute to explain quality in terms of BLEU, Meteor and TER.
Some of the conclusions from these models are straightforward and match the empirically developed intuition. However, the insight gained from this type of analysis can be valuable for designing new systems for new translation tasks. To this end, we gave hints on how this information can be used and provided a practical example on how the information from the regression models can be transformed into features to improve translation.
As a follow up, we plan to carry on with ablation studies that showcase the full potential of this analytic framework. As a future research direction, we anticipate to build a handbook in which important features in our models can be transferred to actions to improve our MT systems.

References

Ayan, N. F. and Dorr, B. J. (2006). Going Beyond AER: An Extensive Analysis of Word Alignments and Their Impact on MT. In *Proc. of the Joint Conference of the International Committee on Computational Linguistics and the Association for Computational Linguistics (COLING-ACL)*, pages 9–16.

Birch, A., Osborne, M., and Koehn, P. (2008). Predicting Success in Machine Translation. In *Proc. of the 2008 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 745–754.

Dagan, I., Glickman, O., and Magnini, B. (2006). The pascal recognising textual entailment challenge. *Machine Learning Challenges. Evaluating Predictive Uncertainty, Visual Object Classification, and Recognising Tectual Entailment*, pages 177–190.

Denkowski, M. and Lavie, A. (2011). Meteor 1.3: Automatic Metric for Reliable Optimization and Evaluation of Machine Translation Systems. In *Proc. of the Workshop on Statistical Machine Translation at the Conference on Empirical Methods in Natural Language Processing*, pages 85–91.

Devlin, J. and Matsoukas, S. (2012). Trait-based hypothesis selection for machine translation. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 528–532, Montréal, Canada. Association for Computational Linguistics.

Fraser, A. and Marcu, D. (2007). Measuring Word Alignment Quality for Statistical Machine Translation. *Computational Linguistics*, 33:293–303.

Guzman, F., Gao, Q., and Vogel, S. (2009). Reassessment of the Role of Phrase Extraction in PBSMT. In *Proc. of the Machine Translation Summit XII*, pages 49–56.

Koehn, P. (2004). Statistical Significance Tests for Machine Translation Evaluation. In *Proc. of Conference on Empirical Methods for Natural Language Processing (EMNLP)*, volume 4, pages 388–395.

Koehn, P., Hoang, H., Birch, A., Callison-burch, C., Federico, M., Bertoldi, N., Cowan, B., Shen, W., Moran, C., Zens, R., Dyer, C., Bojar, O., Constantin, A., and Herbst, E. (2007). Moses: Open Source Toolkit for Statistical Machine Translation. In *Proc. of Association of Computer Linguistics*, pages 177–180.

Koehn, P., Och, F. J., and Marcu, D. (2003). Statistical Phrase-based Translation. In *Proc. of Human Language Technology Conference of the North American Chapter of the Association of Computer Linguistics (HLT-NAACL)*, pages 127–133.

Lambert, P., Gispert, A., Banchs, R., and Mariño, J. B. (2006). Guidelines for Word Alignment Evaluation and Manual Alignment. *Language Resources and Evaluation*, 39(4):267–285.

Lambert, P., Ma, Y., Ozdowska, S., and Way, A. (2009). Tracking Relevant Alignment Characteristics for Machine Translation. In *Proc. of the Machine Translation Summit XII*, pages 268–275.

Lambert, P., Petitrenaud, S., Ma, Y., and Way, A. (2010). Statistical Analysis of Alignment Characteristics for Phrase-based Machine Translation. In *Proc. of the 14th Annual conference of the European Association for Machine Translation (EAMT)*.

Lopez, A. and Resnik, P. (2006). Word-based Alignment, Phrase-based Translation: What's the Link? In *Proc. of the 7th Biennial Conference of the Association for Machine Translation in the Americas (AMTA)*, pages 90–99.

Niehues, J., Herrmann, T., Mediani, M., and Waibel, A. (2010). The karlsruhe institute for technology translation system for the acl-wmt 2010. In *Proceedings of the Joint Fifth Workshop on Statistical Machine Translation and MetricsMATR*, pages 144–148, Uppsala, Sweden. Association for Computational Linguistics.

Niehues, J. and Vogel, S. (2008). Discriminative Word Alignment via Alignment Matrix Modeling. *Computational Linguistics*, pages 18–25.

Pado, S., Galley, M., Jurafsky, D., and Manning, C. D. (2009). Robust machine translation evaluation with entailment features. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 297–305, Suntec, Singapore. Association for Computational Linguistics.

Papineni, K., Roukos, S., Ward, T., and Zhu, W.-j. (2002). Bleu: a Method for Automatic Evaluation of Machine Translation. In *Proc. of the 40th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 311–318.

Snover, M., Dorr, B., Schwartz, R., Micciulla, L., and Makhoul, J. (2006). A Study of Translation Edit Rate with Targeted Human Annotation. In *Proc. of Association for Machine Translation in the Americas (AMTA)*, pages 223 – 231.

Specia, L. (2011). Exploiting objective annotations for measuring translation post-editing effort. In *Proceedings of the 15th Conference of the European Association for Machine Translation*, pages 73–80.

Specia, L., Cancedda, N., Dymetman, M., Turchi, M., and Cristianini, N. (2009). Estimating the sentence-level quality of machine translation systems. In *Proceedings of the 13th Annual Conference of the European Association for Machine Translation (EAMT-2009)*, pages 28–35.

Geolocation Prediction in Social Media Data by Finding Location Indicative Words

HAN Bo[1,2] *Paul COOK*[1] *Timothy BALDWIN*[1,2]
(1) University of Melbourne
(2) NICTA
hanb@student.unimelb.edu.au, paulcook@unimelb.edu.au, tb@ldwin.net

Abstract

Geolocation prediction is vital to geospatial applications like localised search and local event detection. Predominately, social media geolocation models are based on full text data, including common words with no geospatial dimension (e.g. *today*) and noisy strings (*tmrw*), potentially hampering prediction and leading to slower/more memory-intensive models. In this paper, we focus on finding location indicative words (LIWs) via feature selection, and establishing whether the reduced feature set boosts geolocation accuracy. Our results show that an information gain ratio-based approach surpasses other methods at LIW selection, outperforming state-of-the-art geolocation prediction methods by 10.6% in accuracy and reducing the mean and median of prediction error distance by 45km and 209km, respectively, on a public dataset. We further formulate notions of prediction confidence, and demonstrate that performance is even higher in cases where our model is more confident, striking a trade-off between accuracy and coverage. Finally, the identified LIWs reveal regional language differences, which could be potentially useful for lexicographers.

Keywords: Social Media, Geolocation, Feature Selection.

Proceedings of COLING 2012: Technical Papers, pages 1045–1062,
COLING 2012, Mumbai, December 2012.

1 Introduction

With the ever-growing popularity of social media, massive volumes of user-generated data are produced everyday, e.g. in the form of Twitter messages (tweets) and Facebook updates.[1] This data provides many new opportunities and challenges for natural language processing. One such challenge is geolocation prediction: predicting the geolocation of a message or user based on their social media posts. In this paper, we focus on user-level geolocation based on the aggregated body of tweets from a user, and estimate the user's location at the city level.

As is well established in previous work (Cheng et al., 2010; Wing and Baldridge, 2011; Kinsella et al., 2011), it is reasonable to assume that user posts in social media reflect their geospatial locum, because lexical priors differ from region to region. For example, a user in London is much more likely to talk about *Piccadilly* and *British* than a user in New York or Beijing. That is not to say that those terms are uniquely associated with London, of course: *British* could be used by a user outside of the UK to discuss something relating to the UK. However, the use of a range of such terms with high relative frequency is strongly indicative of the fact that a user is located in London.

Our objective in this work is to automatically identify "location indicative words" (LIWs), that is words that implicitly or explicitly encode an association with a particular location. To this end, we refine the geolocation task in Section 3, and explore the impact of LIWs on user geolocation.

Our contributions are as follows: (1) we apply feature selection methods for automatically learning LIWs, and show that both accuracy and efficiency in geolocation are vastly improved using the resultant feature set, achieving state-of-the-art performance over an existing dataset; (2) we develop a city-based world division and a new global geolocation dataset, and demonstrate the effectiveness of the proposed method over this dataset;[2] (3) we conduct a pilot study on the correlation between prediction confidence, as measured by a series of heuristic variables, and classifier accuracy (see Section 5.4); and (4) we find that LIWs selected by our methods are both intuitive and have potential utility in lexicographic research on regional language differences.

The remainder of the paper is organised as follows: Section 2 introduces related work and describes the key questions investigated in this paper. Section 3 outlines the task setting, datasets and evaluation metrics used in this research. Section 4 describes different feature selection methods for extracting LIWs. Section 5 compares the results of the different feature selection methods and discusses their impact on geolocation prediction, and proposes several methods for associating beliefs with predictions. Finally, we conclude the paper and outline possible future work.

2 Related Work and Key Questions

While acknowledging potential privacy concerns (Mao et al., 2011; Pontes et al., 2012), accurate user geolocation is a key driver for location-specific services such as localised search, and has been the target of research across different disciplines. The most reliable and straightforward approach to geolocation prediction is IP-based methods (Buyukokkten et al., 1999), but in many contexts, it is not possible to access the IP of the device used to post content, or the IP is relatively uninformative (as is the case with, e.g. mobile devices). As a result, research has focused on the harder task of geolocation prediction via the textual content of a document (or document set). In the information retrieval community, e.g. web pages (Ding et al., 2000; Amitay et al., 2004; Zong et al., 2005; Silva et al., 2006), search query logs (Wang et al., 2005; Backstrom et al., 2008), Wikipedia edit logs

[1] www.twitter.com; www.facebook.com

[2] The dataset is available from http://www.csse.unimelb.edu.au/~tim/etc/coling2012-geo.tgz.

(Lieberman and Lin, 2009) and Flickr image tags (Crandall et al., 2009; Serdyukov et al., 2009; Hauff and Houben, 2012) have been used as the basis for geolocation prediction. These methods are primarily designed for longer or more homogeneous document sets. In contrast, social media data consists of terse noisy texts, presenting a challenge for these approaches. For instance, any reliance on named entity recognition is thwarted by the unedited nature of social media data, where spelling and capitalisation are much more ad hoc than in edited document collections.

The spatial data mining community has tended to approach the task via identifying geographical references in documents (also known as *geoparsing*: Leidner and Lieberman (2011)). Methods range from naive gazetteer matching and rule-based approaches (Bilhaut et al., 2003), to machine learning-based methods (mainly based on named entity recognition: Qin et al. (2010); Gelernter and Mushegian (2011)). The principal drawback of these methods is that they rely on explicit mentions of addresses or formal placenames in the text, rather than words which are more informally associated with a place. In social media data, we can't rely on a given user mentioning an address or formal placename, severely limiting the coverage of such methods.[3]

There has been a limited amount of work on geolocation prediction based on social network analysis (Backstrom et al., 2010), but social networks are dynamic and the data is often hard to obtain. In terms of text-based geolocation prediction, Cheng et al. (2010) estimate the city-level user geolocation for the continental US with a simple probabilistic model, which they complement with strictly local words and smoothing. Compared with their approach, our LIW selection requires no explicit training data and is more flexible. Wing and Baldridge (2011) use KL-divergence (Kullback and Leibler, 1951) to measure the similarity between different geo-grids specified by geospatial coordinates. Recently, Roller et al. (2012) extend this idea using a KD-tree-based adaptive grid and grid centroids, achieving state-of-the-art geolocation prediction results. Li et al. (2011) investigated the prediction of Places of Interest (POIs) based on linear rank combination of content and temporal factors. Kinsella et al. (2011) compare a variety of geolocation prediction classification models at different location granularities. Adams and Janowicz (2012) utilise external geo-reference data to infer locations. Mahmud et al. (2012) combine timezone information and content-based classifiers in a hierarchical model for geolocation. They only consider nouns, hashtags and place names as features. Recently, Li et al. (2012) integrate both friendship and content information in a probabilistic model. In addition, topic modelling has been applied to the study of geospatially-related tasks including user dialect (Eisenstein et al., 2010), topic discovery (Yin et al., 2011), object matching (Dalvi et al., 2012), factorisation of different geospatial features (Hong et al., 2012), and spatial-temporal analysis (Bauer et al., 2012).

Most work uses the full token set from the training document collection, or a relatively rudimentary approach to feature selection. In this paper, we propose a targeted approach to identify LIWs using various feature selection methods (Yang and Pedersen, 1997), focusing on two key questions:

1. What empirical properties do we observe in LIWs, and what feature selection methods best capture those properties?
2. Can we boost the accuracy of geolocation prediction through targeted identification of LIWs?

3 Geolocation Task Scope and Formulation

We approach geolocation as a text classification task. Tweets from each city are employed to represent a class. All tweets from a given user are aggregated and assigned to the city where that user is based. There are four key components to a geolocation prediction system, which we discuss

[3] An exception is automatically-generated posts from services such as FourSquare which explicitly mention an address.

in turn below: (1) the representation of different geolocations, (2) the model, (3) the data, and (4) the feature set. We then discuss evaluation metrics for geolocation prediction.

3.1 Representation: Earth Grid vs. City

Geolocations can be captured as points, or clustered based on grids (Wing and Baldridge, 2011; Roller et al., 2012) or population centres (Cheng et al., 2010; Kinsella et al., 2011). A point-based representation presents computational challenges, and is too fine-grained for our task. We opt for a city-based representation rather than a grid-based representation because there is considerable variability in the shape and size of geographical regions: a coarse-grained grid cell is perhaps appropriate in central Siberia, but for densely-populated and linguistically/culturally diverse regions such as Luxembourg, doesn't lead to a natural representation of the administrative, population-based or language boundaries in the region. A city-based representation is able to capture these boundaries more intuitively. The only downside to a city-based representation is that it is inappropriate for classifying users in rural areas. As we will see, however, the bulk of users on services such as Twitter are, unsurprisingly, based in cities.

We use the publicly-available Geoname dataset as the basis for our city categorisation.[4] Geoname contains city-level metadata, including the full city name, population, latitude and longitude. The city name is associated with hierarchical regional information, like the state and country it is based in, so that London in Britain, e.g. is distinguished from London in Canada. We hence use a city-region-country format to represent each city (e.g. Toronto, Canada is represented as toronto-08-ca, where 08 signifies the province of Ontario and ca signifies Canada). Because region coding schemes vary across different countries, we only employ the first and second level region fields in Geoname as the region. Furthermore, if the second level field is too specific (i.e. longer than 4 letters), we then only incorporate the first level region field (e.g. instead of using melbourne-07-24600-au, we use melbourne-07-au). Moreover, because cities are sometimes complex in structure (e.g. Boston in Massachusetts colloquially refers to the metropolitan area rather than the city, which is made up of cities including the cities of Boston, Revere and Chelsea), we collapse together cities which are adjacent to one another within a single administrative region, as follows:

1. Identify all cities which share the same region code (i.e. are located in the same state, province, county, etc.) in the Geoname dataset.
2. For each region, find the city c with the highest population.
3. Collapse all cities within 50km of c into c.
4. Select the next-largest city c, and repeat.
5. Remove all cities with a population less than 100K. The remaining cities form our city-based representation of geolocations.

As a result of these procedures, Boston ends up as a single city (incorporating Revere and Chelsea), but neighbouring Manchester is a discrete city (incorporating Bedford) because it is in New Hampshire. This algorithm identifies a total of 3,709 cities throughout the world.

3.2 Generative vs. Discriminative models

Generative models (e.g. naive Bayes) are based on estimation of the class priors (i.e. $P(c_i)$) and the probability of observing a given term vector given a class (i.e. $P(w_1, w_2, \ldots, w_n|c_i)$). In contrast, discriminative models are based on estimation of a given class given a term vector (i.e.

[4] http://www.geonames.org.

Name	Cities	Users	Tweets	Types	Tokens	Region
NA	378	500K	38M	4.92M	436M	North America
WORLD	3135	1.39M	12M	0.85M	103M	the world

Table 1: Details of the two datasets used in this research.

$P(c|w_1, w_2, \ldots, w_n)$). The objective of both models is to find a city $c_{max} \in C$ such that the relevant probability is maximised. We experiment with both types of models in our experiments in Section 5.3. In this paper, we choose a generative multinomial naive Bayes (NB) model as our benchmark, for two reasons: (1) it incorporates a class prior, allowing it to classify an instance in the absence of any features shared with the training data; and (2) generative models outperform discriminative models when training data is relatively scarce (Ng and Jordan, 2002).[5]

3.3 Data

In this paper, we employ two geo-tagged datasets: (1) the regional North America geolocation dataset of Roller et al. (2012) (NA hereafter), for benchmarking purposes; and (2) a novel dataset that covers the entire globe (WORLD), collected for the purposes of this research via the Twitter public Streaming API[6] from September 21 2011 to February 29 2012.

In building WORLD, we first filter non-English tweets using `langid.py`, an open-source language identification tool (Lui and Baldwin, 2012), and then apply a Twitter tokeniser (adapted from O'Connor et al. (2010)). We restrict WORLD to English tweets in order to create a dataset similar to NA (in which English is the predominant language), but covering the entire world. A further reason for only using English tweets is to control for the influence of language priors on geolocation performance. For example, we expect that a language such as Japanese would tend to be more skewed towards particular cities than English, making the task of text-based geolocation easier.[7] We further eliminate Foursquare check-ins, as they mention the location of the user and are geo-tagged, and duplicate tweets. We also remove tweets from users with less than 10 geo-tagged tweets to reduce feature sparsity. Finally, we eliminate all tweets which aren't close to a city by dividing the earth into $0.5' \times 0.5'$ grids, and discarding any tweet for which no city is found in any of the 8 neighbouring grid cells. We then assign each user to the single city in which the majority of their tweets occur. Note that the processing described in this paragraph applies only to WORLD; NA was left as-is to ensure comparability with previous work.

Analysis of a sample of 26 million tweets (not filtered as above) reveals that 92.1% of tweets are "close" to (in a neighbouring $0.5' \times 0.5'$ grid cell) of one of our 3,709 cities, and that the top 40% of cities contain 90% of the tweets, as shown in Figure 1.

A statistical profile of NA and WORLD is presented in Table 1.[8] We also analyse the spread of WORLD in Figure 2, in terms of: (1) the number of users with a given number of tweets; and (2) the number of users with differing levels of geographical spread in their tweets, measured as the average distance between each of a user's tweets and the centre of the city to which that user is allocated.[9] This analysis shows that most users have a relatively small number of geo-tagged tweets, and most

[5] There is certainly an abundance of Twitter data to train models over, but the number of Twitter users with sufficient amounts of geo-tagged tweets to be able to perform geolocation prediction is small, relative to the number of parameters in the model (the product of the number of features and classes).

[6] `https://dev.twitter.com/docs/streaming-apis`

[7] All of the methods we consider could nevertheless be easily applied to a mixed-language setting in the future.

[8] WORLD has only 3,135 (as opposed to 3,709) cities because some cities have no tweets.

[9] The geographical spread is calculated over a random sub-sample of 10 tweets for a given user, for efficiency reasons.

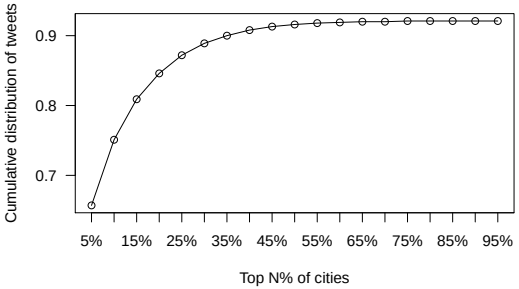


Figure 1: Cumulative coverage of tweets for increasing numbers of cities based on 26 million geo-tagged tweets.

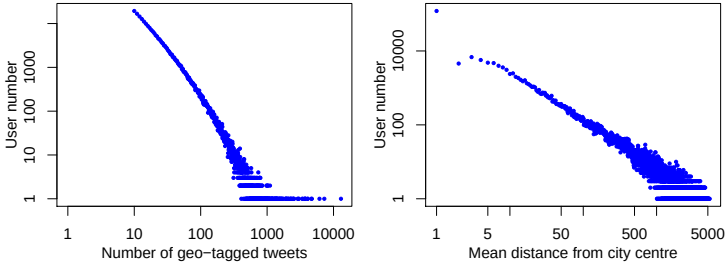


Figure 2: The number of users with different numbers of tweets, and different mean distances from the city center, for WORLD.

users stay near a single city.

3.4 Features: All unigrams vs. Location Indicative Words

Feature selection is a key contribution of this paper, based on the notion of "location indicativeness", as described in Section 4. Our hypothesis is that using only location indicative words (LIWs) as features will be more efficient and effective than using all terms. Rather than engineering new features or attempting to capture named entities or higher-order n-grams, we focus on feature selection over simple term unigrams. This is partly a pragmatic consideration (preliminary results with both named entities and higher order n-grams were disappointing). Partly, however, it is for comparability with past work, in determining whether a strategically selected subset of terms can lead to significant gains in geolocation accuracy.

3.5 Evaluation Metrics

Having reformulated the geolocation prediction task into a discrete class space through the use of our city class set, it is possible to use simple classification accuracy to evaluate our models. However, given that all of our class labels have a location (in the form of latitude–longitude coordinates),

we can also sensitise our evaluation to the distance-based error in predictions. For instance, if the correct location for a user is Seattle, USA, a prediction of Vancouver, Canada is arguably better than a prediction of Los Angeles, USA, on the basis of geospatial proximity. In line with past work (Cheng et al., 2010; Wing and Baldridge, 2011; Roller et al., 2012), we use a number of evaluation metrics which capture spatial proximity, in addition to classification accuracy:

1. **Acc** : the classification accuracy of the highest-probability prediction of the model;
2. **Acc@161** : the classification accuracy of the highest-probability prediction of the model, within a circle of radius of 100 miles (161 kilometres) from the true city centre of the user;
3. **Mean and Median Error**: mean and median prediction error, measured in kilometres between the predicted city centres and the true geolocations.

4 Finding Location Indicative Words

In this section, we experiment with different methods for ranking location indicative words. As a first step, to determine the statistical "signature" of LIWs, we manually pre-identified seed sets of: (1) local words (denoted as 1-local) that are used primarily in a single city, namely *yinz* (used in Pittsburgh to designate locals), *dippy* (used in Pittsburgh to refer to a style of fried egg, or something that can be dipped in coffee, etc.) and *hoagie* (used primarily in Philadelphia, to refer to a kind of sandwich);[10] (2) semi-local words (*n*-local) that refer to some feature of a relatively limited subset of cities, namely *ferry* (found, e.g. in Seattle, New York and Sydney), *Chinatown* (common in many of the largest cities in the USA, Canada and Australia, but much less common in European and Asian cities), and *tram* (found, e.g, in Vienna, Melbourne and Prague); and (3) common words (common) which aren't expected to have substantial regional frequency variation, namely *twitter*, *iphone* and *today*. We use this small set of 9 words to empirically motivate our feature selection approach.

4.1 Decoupling City Frequency and Word Frequency

High-utility LIWs should have both of the following properties:

1. High Term Frequency (TF): there should be a reasonable expectation of observing it for a given user;
2. High Inverse City Frequency (ICF): the term should occur in tweets associated with a relatively small number of cities.

We calculate the ICF of a term i simply as $icf_i = \frac{N}{cf_i}$, where N is the number of cities and cf_i is the number of cities with users who use that term in the training data. Combining the two together, we are seeking words with high "TF-ICF", analogous to seeking terms with high TF-IDF values in information retrieval. As with TF-IDF, however, the exact formulation for calculating the individual values and combining them into a single term weight is not necessarily obvious. A simple TF×ICF product is dominated by the TF component: for example, *twitter* scores as highly as *Jakarta*, because *twitter* has a very high TF. We resolve this issue by decoupling the two factors and applying a radix sort ranking: we first rank features by ICF then by TF, in decreasing order. This procedure has the desired effect of promoting local words and demoting common words. We present evidence for this hypothesis over the pre-identified 1-local, *n*-local and common words in Figure 3.

We can observe that 1-local words have high ICF and relatively low TF, *n*-local words have mid-range ICF and TF values, and common words have low ICF and high TF values, anecdotally justifying our feature ranking method. In order to filter out low-utility words and noise, we only keep words with

[10]These terms are identified with the aid of datasets of regional terms such as DARE `http://dare.wisc.edu/`.

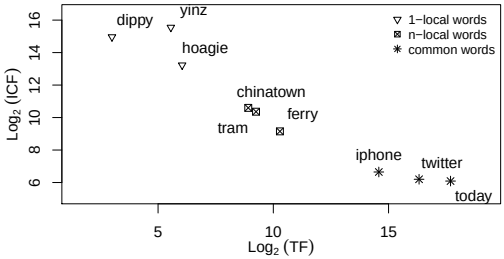


Figure 3: Inverse city frequency vs. term frequency on WORLD

1-local terms	IGR	*n*-local terms	IGR	common terms	IGR
yinz	0.287	ferry	0.125	iphone	0.012
dippy	0.169	tram	0.188	twitter	0.005
hoagie	0.183	chinatown	0.107	today	0.027

Table 2: Information gain ratio numbers for our sample terms based on WORLD

minimum character length of 3 and $TF \geq 10$ hereafter. As this approach is largely based on the inverse city frequency, we denote it as *ICF* below.

4.2 Information Gain Ratio

In addition to ICF, we also employ Information Gain (IG), an information-theoretic measure of the decrease in entropy a word brings about, where higher values indicate greater predictability on the basis of that feature. Given a set of words $\mathbf{w}$, the IG of a word $w_i \in \mathbf{w}$ across all cities (C) is calculated as follows:

$$IG(w_i) = H(C) - H(C|w_i) \quad (1)$$

$$\propto P(w_i)\sum_{j=1}^{m} P(c_j|w_i) log P(c_j|w_i) + P(\bar{w}_i)\sum_{j=1}^{m} P(c_j|\bar{w}_i) log P(c_j|\bar{w}_i) \quad (2)$$

where $H(C|w_i)$ is the conditional entropy given w_i, which is proportional to $IG(w_i)$.

Words carry varying amounts of "intrinsic entropy", which is defined as $IV(w_i) = -P(w_i) log P(w_i) - P(\bar{w}_i) log P(\bar{w}_i)$. Local/regional words occurring in a small number of cities often have a low intrinsic entropy, where non-local common words have a high intrinsic entropy. For words with comparable IGs, the words with smaller entropies are preferred. Therefore, following Quinlan (1993) we further normalise $IG(w_i)$ using the intrinsic entropy of word $IV(w_i)$, culminating in information gain ratio (IGR): $IGR(w_i) = IG(w_i)/IV(w_i)$. Returning to our earlier sample words, we present IGR scores for WORLD in Table 2.

4.3 Maximum Entropy-based Feature Weights

The previous two feature selection methods optimise across all classes simultaneously. Given that some LIWs may be strongly associated with certain locations, but are less tied to other locations, we also conduct per-class feature selection based on maximum entropy (ME) modelling.[11]

[11] https://github.com/lzhang10/maxent

1-local	Weight	City	*n*-local	Weight	City	common	Weight	City
yinz	6.8e-3	Pittsburgh, US	ferry	1.4e-2	San Francisco, US	iphone	3.2e-2	London, UK
dippy	4.6e-4	Gosport, UK	tram	2.7e-2	Melbourne, AU	twitter	3.7e-2	Bowie, US
hoagie	4.2e-3	Philadelphia, US	chinatown	9.5e-3	Singapore	today	9.3e-2	London, UK

Table 3: ME-based feature weights, and associated cities, for our sample terms based on WORLD.

Given a collection of cities C, the ME model calculates the probability of a user (e.g. represented by word sequence: $w_1, w_2, \ldots, w_n$) assigned to a city c by linearly combining eligible ME feature weights:

$$p(c|w_1, w_2, \ldots, w_n) = \frac{1}{Z} \sum_{k=1}^{m} \lambda_k f_k \tag{3}$$

Here, Z is the normalisation factor, m is the total number of features, and f_k and λ_k are the features and feature weights, respectively. As with other discriminative models, it is possible to incorporate arbitrary features into ME, however, a feature (function) in our task is canonically defined as a word w_i and a city C_j. When w_i occurs in C_j, a feature $f_k(w_i, C_j)$ is denoted as $[class = C_j \wedge w_i \in C_j]$. Each f_k maps to a feature weight denoted as $\lambda_k \in \mathscr{R}$.

Our goal is to estimate feature weights using a ME model. The key idea is that ME features connect a word and a class, with larger λ_k weights indicating stronger word–class associations. Therefore, we should be able to use the learned weights as a means of both ranking features and grouping features by cities. With all features, we generate a weight per term–class pair and rank all weights in decreasing order. Only the first rank position is kept for a given term, and this then forms the final aggregated class-independent feature rank. We don't incorporate a regularizer in our ME model (which can help to avoid over-fitting) because we already removed infrequent words (see Section 4.1), which serves as a basic count-based heuristic regularizer. The comparable results on the development and test sets presented in Sections 5.2 and 5.3 indicate that the feature selection is indeed not overfitting.

We show the ME-based weights, and associated cities, for our sample terms for the WORLD dataset in Table 3. Note that in the case of the 1-local terms *yinz* and *hoagie*, the expected city associations have been learned. Such associations could further be of use to lexicographers in identifying regional usages from social media.

5 Experiments and Analysis

5.1 Comparison of Feature Selection Methods

First we compare the effectiveness of the different feature selection methods in experiments on both NA and WORLD. In total, 214K and 96K features are extracted from the training sections of NA and WORLD, respectively. For each feature selection method, we select the $top\ N\%$ of these features, and then use the selected features in multinomial naive Bayes classification; we compare performance using Acc@161. In this section we consider results only on the development portion (10K held-out users) of each dataset; we use these results to optimise feature selection parameters which we then use in subsequent experiments in the following subsections. Results for NA are shown in Figure 4; results for WORLD are similar and thus omitted from the paper.

For ICF and IGR, Acc@161 rises as the percentage of features selected is increased, and drops dramatically at a very high percentage of selected features. The features which are selected last

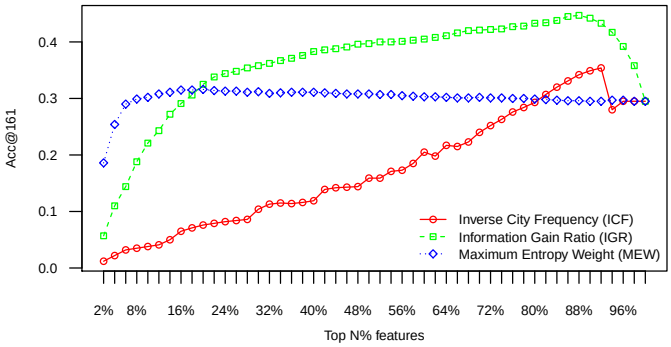


Figure 4: Acc@161 for varying percentages of features selected using the feature selection methods on the NA dataset. The Optima for ICF, IGR, and MEW are 92%, 88%, and 20%, respectively.

appear to be high-frequency function words (e.g. *the*) and common terms (e.g. *facebook*), which give little indication as to geolocation and lead to prediction errors. By identifying and removing these features, performance can be improved. On the other hand, when insufficient features are used, naive Bayes appears to be under-fitting the training data, and tends to assign classes according to the prior. For instance, when using just the top 2% of features, the most likely class in each case is monterrey-19-mx, because Spanish words are highly location indicative of the small number of Mexican Cities in the NA dataset. For maximum entropy weighting (MEW) we see a very different pattern: we attain the highest Acc@161 using the top 20% features, with performance gradually decreasing as more features are added. The overall poor performance of MEW seems to be due at least in part to the first-occurrence heuristic (see Section 4.3), which causes some non–location indicative words to be ranked higher than local words. Overall, IGR achieves the highest accuracy.

We observe that, as expected, the highly-ranked LIWs for IGR include local dialectal terms (e.g. *yinz*) and place names (e.g. *portland*). We further note the importance of word frequency on location indicativeness. For example, in WORLD the *n*-local term *tram* is roughly an order of magnitude more frequent than the 1-local term *hoagie*, but has higher IGR (see Table 2). Although 1-local words might be useful in geolocation, the impact of infrequent terms on overall accuracy is limited. In particular, frequent words with some geographical association might be more informative for geolocation than words with highly-local distribution but lower frequency. Furthermore, from this analysis it seems that a binary distinction cannot be made between local and non-local words (as in Cheng et al., 2010), but rather that many words carry some geographical indications. As for LIW selection by class (city) with MEW, city names are unsurprisingly strong indicators for locations. For example, *philadelphia* and *philly* are amongst the top-three words associated with philadelphia-pa101-us in NA. Furthermore, upper level administrative regions are also useful for geolocation with, e.g. *georgia* being a strong indicator of atlanta-ga121-us in NA.

5.2 Improved Accuracy with Location Indicative Words

In this subsection we compare the accuracy of classifiers trained using just the optimised LIWs obtained in the previous subsection to that of the full model. The performance is measured on the

Dataset	Features	Mean	Median	Acc@161	Acc
NA	Full	1010	571	0.308	0.171
	ICF	1026	533	0.359	0.209
	IGR	**814**	**260**	**0.450**	**0.260**
	MEW	890	520	0.326	0.183
WORLD	Full	2215	917	0.203	0.081
	ICF	2299	878	0.239	0.107
	IGR	3002	926	**0.259**	**0.124**
	MEW	**1953**	**646**	0.241	0.103

Table 4: Geolocation performance of the full feature set compared to that of each feature selection methodology on both NA and WORLD. The best results for each dataset and accuracy metric are shown in boldface.

test data (10K held-out users for both NA and WORLD).

Results on NA show that using LIWs offers an improvement over the full feature set for all evaluation metrics and all feature selection approaches (except for ICF with mean distance). On WORLD the findings are similar in terms of Acc and Acc@161, however, mean distance in particular is substantially higher for IGR. We hypothesize that this is because incorrect world-level predictions can potentially be off by thousands of kilometres, driving up the mean distance more than the other metrics. Overall, these numbers clearly demonstrate that identification of LIWs can improve text-based geolocation. IGR performs best in terms of Acc@161 on both datasets, achieving a 14.2% and 5.6% absolute improvement over the full feature set on NA and WORLD, respectively. Finally, it is worth noting that the raw accuracy on NA is higher than that on WORLD. This is unsurprising because the smaller average number of tweets and larger number of classes for WORLD make it a more challenging dataset.

5.3 Comparison with Benchmarks

We further compare the best-performing method from Section 5.2 to benchmarks and baselines. Here we only consider NA, for which results have been previously reported. We experiment with two partitionings of the Earth's surface: (1) the new city-based division used in the previous experiments, and (2) the KD-tree based partitioning of Roller et al. (2012) which creates grid cells containing roughly even amounts of data, but differing geographical sizes, such that higher-population areas are represented with finer-grained grids. We consider the following methods:

Baseline Because the geographical distribution of tweets is skewed towards higher-population areas (as indicated in Figure 1), we consider a most-frequent class baseline. We assign all users the coordinates of the most-common city centre or KD-tree grid centroid in the training data.

Placemaker Following Kinsella et al. (2011), we obtain results from Yahoo! Placemaker,[12] a publicly-available geolocation service. The first 50K bytes (the maximum query length allowed by Placemaker) from the tweets for each user are passed to Placemaker as queries. The returned city centre predictions are mapped to our collapsed city representations. For queries without results, or with a predicted location outside NA, we back off to the baseline.

KL divergence The previous best published results over NA were achieved using KL divergence and the KD-tree grid. Specifically, KL divergence is measured between the distribution of terms in a user's aggregated tweets and that in each grid cell, with the predicted location being the centroid of the most-similar grid cell. We use the same settings as Roller et al..

Multinomial naive Bayes This is the model used in Section 5.2.

[12] `http://developer.yahoo.com/geo/placemaker/`, accessed in August 2012.

Partition	Method	Mean	Median	Acc@161	Acc
KD-tree	Baseline	1528	1189	0.118	0.003
	KL	859	469	0.344	0.117
	KL+IGR	766	**273**	**0.437**	**0.161**
	NB	835	404	0.367	0.122
	NB+IGR	**763**	280	0.432	0.153
City	Baseline	2707	3089	0.062	0.003
	Placemaker	2188	1857	0.150	0.049
	NB	1010	571	0.308	0.171
	NB+IGR	**814**	**260**	**0.450**	**0.260**
	ME	1336	878	0.232	0.129
	ME+IGR	891	369	0.406	0.229

Table 5: Geolocation performance for baselines, KL divergence (KL), multinomial naive Bayes (NB), and Maximum Entropy (ME). Results using the optimised feature set (+IGR) are also shown. The best-performing method for each evaluation metric and partitioning is shown in boldface.

Maximum entropy The features from Section 5.2 with a maximum entropy learner.[13]

The results are shown in Table 5. We begin by considering the baseline results. The most-frequent class for the KD-tree grid is New York (the state), while for the city-based partition, it is los angeles-ca037-us. Both baselines perform below the other models, suggesting that geolocation cannot be trivially solved. Looking at the results for Placemaker (which we only consider for the city-based partition) we see very high mean and median scores. This appears to be due to the scope and domain of this service, which predicts locations at the world level, whereas the other methods are restricted to North America.

The KL-divergence method of Roller et al. (KL) and multinomial naive Bayes method (NB) both clearly outperform the baseline. Moreover, approaches incorporating the best features selected in Section 5.2 — KL+IGR and NB+IGR — both outperform KL and NB, demonstrating that for a variety of approaches, identification of LIWs can improve text-based geolocation.[14] From the results on the KD-tree grid it is not decisively clear which of KL or NB is better for our task: in terms of Acc@161, e.g., NB outperforms KL, but KL+IGR outperforms NB+IGR.

Turning to the results for the city-based grid, our best-performing method from Section 5.2 (City, NB+IGR) performs best overall in terms of Acc, Acc@161, and median distance, confirming the effectiveness of LIWs. Compared to the best published results at the time of writing (KD-tree, KL), our method offers a 10.6% absolute improvement in terms of Acc@161, and reduces the mean and median prediction error by 45Km and 209Km, respectively.[15] Finally, although maximum entropy (ME) performs poorly compared to NB, ME+IGR is still a substantial improvement over ME. We plan to further explore the reasons for ME's poor performance in future work.

In addition to improving the accuracy of geolocation, LIW-based feature selection leads to more compact models, which are more efficient in terms of computational processing and memory.

[13] Although there are many other classifiers we could consider, when sufficient training data for each class is available, the performance of different methods is comparable (Yang and Liu, 1999). Moreover, many state-of-the-art classifiers (Wu et al., 2007) are not primarily designed for massively multi-class problems (e.g. support vector machines (Vapnik, 1995)), or are not efficient when applied to such problems (e.g. *k*-nearest neighbour (Steinbach et al., 2006)).

[14] Note that after LIWs are selected, a small proportion of users end up with no features. These users are not geolocatable in the case of KL, a discriminative model. We turn off feature selection for such users, and backoff to the full feature set, so that the number of test cases is consistent in different settings.

[15] Acc is not comparable for the different partitionings, i.e. KD-tree vs. city, because of the differing numbers of grid cells. High accuracy could trivially be achieved with a very coarse-grained grid.

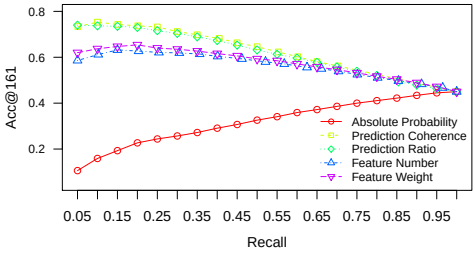


Figure 5: Acc@161 for classification of the top-n% most-confident predictions for each measure of prediction confidence.

Comparing the model based on LIWs selected using IGR with the full model, we find that the prediction time is faster by a factor of roughly five.

5.4 The Confidence of Geolocation Prediction

In the task setup to date, we have forced our models to geolocate all users. In practice, however, many users don't mention any explicitly geolocating terms in their posts, making the task nigh on impossible even for a human oracle. An alternative approach would be to predict a user geolocation only when the model is confident of its prediction. Here, we take our best-performing method (city-based grid, multinomial naive Bayes classifier with LIWs selected using IGR) and consider different methods for selecting users where the model is sufficiently confident of its prediction:

Absolute probability (AP) Only consider predictions with probability above a specified threshold.

Probability ratio (PR) If the model is confident in its prediction, the first prediction will tend to be much more probable than other predictions. We formulate this intuition as PR, the ratio of the probability of the first and second predictions.

Prediction coherence (PC) We hypothesize that for reliable predictions, the top-ranked locations will tend to be geographically close. In this preliminary exploration of coherence, we formulate PC as the sum of the reciprocal ranks of the predictions corresponding to the second-level administrative region in our class representation (i.e. state or province) of the top-ranking prediction, calculated over the top-10 predictions. For example, suppose the top-10 second-level predictions were in the following states: TX, FL, TX, TX, CA, TX, TX, FL, CA, NY. The top-ranking state-level prediction is therefore TX, which also occurs at ranks 3, 4, 6 and 7. In this case, PC would be $\frac{1}{1} + \frac{1}{3} + \frac{1}{4} + \frac{1}{6} + \frac{1}{7}$.

Feature number (FN) We take the number of features found in a user's posts as the prediction accuracy. The intuition here is that a geolocation prediction based on more features is more reliable than a prediction based on less features.

Feature weight (FW) Similar to FN, but in this case we use the sum of IGR of all features, rather than just the number of features.

For this analysis we use the NA dataset. We sort the predictions by confidence (independently for each measure of prediction confidence) and measure Acc@161 amongst the top-n% of predictions for the following values of n: $\{0.0, 0.05, ..., 1.0\}$, akin to a precision–recall curve. Results are shown in Figure 5. The naive AP method is least reliable with, surprisingly, accuracy increasing

as AP decreases. It appears that the raw probabilities are not an accurate reflection of prediction confidence. In comparison, PR — which focuses on relative, as opposed to raw, probabilities — performs much better, with higher PR generally corresponding to higher accuracy. Nevertheless, the best-performing method is PC, which only uses the probabilities to rank the class predictions, and roughly captures the geographical proximity of the top predictions, confirming our hypothesis that accuracy will tend to be higher when the top-ranked predictions are relatively near each other. For PC, Acc@161 is about 75% when only the 10% most-confident predictions are considered, which is well above the 45% Acc@161 for the full dataset. FN and FW show similar trends to PC and PR, but don't perform as well. These experiments suggest that there is indeed a trade-off between coverage and accuracy, which could be exploited to obtain higher-accuracy predictions by applications that do not require all the data to be classified.

Discussion and Conclusion

We have investigated various methods for applying feature selection to identify LIWs (location indicative words) for the task of text-based geolocation. Our results on two different datasets demonstrate that using LIWs leads to an improvement over using a full feature set for a variety of evaluation metrics. Furthermore, our best method using LIWs outperforms the previous state-of-the-art on a standardised dataset, and is much faster. These results demonstrate the potential for improving text-based geolocation through feature selection. The LIWs identified by our method, and their associations with particular locations, may also be useful for lexicographers in describing regional usage and variation. We further considered prediction confidence, and showed that it is possible to strike a trade-off between coverage and accuracy; given the very large amount of Twitter data available, a system which gives more-accurate predictions, but only for a subset of the data, may be useful in some applications. Finally, the proposed LIW selection methods, although developed and evaluated on English datasets, could be easily applied in a multilingual setting.

This paper (as well as previous work on this topic) only considered tweets with gold-standard geo-tags, but in an applied setting we envision these models being applied to non-geotagged tweets to infer their locations. However, it might not be the case that geo-tagged tweets (typically sent from a GPS-enabled device such as a smart phone) have the same properties as those which are not geo-tagged (and are sent from a variety of devices, including desktop computers). In future work, we intend to investigate the relationship between these two sources of data. Although the aim of this paper was to examine the relationship between text and location, there are nevertheless further sources of information available on Twitter, such as user profile and social network information, that could be leveraged in a method for geolocation. In future work we intend to consider the incorporation of such information into our methods. Finally, this paper proposed a new city-based representation of locations. We plan to continue in this direction in future work to explore alternative regional partitionings, as well as hierarchical classification methods.

Acknowledgements

NICTA is funded by the Australian government as represented by Department of Broadband, Communication and Digital Economy, and the Australian Research Council through the ICT Centre of Excellence programme.

References

Adams, B. and Janowicz, K. (2012). On the geo-indicativeness of non-georeferenced text. In *Proceedings of Sixth International AAAI Conference on Weblogs and Social Media*, ICWSM '12,

Dublin, Ireland.

Amitay, E., Har'El, N., Sivan, R., and Soffer, A. (2004). Web-a-where: geotagging web content. In *Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '04, pages 273–280, Sheffield, United Kingdom. ACM.

Backstrom, L., Kleinberg, J., Kumar, R., and Novak, J. (2008). Spatial variation in search engine queries. In *Proceeding of the 17th international conference on World Wide Web*, WWW '08, pages 357–366, Beijing, China. ACM.

Backstrom, L., Sun, E., and Marlow, C. (2010). Find me if you can: improving geographical prediction with social and spatial proximity. In *Proceedings of the 19th international conference on World wide web*, WWW '10, pages 61–70, Raleigh, North Carolina, USA. ACM.

Bauer, S., Noulas, A., Seaghdha, D. O., Clark, S., and Mascolo, C. (2012). Talking places: Modelling and analysing linguistic content in foursquare. In *Proceedings of The ASE/IEEE International Conference on Social Computing*, SocialCom 2012, Amsterdam, Netherlands.

Bilhaut, F., Charnois, T., Enjalbert, P., and Mathet, Y. (2003). Geographic reference analysis for geographic document querying. In *Proceedings of the HLT-NAACL 2003 workshop on Analysis of geographic references - Volume 1*, HLT-NAACL-GEOREF '03, pages 55–62. Association for Computational Linguistics.

Buyukokkten, O., Cho, J., Garcia-Molina, H., Gravano, L., and Shivakumar, N. (1999). Exploiting geographical location information of web pages. In *ACM SIGMOD Workshop on The Web and Databases (WebDB'99)*, pages 91–96.

Cheng, Z., Caverlee, J., and Lee, K. (2010). You are where you tweet: a content-based approach to geo-locating twitter users. In *Proceedings of the 19th ACM international conference on Information and knowledge management*, CIKM '10, pages 759–768, Toronto, ON, Canada. ACM.

Crandall, D. J., Backstrom, L., Huttenlocher, D., and Kleinberg, J. (2009). Mapping the world's photos. In *Proceedings of the 18th international conference on World wide web*, WWW '09, pages 761–770, Madrid, Spain. ACM.

Dalvi, N., Kumar, R., and Pang, B. (2012). Object matching in tweets with spatial models. In *Proceedings of the fifth ACM international conference on Web search and data mining*, WSDM '12, pages 43–52, Seattle, Washington, USA. ACM.

Ding, J., Gravano, L., and Shivakumar, N. (2000). Computing geographical scopes of web resources. In *Proceedings of the 26th International Conference on Very Large Data Bases*, VLDB '00, pages 545–556, Cairo, Egypt.

Eisenstein, J., O'Connor, B., Smith, N. A., and Xing, E. P. (2010). A latent variable model for geographic lexical variation. In *Empirical Methods in Natural Language Processing*, pages 1277–1287, Cambridge, MA, USA.

Gelernter, J. and Mushegian, N. (2011). Geo-parsing messages from microtext. *Transactions in GIS*, 15(6):753–773.

Hauff, C. and Houben, G.-J. (2012). Geo-location estimation of flickr images: social web based enrichment. In *Proceedings of the 34th European conference on Advances in Information Retrieval*, ECIR'12, pages 85–96, Barcelona, Spain. Springer-Verlag.

Hong, L., Ahmed, A., Gurumurthy, S., Smola, A. J., and Tsioutsiouliklis, K. (2012). Discovering geographical topics in the twitter stream. In *Proceedings of the 21st international conference on World Wide Web*, WWW '12, pages 769–778, Lyon, France. ACM.

Kinsella, S., Murdock, V., and O'Hare, N. (2011). "i'm eating a sandwich in glasgow": modeling locations with tweets. In *Proceedings of the 3rd international workshop on Search and mining user-generated contents*, SMUC '11, pages 61–68, Glasgow, Scotland, UK. ACM.

Kullback, S. and Leibler, R. A. (1951). On information and sufficiency. *Annals of Mathematical Statistics*, 22:49–86.

Leidner, J. L. and Lieberman, M. D. (2011). Detecting geographical references in the form of place names and associated spatial natural language. *SIGSPATIAL Special*, 3(2):5–11.

Li, R., Wang, S., Deng, H., Wang, R., and Chang, K. C.-C. (2012). Towards social user profiling: unified and discriminative influence model for inferring home locations. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '12, pages 1023–1031, Beijing, China. ACM.

Li, W., Serdyukov, P., de Vries, A. P., Eickhoff, C., and Larson, M. (2011). The where in the tweet. In *Proceedings of the 20th ACM international conference on Information and knowledge management*, CIKM '11, pages 2473–2476, Glasgow, Scotland, UK. ACM.

Lieberman, M. D. and Lin, J. (2009). You are where you edit: Locating wikipedia contributors through edit histories. In *ICWSM*.

Lui, M. and Baldwin, T. (2012). langid.py: An off-the-shelf language identification tool. In *Proceedings of the ACL 2012 System Demonstrations*, pages 25–30, Jeju Island, Korea.

Mahmud, J., Nichols, J., and Drews, C. (2012). Where is this tweet from? inferring home locations of twitter users. In *Proceedings of Sixth International AAAI Conference on Weblogs and Social Media*, ICWSM '12, Dublin, Ireland.

Mao, H., Shuai, X., and Kapadia, A. (2011). Loose tweets: an analysis of privacy leaks on twitter. In *Proceedings of the 10th annual ACM workshop on Privacy in the electronic society*, WPES '11, pages 1–12, Chicago, Illinois, USA. ACM.

Ng, A. Y. and Jordan, M. (2002). On discriminative vs. generative classifiers: A comparison of logistic regression and naive Bayes. In Thomas G. Dietterich, S. B. and Ghahramani, Z., editors, *Advances in Neural Information Processing Systems 14*. MIT Press.

O'Connor, B., Krieger, M., and Ahn, D. (2010). TweetMotif: Exploratory search and topic summarization for Twitter. In *Proceedings of Fourth International AAAI Conference on Weblogs and Social Media*, pages 384–385, Washington, USA.

Pontes, T., Vasconcelos, M., Almeida, J., Kumaraguru, P., and Almeida, V. (2012). We know where you live: Privacy characterization of foursquare behavior. *4th International Workshop on Location-Based Social Networks (LBSN 2012)*.

Qin, T., Xiao, R., Fang, L., Xie, X., and Zhang, L. (2010). An efficient location extraction algorithm by leveraging web contextual information. In *Proceedings of the 18th SIGSPATIAL International Conference on Advances in Geographic Information Systems*, GIS '10, pages 53–60, San Jose, California. ACM.

Quinlan, J. R. (1993). *C4.5: Programs for Machine Learning*. Morgan Kaufmann, San Mateo, USA.

Roller, S., Speriosu, M., Rallapalli, S., Wing, B., and Baldridge, J. (2012). Supervised text-based geolocation using language models on an adaptive grid. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 1500–1510, Jeju Island, Korea.

Serdyukov, P., Murdock, V., and van Zwol, R. (2009). Placing flickr photos on a map. In *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '09, pages 484–491, Boston, MA, USA. ACM.

Silva, M. J., Martins, B., Chaves, M. S., Afonso, A. P., and Cardoso, N. (2006). Adding geographic scopes to web resources. *Computers, Environment and Urban Systems*, 30:378–399.

Steinbach, P., Kumar, M., and Tan, V. (2006). Introduction to data mining. *International Edition.–NY.: Addison Wesley*.

Vapnik, V. N. (1995). *The nature of statistical learning theory*. Springer-Verlag New York, Inc., New York, NY, USA.

Wang, L., Wang, C., Xie, X., Forman, J., Lu, Y., Ma, W.-Y., and Li, Y. (2005). Detecting dominant locations from search queries. In *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '05, pages 424–431, Salvador, Brazil. ACM.

Wing, B. P. and Baldridge, J. (2011). Simple supervised document geolocation with geodesic grids. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1*, HLT '11, pages 955–964, Portland, Oregon, USA. Association for Computational Linguistics.

Wu, X., Kumar, V., Ross Quinlan, J., Ghosh, J., Yang, Q., Motoda, H., McLachlan, G. J., Ng, A., Liu, B., Yu, P. S., Zhou, Z.-H., Steinbach, M., Hand, D. J., and Steinberg, D. (2007). Top 10 algorithms in data mining. *Knowl. Inf. Syst.*, 14(1):1–37.

Yang, Y. and Liu, X. (1999). A re-examination of text categorization methods. In *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '99, pages 42–49, Berkeley, California, United States. ACM.

Yang, Y. and Pedersen, J. O. (1997). A comparative study on feature selection in text categorization. In *Proceedings of the Fourteenth International Conference on Machine Learning*, ICML '97, pages 412–420, San Francisco, CA, USA.

Yin, Z., Cao, L., Han, J., Zhai, C., and Huang, T. (2011). Geographical topic discovery and comparison. In *Proceedings of the 20th international conference on World wide web*, WWW '11, pages 247–256, Hyderabad, India.

Zong, W., Wu, D., Sun, A., Lim, E.-P., and Goh, D. H.-L. (2005). On assigning place names to geography related web pages. In *ACM/IEEE Joint Conference on Digital Libraries*, pages 354–362.

Readability Classification for German using lexical, syntactic, and morphological features

Julia HANCKE *Sowmya VAJJALA* *Detmar MEURERS*
Seminar für Sprachwissenschaft, Universität Tübingen
{jhancke,sowmya,dm}@sfs.uni-tuebingen.de

ABSTRACT
We investigate the problem of reading level assessment for German texts on a newly compiled corpus of freely available *easy* and *difficult* articles, targeted at adult and child readers respectively. We adapt a wide range of syntactic, lexical and language model features from previous research on English and combined them with new features that make use of the rich morphology of German. We show that readability classification for German based on these features is highly successful, reaching 89.7% accuracy, with the new morphological features making an important contribution.

TITLE AND ABSTRACT IN GERMAN

Lesbarkeitsklassifizierung für das Deutsche mit lexikalischen, syntaktischen und morphologischen Merkmalen

Wir untersuchen das Problem der Lesbarkeitsklassifizierung für deutsche Texte anhand eines neuen Korpus frei zugänglicher Artikel, die einerseits Erwachsene und andererseits Kinder als Zielgruppe haben. Wir adaptieren eine Vielzahl syntaktischer, lexikalischer und *language model* Merkmale aus der englischen Lesbarkeitsforschung und kombinierten sie mit neuen Merkmalen, die sich die ausgeprägte Morphologie des Deutschen zu Nutze machen. Wir zeigen, dass diese Merkmale sehr erfolgreich dazu eingesetzt werden können, deutsche Texte nach ihrer Lesbarkeit zu klassifizieren. In unseren Experimenten erreicht die Klassifikation eine Genauigkeit von 89,7%, wozu die neuen morphologischen Merkmale einen wichtigen Beitrag leisten.

KEYWORDS: Readability, Complexity, Simplification, Second Language Acquisition, Proficiency.

KEYWORDS IN GERMAN: Lesbarkeit, Komplexität, Textvereinfachung, Zweitspracherwerb, Sprachniveau.

Proceedings of COLING 2012: Technical Papers, pages 1063–1080,
COLING 2012, Mumbai, December 2012.

1 Introduction

The last decade has seen an increasing interest in building automatic readability assessment systems. Such systems can help users in finding texts that they can understand, for example by identifying appropriate texts from the huge number of documents available on the web (Bennöhr, 2005; Miltsakaki, 2009; Ott and Meurers, 2010; Collins-Thompson, 2011). This is particularly relevant for first or second language learners as well as for people with intellectual disabilities. Readability classification systems can also be used as a starting point in identifying targets for text simplification, with the goal of ensuring that information can be accessed and understood by a broader audience. The need for such applications is likely to increase given the increasing relevance of information from the web for everyday life.

While early research on readability assessment derived readability formulae from superficial language properties, most current research takes advantage of natural language processing tools to analyze the texts. The resulting features are combined for classification using machine learning. While syntactic, lexical, language model and discourse features have been examined extensively in readability classification, the influence of morphological indicators has received only little attention. This may also be due to the fact that most of the readability research has focused on English. However, there is some recent interest in automatic readability assessment for other languages such as French, German, Italian and Portuguese.

In this paper, we present and evaluate a readability classification approach for German, as a first step towards our overall goal of identifying targets for text simplification. Since there are no existing German corpora that met our needs, we created a two-level readability corpus collected from publicly accessible websites. On this basis, we explore a wide range of syntactic, lexical and language model features derived from previous research on English. In addition, as German has a rich morphology, morphological features may also provide valuable information on the reading level. Hence, we devised a new set of features based on the inflectional and derivational morphology of German. We then conducted classification experiments to examine how well the different feature groups work as indicators of the reading level. We also compared the performance of these feature groups in isolation as well as in combination with each other. To summarize, in this paper, we show that the features used in English readability research can be successfully applied to German, and that the addition of German language-specific morphological features will improve the classification accuracy.

The paper is organized as follows: Section 2 summarizes related work in the field of readability assessment. Section 3 provides information about the origin and nature of our dataset. Section 4 introduces our approach to readability classification including the features we used. Section 5 describes our experimental setup and the results. We conclude this paper with a discussion of the results and directions of future work.

2 Related Work

Research on English readability assessment has a long history spanning several decades (DuBay, 2006). Many traditional readability formulae, such as the Flesch-Kincaid score (Kincaid et al., 1975), relied on easy to calculate approximations of syntactic or lexical complexity, such as number of characters per word or average sentence length. Other early approaches, like the Dale-Chall formula (Chall and Dale, 1995), approximated semantic complexity by using word frequency lists. More recent approaches benefit from advances in natural language processing and machine learning. Si and Callan (2001) and Collins-Thompson and Callan (2004)

used a unigram model for readability classification. Heilman et al. (2007, 2008) combined unigram models with grammatical features and trained machine learning models for readability assessment. Their aim in the context of the *REAP* project (`http://reap.cs.cmu.edu`) was to retrieve reading material for language learners.

Schwarm and Ostendorf (2005) and Petersen and Ostendorf (2009) combined traditional features with syntactic parse tree features and n-gram language models. Their work is based on the *Weekly Reader*, an educational newspaper consisting of articles at four reading levels. Feng (2010) also used the *Weekly Reader* dataset to build classification models with several discourse features alongside parse features, language model features, and traditional features. The discourse features are motivated by the cognitive processes involved in text understanding and underline their focus on finding appropriate texts for people with mild intellectual disabilities. Crossley and McNamara (2011) and the overall *CohMetrix* project (`http://cohmetrix.memphis.edu`) also explored a wide range of cognitively grounded features related to text cohesion and coherence. Vajjala and Meurers (2012) showed that reading level assessment can benefit from Second Language Acquisition (SLA) research. They enriched the lexical and syntactic features from previous approaches by using features derived from measures of the language proficiency of learners.

Readability research on English has ignored morphological features to a large extent. However, with recent interest in readability assessment for languages other than English, the use of features which are relevant for other languages is gaining some prominence. Research on Italian and French readability is taking advantage of the rich verbal morphology of these languages. Dell'Orletta et al. (2011) worked with a corpus of Italian newspaper text at two different reading levels. They used a mixture of traditional, morpho-syntactic, lexical and syntactic features for building a two class readability model for Italian. Among others, their feature set included verbal mood based features, which relied on the rich verbal morphology of Italian. François and Fairon (2012) built their French readability classification model using a text book corpus designed for adult learners of French. They also considered verb tense and mood based text difficulty features along with several other features. Readability assessment was also studied for Portuguese using various lexical, syntactic, discourse and language modeling features derived from English research (dos Santos Marujo, 2009; Aluisio et al., 2010). Lau (2006) utilized the nature of the Chinese script to form several sub-character and character level features in addition to the common word and sentence level features for Chinese readability classification.

The only published research on German readability assessment that we know of is the *DeLite* readability checker (Vor der Brück and Hartrumpf, 2007; Vor der Brück et al., 2008a,b). *DeLite* was built using a human annotated corpus of 500 texts from the municipal domain, such as city ordinances. It was classified into ten levels of difficulty. The corpus includes mostly legal texts and is generally at a higher level of reading difficulty compared to ordinary texts. *DeLite* makes use of a comprehensive set of features that aim to capture readability at lexical, syntactic, semantic and discourse levels. They also considered some morphological indicators, such as the number of nouns that are derived from adjectives or verbs, the complexity of compounds, and the number of acronyms.

Due to the lack of multi-level graded corpora for languages other than English, researchers often built readability models from freely available collections of two or three classes collected from the web. Dell'Orletta et al. (2011), Aluisio et al. (2010), and Klerke and Søgaard (2012)

report on creating and experimenting with such corpora in Italian, Portuguese and Danish respectively. Napoles and Dredze (2010) performed a similar experiment for English with a corpus built from Wikipedia and Simple Wikipedia. Incidentally, all of these groups worked on readability assessment in the context of text simplification.

3 Data: The *GEO-GEOlino* Corpus

Research on the readability classification of English texts has often used the *Weekly Reader* as a gold standard. An established resource that could be compared to this dataset does not exist for German. The only existing German readability corpus is the one collected for the readability checker *DeLite*, but as discussed in the previous section, it is a domain specific collection, consisting mostly of legal texts from the municipal domain. Hence, we created our own two class corpus (*easy* vs. *difficult*) with articles collected from the web, based on the assumption that texts written for children are easier to read than those for adults.

We crawled articles from the websites of two related German magazines, *GEO* (`http://www.geo.de`) and *GEOlino* (`http://www.geolino.de`), published by Gruner & Jahr. *GEO* is a monthly magazine containing articles in the domains of nature, culture and science. *GEOlino* is a magazine on similar topics by the same publisher, but it is targeted at children from age 8–14. *GEOlino* it is not a simplified version of *GEO*; the content is specifically created for child readers.

Table 1 shows the distribution of the articles we crawled across all topics. The overall corpus we collected consists of 4603 articles from both websites.[1]

	GEOlino			GEO		
Topics:	**Num. tokens**	**Num. sentences**	**Num. articles**	**Num. tokens**	**Num. sentences**	**Num. articles**
Nature	189 004	13 976	321	877 920	54 300	1 459
Human	412 769	33 497	901	443 221	29 482	662
Technology	57 819	4 356	83	204 891	12 674	392
Culture	–	–	–	442 888	30 748	463
Creative	169 800	12 354	322	–	–	–
Total	829 392	65 183	1 627	1 968 920	127 204	2 976

Table 1: Composition of the *GEO-GEOlino* corpus

For the experiments discussed in this paper, we randomly selected an equal number of documents from each of the topics that existed in both *GEO* and *GEOlino*: 321 articles from Nature, 662 from Human and 83 from Technology. We cleaned the data obtained from the web by removing all html markup, meta data, and non-text content. We also eliminated duplicate articles. We then tagged the corpus using a Java interface[2] to the RFTagger (Schmid and Laws, 2008), a statistical tagger that provides a fine grained morphological analysis. The tagged articles, mapped to the Stuttgart-Tübingen Tagset (STTS) for German, were then parsed with the Stanford Parser for German (Rafferty and Manning, 2008), which comes with a German model trained on NEGRA.[3]

4 Features

We modeled readability using five groups of features: features from traditional readability formulas, lexical features, syntactic features, language model features, and morphological features. For the first three groups, we essentially adapted the English features described by

[1] Contact us by email if you are interested in using this corpus for non-commercial research purposes.
[2] `http://www.sfs.uni-tuebingen.de/~nott/rftj-public/`
[3] `http://www.coli.uni-saarland.de/projects/sfb378/negra-corpus/negra-corpus.html`

Vajjala and Meurers (2012) to German. Given the effectiveness of language models in previous work on English (e.g., Petersen and Ostendorf, 2009; Feng, 2010), we included language model features as our fourth group. In the fifth group, we explored new features that encode aspects of the inflectional and derivational morphology and compounding of German.

4.1 Traditional Features (TRAD)

The traditional features group we used includes the average sentence length in words, the average number of characters, and syllables per word. These properties have often been used in traditional readability formulae (e.g., Kincaid et al., 1975). Although they do not analyze readability at a deeper level, they have been popular in the English readability literature for a long time. They also constitute a useful baseline against which to interpret the effect of broader and deeper analysis in current readability classification.

4.2 Lexical Features (LEX)

Vajjala and Meurers (2012) employed the lexical richness measures that were originally developed for judging the proficiency of second language learners (Lu, 2011) for English readability classification. We adapted these features for German and added some further features we considered worth exploring. We added the noun token ratio, the verb token ratio and the verb-noun token ratio as additional lexical features, with the hypothesis that easy documents will include fewer nominalizations compared to difficult documents. Additionally, we added the ratio of *sein* to verbs and *haben* to verbs. The list of all implemented lexical features is shown in Table 2. As the class of lexical words (Lex) mentioned in several of the formulas, we used nouns, adjectives, adverbs, and full verbs – i.e., in terms of the STTS tagset: AD.*, N.*, VV.*.

Lexical Richness Features from SLA	**Other Lexical Features**
Type-Token Ratio	*sein* to Verb Token Ratio
Root Type-Token Ratio	*haben* to Verb Token Ratio
Corrected Type-Token Ratio	Avg. Num. Characters per word
Bilogarithmic Type-Token Ratio	Avg. Num. Syllables per word
Uber Index $= \log(Typ^2)/\log(Tok/Typ)$	Verb Token Ratio $= Tok_{Verb}/Tok$
Measure of Textual Lexical Diversity (McCarthy, 2005)	Noun Token Ratio
Lexical Density $= Tok_{Lex}/Tok$	Verb-Noun Token Ratio
Lexical Word Variation $= Typ_{Lex}/Tok_{Lex}$	
Noun Variation $= Typ_{Noun}/Tok_{Lex}$	
Adjective Variation, Adverb Variation	
Modifier Variation $= (Typ_{Adj} + Typ_{Adv})/Tok_{Lex}$	
Verb Variation 1 $= Typ_{Verb}/Tok_{Verb}$	
Verb Variation 2 $= Typ_{Verb}/Tok_{Lex}$	
Squared Verb Variation 1 $= Typ^2_{Verb}/Tok_{Verb}$	
Corrected Verb Variation 1 $= Typ_{Verb}/\sqrt{2Tok_{Verb}}$	

Table 2: The lexical features we implemented

4.3 Syntactic Features (SYN)

For the syntactic features, we adapted a range of parse tree based features from English readability assessment research to German. The features include the average parse tree height

and the average length and number of NPs, VPs and PPs per sentence (Petersen and Ostendorf, 2009; Feng, 2010). Following Vajjala and Meurers (2012), we also adapted proficiency measures from SLA (Lu, 2010), including various ratios that try to capture level of embedding and coordination, length of production unit, and relationships between specific structures. Table 3 lists all the syntactic features we implemented.

Syntactic Features from SLA	**Other Syntactic Features**
Avg. length of a clause	Num. NPs per sentence
Avg. length of a sentence	Num. VPs per sentence
Avg. length of a T-unit	Num. PPs per sentence
Num. of Clauses per Sentence	Num. VZs per sentence
Num. of T-Units per sentence	Avg. length of a NP
Num. of Clauses per T-unit	Avg. length of a VP
Num. of Complex-T-Units per T-unit	Avg. length of a PP
Dependent Clause to Clause Ratio	Num. Dependent Clauses per sentence
Dependent Clause to T-unit Ratio	Num. Complex-T units per sentence
Co-ordinate Phrases per Clause	Co-ordinate Phrases per sentence
Co-ordinate Phrases per T-unit	Avg. parse tree height
Complex Nominals per Clause	
Complex Nominals per T-unit	
Verb phrases per T-unit	

Table 3: The syntactic features we implemented

Three basic production units, *sentences*, *clauses*, and *T-Units* are relevant for computing these measures. We adapted their definitions from those used in the SLA literature (Lu, 2010). A *sentence* is a group of words delimited with a punctuation mark (period, question mark, exclamation mark, or quotation mark). We implemented sentence splitting using the Stanford Document Processor with the default tokenizer factory.

Clauses for English are characterized as structures that contain a subject and a finite verb (Hunt, 1965). Different from English, German allows subjectless sentences, so we consider all maximal projections headed by a finite verb, as well as elliptical constructions where the finite verb is omitted. One word exclamations such as *Stop!* are excluded. In the parsed data, this notion of a *clause* corresponds to the category S of the NEGRA annotation scheme (Skut et al., 1997).

T-Units are defined as "one main clause plus any subordinate clause or non clausal structure that is attached to or embedded in it" (Hunt 1970, p. 4; cf. Lu, 2010). Only independent clauses (including their dependents) count as a T-unit. Consider, for example, the sentence in (1).

(1) Tom fragt Maria, ob sie die Wahrheit sagt, aber sie antwortet nicht.
Tom asks Maria whether she the truth says but she answers not
'Tom asks Maria, whether she says the truth, but she does not answer.'

Figure 1 shows the parse tree of this example sentence. It includes three clauses (shown by the category S) but only two T-Units (indicated by rectangles). One of the clauses is a dependent clause (underlined) and the T-Unit containing it is therefore a complex T-Unit.

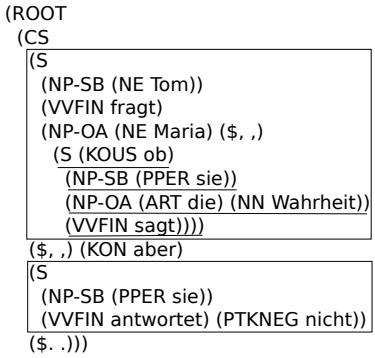


Figure 1: Parse tree of example sentence (1)

At the phrasal level, *coordinated phrases* are defined as coordinated adjective, adverb, noun, and verb phrases. *Verb phrases* include non-finite as well as finite verb phrases. Finally, *complex nominals* are nouns with an adjective, possessive, possessive, prepositional phrase, relative clause, participle, or appositive; nominal clauses, gerunds and infinitives in subject position are also included.

In addition to adapting the patterns and categories for German, we also added VZs ('zu'-marked infinitives) per phrase as a feature. In the so-called incoherent constructions in German (Bech, 1955; Meurers, 2000), the verb always appears in the 'zu'-infinitival form and the phrase it heads is similar in function to a subordinate clause. This naturally is only a coarse approximation of clause-like infinitival constructions given that many verbs selecting 'zu'-infinitivals can alternatively appear in non-clausal, coherent constructions.

On the practical side, TregEx (Levy and Andrew, 2006) was used to extract the patterns that identify the relevant syntactic structures. Lu (2010) created such patterns for English. We modified those patterns to suit the German parse tree structures and categories. As an example, (2a) shows the TregEx pattern for coordinated phrases, for which (2b) is an example structure from our corpus (translating to *parents and children*). (3a) shows the regular expression for complex nominals and (3b) a corpus instance matching that pattern (*a spoiled child*).

(2) a. CAC|CAVP|CNP|CVP
b. (CNP (NN Eltern) (KON und) (NN Kindern))

(3) a. @NP|NN|NE < S | < @AP | < @PP | < @PP | < ADJA|ORD
b. (NP (ART ein) (ADJA verwöhntes) (NN Kind))

4.4 Language Model Features

Following Petersen and Ostendorf (2009), we chose a separate data set for training the language models. For building the *easy* language model, we collected 2000 articles from *News4Kids* (`http://www.news4kids.de`), a German website which adapts news found on the web for children. The *difficult* language model was created using 2000 articles from the website of the

German news channel *NTV* (`http://www.n-tv.de`). For these web texts, we followed the same preprocessing routine as explained in Section 3.

Schwarm and Ostendorf (2005) and Feng (2010) suggested that mixed models combining words and parts of speech are more effective for readability assessment than simple word based models. But Petersen and Ostendorf (2009) reached the opposite conclusion. We experimented with both types of models. For preparing the mixed models, we followed the procedure suggested in Petersen and Ostendorf (2009). We trained a bag-of-words classifier with our language modeling dataset (*News4Kids* and *NTV*) and employed Information Gain (Yang and Pedersen, 1997) for feature selection. All words below an empirically determined threshold were replaced by their part of speech.

We used the SRI Language Modeling toolkit (Stolcke, 2002) for training unigram, bigram and trigram models on the *words-only* and *mixed word/part of speech* corpora for each of our two reading levels. This resulted in twelve language models. For all models, we selected Kneyser-Ney smoothing (Chen and Goodman, 1999) as smoothing technique. The perplexity scores from all twelve language models were used as features for building the classifier. These features are listed in Table 4.

Level of Difficulty	Word Based Model	Mixed Model
Easy	Unigram perplexity	Unigram perplexity
	Bigram perplexity	Bigram perplexity
	Trigram perplexity	Trigram perplexity
Difficult	Unigram perplexity	Unigram perplexity
	Bigram perplexity	Bigram perplexity
	Trigram perplexity	Trigram perplexity

Table 4: The twelve perplexity scores used as language model features

4.5 Morphological Features (MORPH)

German has a rich inflectional and derivational morphology. Although not to the same extent as Romance languages, German uses inflectional morphemes to convey a range of grammatical meanings. For example, person and number of a verb are indicated by inflectional endings (e.g., *ich kaufe* [*I buy*], *du kaufst* [*you buy*]). On the nominal side, German has four cases and nouns fall into several different declension paradigms. Case information is sometimes carried by the articles and sometimes by the noun.

German derivational morphology uses various prefixes and suffixes. Nominalizations with a suffix (*regieren* [*govern*] – *Regierung* [*government*]) or without (*laufen* [*to run*] – *der Lauf* [*the run*]) are very common. Compounding is another productive morphological process in German.

Since these three morphological processes may reflect distinctions of relevance for identifying the reading level of a text, we included them in our experiments.

4.5.1 Inflectional Morphology of the Verb (INFLV) and the Noun (INFLN)

We wanted to investigate if the verbal tense and mood encodes relevant information about a text's difficulty. Hence, we examined a broad range of features related to the inflectional properties of the verb including person, tense, mood and type of verb (finite, non-finite, auxiliary). Additionally, we included the case properties of nouns as features. The case of a

nominal argument, for example, reflects the nature and complexity of the subcategorization frame of the head that selects it. The list of all our features related to inflectional morphology is shown in Table 5.

Verb (INFLV)	**Noun** (INFLN)
Num. infinitive Vs / Num. Vs	Num. accusative Ns / Num. Ns
Num. participle Vs / Num. Vs	Num. dative. Ns / Num. Ns
Num. imperative Vs / Num. Vs	Num. genitive Ns / Num. Ns
Num. present tense Vs / Num finite Vs	Num. nominative Ns / Num. Ns
Num. past tense Vs / Num. finite Vs	
Num. 1st person Vs / Num. finite Vs	
Num. 2nd person Vs / Num. finite Vs	
Num. 3rd person Vs / Num. finite Vs	
Num. subjunctive Vs / Num. finite Vs	
Num. finite Vs / Num. Vs	
Num. modal Vs / Num. Vs	
Num. auxiliary Vs / Num. Vs	
Num. Vs / Num. S	

Table 5: The features based on inflectional morphology

4.5.2 Derivational Morphology of the Noun (DERIV)

Two features based on derivational morphology were previously used for German. Vor der Brück et al. (2008b) measured the number of words that are derived from a verb or an adjective. Derived nouns are linguistically more complex than simple nouns. Additionally, derivational suffixes often differ for words which are native German words and lexical items that have come into German from other languages, e.g., *linguist* can be translated into German as either *Lingu**ist*** or as *Sprachwissenschaft**ler***.

We included relatively fine grained derivational properties by taking into account the distribution of a number of individual suffixes. We manually compiled a set of suffixes from a comprehensive overview of native and foreign suffixes (Fleischer and Barz, 1995). For each suffix, we listed all of the different gender and number forms. The counts are accumulated per type of suffix. For example, if there are two instances of *-ismus* and one instance of *-ismen* (the plural form), then a sum of three instances is recorded for *-ismus*. To avoid counting instances of homomorphic nouns as suffixes (e.g., *Ei* [*egg*] vs. suffix *-ei*), we only considered polysyllabic words for the suffix analysis. A list of all the suffixes we used is shown in Table 6.

All occurrences of each suffix (and its forms) were counted per document. Three different ratios can then be generated for each suffix. Let S be the count of a given suffix from Table 6 for a given document, and let T, N and DN be the number of tokens, nouns, and derived nouns in that document. We defined derived noun as a noun which ends in one of the suffixes listed in Table 6. The suffix token ratio (STR), suffix noun ratio (SNR) and suffix derived noun ratio (SDNR) can then be calculated as follows:

- **Suffix Token Ratio (STR)** $= \frac{1}{T}S$
- **Suffix Noun Ratio (SNR)** $= \frac{1}{N}S$
- **Suffix Derived Noun Ratio (SDNR)** $= \frac{1}{DN}S$

suffix	further suffix forms
ant	anten, antin, antinnen
arium	arien
ast	asten, astin, astinnen
at	ate
ator	atoren, atorin, atorinnen
atur	aturen
ei	eien
er	erin, erinnen
ent	ents
enz	enzen
eur	eure, eurin, eurinnen
heit	heiten

suffix	further suffix forms
ist	isten, istin, istinnen
ion	ionen
ismus	ismen
ität	itäten
keit	keiten
ling	lingen
nis	nisse
schaft	schaften
tum	tümer
ung	ungen
ur	
werk	werke
wesen	

Table 6: List of German derivational suffixes used

For example, consider the sentence in (4) and assume that this is a one sentence document.

(4) Die Regier**ung** muss die Nutz**ung** der landwirtschaftlichen Flächen mit großer
the government has to the use of agricultural areas with great
Präzis**ion** planen.
precision plan
'The government has to plan the use of the agricultural areas with great precision.'

In this example, there are two instances of the suffix *-ung* and one instance of the suffix *-ion* (shown in bold). *Regierung* and *Nutzung* are both nominalizations of native words with native origin, and *Präzision* is of foreign origin. This one sentence document includes twelve tokens, four nouns and three derived nouns, so one obtains $\text{STR(ung)} = 2/12$, $\text{SNR(ung)} = 2/4$ and $\text{SDNR(ung)} = 2/3$. For the suffix *-ion* we get $1/12$, $1/4$ and $1/3$, respectively. The values for all other suffixes are zero. In addition to the specific suffix ratios, we also computed the overall ratio of derived nouns to nouns.

4.5.3 Nominal Compounds (COMP)

Compounds are frequent in German and compounding is a productive mechanism of word formation in German Fleischer and Barz (1995, p. 85). Vor der Brück et al. (2008b) included two compounding features into the classifier for the readability checker *DeLite*: the number of words that form a compound and the number of semantic concepts that are incorporated into a compound. In our approach, we considered the ratio of compound nouns to all nouns and the average number of words in a compound as our compounding based features. For identifying compounds and splitting them into their parts we used JWordSplitter 3.4[4].

5 Experiments and Results

We performed our classification experiments using the *Geo-GeoLino* dataset described in Section 3. We employed the *WEKA* (Hall et al., 2009) implementation of the Sequential Minimal

[4]http://www.danielnaber.de/jwordsplitter

Optimization (SMO) algorithm to create our classification models. For all our classification experiments, we report the overall accuracy after 10-fold cross validation.

We first report the results with language modeling (LM) and the morphological feature groups separately. We then report the performance of each of the other feature groups introduced in Section 4 separately, before turning to combinations.

5.1 Language Modeling

In previous research on the use of language model features for readability assessment, it remained unclear if word based models or mixed word/part of speech models are more effective. While Schwarm and Ostendorf (2005) reported that the mixed models performed better, this was not confirmed by Petersen and Ostendorf (2009). Schwarm and Ostendorf (2005) trained the language models on the same dataset they used for training their classifier, whereas Petersen and Ostendorf (2009) used different datasets for training the language models and the classifier. Feng (2010), who also trained the language models on the same data as the readability classifier, reports that the mixed models outperformed the purely word based models.

As we discussed in section 4.4, we followed Petersen and Ostendorf (2009) in training the language model on a separate data set, to test whether the information picked up about easy vs. difficult texts generalizes across corpora. We trained one classifier on the purely word based models and another one on the mixed model. We combined all perplexity scores from both word based and mixed models. In our experiments, the mixed model (71%) performed only slightly better than the word based model (70.8%). However, an experiment using all twelve perplexity scores as features showed that a combination of mixed and word based models improved classification accuracy (77.6%). So we kept all twelve scores in the feature group for further experiments.

5.2 Morphological Features

We investigated the morphological features in detail given that most of them had not been employed before. In the case of derivational morphology, as described in Section 4.5.2, we calculated three ratios: suffix token ratio, suffix noun ratio and suffix derived noun ratio for each of the 25 derivational suffixes and also calculated the overall ratio of derived nouns to nouns, leading to a total of 76 features. We first performed classification experiments with all three suffix ratio feature subsets separately. The suffix token ratio features performed best (76.6%), followed by suffix noun ratio features (74.4%) and suffix derived noun ratio features (74.0%). However, a combination of all the derivational suffix based features achieved a higher accuracy (78.5%) than the individual subsets. The verbal inflection (INFLV) and nominal inflection (INFLN) features (Table 5) alone achieved accuracies of 74.3% and 67.2%, respectively. Combining all the inflectional features resulted in a classifier that performed much better, at 79.0%.

Table 7 shows the results for each of our morphological feature groups. The derivational features (DERIV) achieved the highest accuracy followed by the features that use verbal inflection (INFLV) and nominal inflection (INFLN). The nominal compound features (COMP) were the least effective predictors among the morphological groups.

The combined feature set (MORPH) consisting of all four morphological subsets performed with an accuracy 85.4%, which is better than any of the individual subsets. Removing the compounding features from MORPH had no impact on the accuracy.

Feature set	Num. Features	Accuracy
DERIV	76	78.5%
INFLN	4	67.2%
INFLV	13	74.3%
INFLN & INFLV	17	79.0%
COMP	2	56.96%
MORPH	95	85.4%

Table 7: Results for the different types of morphological features

5.3 Most Predictive Features

Apart from the various feature subsets, we also determined the most predictive features using Information Gain. The ten most predictive features (TOP 10) are shown in Table 8. Training a classifier with the TOP10 features resulted in an accuracy of 84.3%.

Feature	Group
Avg. Word Length	Lex/Trad
Num. 2nd person Vs / Num. finite Vs	Morph
Num. Syllables Per Word	Lex/Trad
Num. 3rd person Vs / Num. finite Vs	Morph
Avg. length of a T-unit	Syn
Avg. length of a Sentence	Syn/Trad
Complex Nominals per Clause	Syn
Complex Nominals per T-unit	Syn
Num. PPs per sentence	Syn
Avg. length of a clause	Syn

Table 8: The ten most predictive features according to Information Gain

Most of the features in the TOP 10 belong to the syntactic feature group, but morphology features and some traditional measures are included as well. The dominance of syntactic features, especially of those from SLA research, confirms the conclusions of Vajjala and Meurers (2012) for English that these measures are particularly effective for readability classification. The list also indicates that investigating the usefulness of morphological features for readability classification was well-worth the effort, even at this relatively shallow level of morphological modeling. Note also that the three traditional readability features (TRAD) that we view as a baseline are among the TOP10 features. While these superficial features have little conceptual value, they seem to be good predictors of reading level.

5.4 Results for Feature Groups and Combinations

We trained classifiers with various individual feature groups and some of their combinations. Table 9 summarizes the results for the different classification models. For the feature group combinations, only the most successful combinations are shown.

Feature set	Num. Features	Accuracy
TRAD	3	82.2%
LEX	23	82.1%
SYN	26	76.8%
MORPH	95	85.4%
LM	12	77.6%
SYN & MORPH	120	86.7%
LEX & LM & MORPH	130	89.4%
ALL	155	89.7%
TOP 10	10	84.3%

Table 9: A comparison of all five feature groups

Among the models trained on a single feature group, the morphological classifier (MORPH) performed best with an accuracy of 85.4%. The lexical classifier (LEX) (82.1%) and the baseline classifier (TRAD) (82.2 %) performed almost equally well, but slightly worse than the morphological classifier (MORPH). The classifiers trained on language model features (LM) and syntactic features (SYN) proved to be less effective predictors when taken on their own. However, they proved to be valuable when combined with other feature groups. Our experiments combining different feature groups showed that the syntactic (SYN) and morphological (MORPH) feature groups together were the most predictive two group combination with 86.7 % accuracy. When using three feature groups a combination of lexical (LEX), language model (LM) and morphological (MORPH) features performed best (89.4%).

Overall, the best result was achieved by combining all feature groups (ALL), which resulted in 89.7% accuracy. Compared to the traditional readability measures (TRAD) as baseline (82.2%), our best model improved classification performance by 7.5%. The classifier built with the ten best features (TOP 10) at 84.3% accuracy performed at about the level of the best single feature group (MORPH).

Conclusion and Outlook

As empirical basis of our work, we created the *GEO-GEOlino* corpus, a German corpus with two different reading levels that we collected from magazine articles that were available online. The *easy* reading level consists of the *GEOlino* articles targeted at children, while the *GEO* articles targeted at adults were labeled as *difficult*.

We trained classifiers with syntactic, lexical, and language model features derived from research on English, to see how well they can predict the reading level of German texts. We then introduced language-specific morphological complexity indicators as an additional group of features. We inspected a broad set of inflectional properties for German and for the first time made use of the derivational and inflectional morphology of nouns as features for readability classification of German. The novel morphological features proved to be especially good indicators for reading level, outperforming all other feature groups, when considered in isolation.

While all the individual feature groups except morphological features performed below the baseline, combinations of various feature sets resulted in higher accuracies. The best performance was obtained by combining all features. This achieved an accuracy of 89.7%, which is 7.5% above a baseline classifier using only traditional readability measures.

In terms of outlook, we are investigating how well the trained models generalize to other data sets, for which obtaining more graded reading material for German is an important next step. Going beyond readability classification of entire documents, we want to explore which features are effective not only at the document level but already at paragraph or sentence levels. Being able to identify simple and difficult paragraphs or sentences is relevant for identifying targets for simplification – the next step for our overall goal of building text simplification systems.

Finally, some of the features used in this paper were originally developed as measures of language proficiency in Second Language Acquisition research. Given how well SLA measures of language proficiency (based on texts produced by the learners) work as features for readability classification of native texts, it would be natural to take the features developed for our readability research back to the SLA domain and explore their applicability to classifying the language proficiency of language learners. In addition to quantitatively testing their impact for proficiency classification, strengthening that link could also help with qualitatively interpreting and further refining the different features on the background of SLA insights into stages of language development.

Acknowledgments

We would like to thank the anonymous reviewers for their feedback on the paper, and Niels Ott and Ramon Ziai for the RFTagger Java interface. The research leading to the reported results has received funding through the European Commission's 7th Framework Program under grant agreement number 238405 (CLARA, `http://clara.uib.no`).

References

Aluisio, S., Specia, L., Gasperin, C., and Scarton, C. (2010). Readability assessment for text simplification. In *Proceedings of the NAACL HLT 2010 Fifth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 1–9, Los Angeles, California.

Bech, G. (1955). *Studien über das deutsche verbum infinitum*. Historisk-filologiske Meddelelser udgivet af Det Kongelige Danske Videnskabernes Selskab. Bind 35, no. 2, 1955; Bind 36, no. 6, 1957; Kopenhagen. Reprinted 1983, Tübingen: Max Niemeyer Verlag.

Bennöhr, J. (2005). A web-based personalised textfinder for language learners. Master's thesis, School of Informatics, University of Edinburgh.

Chall, J. S. and Dale, E. (1995). *Readability Revisted: The New Dale-Chall Readability Formula*. Brookline Books.

Chen, S. F. and Goodman, J. T. (1999). An empirical study of smoothing techniques for language modeling. *Computer Speech and Language*, 13:359–394.

Collins-Thompson, K. (2011). Improving information retrieval with reading level prediction. In *SIGIR 2011 Workshop on Enriching Information Retrieval*.

Collins-Thompson, K. and Callan, J. (2004). A language modeling approach to predicting reading difficulty. In *Proceedings of HLT/NAACL 2004*, Boston, USA.

Crossley, S. A. and McNamara, D. S. (2011). Understanding expert ratings of essay quality: Coh-metrix analyses of first and second language writing. *International Journal of Continuing Engineering Education and Life-Long Learning*, 21(2/3):170–191.

Dell'Orletta, F., Montemagni, S., and Venturi, G. (2011). Read-it: Assessing readability of italian texts with a view to text simplification. In *Proceedings of the 2nd Workshop on Speech and Language Processing for Assistive Technologies*, pages 73–83.

dos Santos Marujo, L. C. (2009). Reap.pt (reap-portuguese). Master's thesis, Universidade Tecnica de Lisboa.

DuBay, W. H. (2006). *The Classic Readability Studies*. Impact Information, Costa Mesa, California.

Feng, L. (2010). *Automatic Readability Assessment*. PhD thesis, City University of New York (CUNY).

Fleischer, W. and Barz, I. (1995). *Worbildung der deutschen Gegenwartssprache*. Niemeyer, Tübingen, Germany.

François, T. and Fairon, C. (2012). An "ai readability" formula for french as a foreign language. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*.

Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., and Witten, I. H. (2009). The weka data mining software: An update. In *The SIGKDD Explorations*.

Heilman, M., Collins-Thompson, K., Callan, J., and Eskenazi, M. (2007). Combining lexical and grammatical features to improve readability measures for first and second language texts. In *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics (HLT-NAACL-07)*, pages 460–467, Rochester, New York.

Heilman, M., Collins-Thompson, K., and Eskenazi, M. (2008). An analysis of statistical models and features for reading difficulty prediction. In *Proceedings of the 3rd Workshop on Innovative Use of NLP for Building Educational Applications at ACL-08*, Columbus, Ohio.

Hunt, K. W. (1965). Grammatical structures written at three grade levels. NCTE Research Report No. 3.

Hunt, K. W. (1970). Do sentences in the second language grow like those in the first? *TESOL Quarterly*, 4(3):195–202.

Kincaid, J. P., Fishburne, R. P. J., Rogers, R. L., and Chissom, B. S. (1975). Derivation of new readability formulas (Automated Readability Index, Fog Count and Flesch Reading Ease formula) for Navy enlisted personnel. Research Branch Report 8-75, Naval Technical Training Command, Millington, TN.

Klerke, S. and Søgaard, A. (2012). Danish parallel corpus for text simplification. In *In Proceedings of Language Resources and Evaluation Conference (LREC), 2012*.

Lau, T. P. (2006). Chinese readability analysis and its applications on the internet. Master's thesis, CUHK, Hongkong.

Levy, R. and Andrew, G. (2006). Tregex and tsurgeon: tools for querying and manipulating tree data structures. In *5th International Conference on Language Resources and Evaluation*, Genoa, Italy.

Lu, X. (2010). Automatic analysis of syntactic complexity in second language writing. *International Journal of Corpus Linguistics*, 15(4):474–496.

Lu, X. (2011). The relationship of lexical richness to the quality of esl learners' oral narratives. *The Modern Languages Journal*.

McCarthy, P. M. (2005). *An assessment of the range and usefulness of lexical diversity measures and the potential of the measure of textual, lexical diversity (MTLD)*. PhD thesis, University of Memphis.

Meurers, W. D. (2000). *Lexical Generalizations in the Syntax of German Non-Finite Constructions*. Phil. dissertation, Eberhard-Karls-Universität Tübingen. Published as: Arbeitspapiere des SFB 340, Nr. 145.

Miltsakaki, E. (2009). Matching readers' preferences and reading skills with appropriate web texts. In *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics: Demonstrations Session*, EACL '09, pages 49–52, Stroudsburg, PA, USA. Association for Computational Linguistics.

Napoles, C. and Dredze, M. (2010). Learning simple wikipedia: a cogitation in ascertaining abecedarian language. In *Proceedings of the NAACL HLT 2010 Workshop on Computational Linguistics and Writing: Writing Processes and Authoring Aids*, CL&W '10, pages 42–50, Stroudsburg, PA, USA. Association for Computational Linguistics.

Ott, N. and Meurers, D. (2010). Information retrieval for education: Making search engines language aware. *Themes in Science and Technology Education. Special issue on computer-aided language analysis, teaching and learning: Approaches, perspectives and applications*, 3(1–2):9–30.

Petersen, S. E. and Ostendorf, M. (2009). A machine learning approach to reading level assessment. *Computer Speech and Language*, 23:86–106.

Rafferty, A. N. and Manning, C. D. (2008). Parsing three German treebanks: lexicalized and unlexicalized baselines. In *Proceedings of the Workshop on Parsing German*, PaGe '08, pages 40–46, Stroudsburg, PA, USA. Association for Computational Linguistics.

Schmid, H. and Laws, F. (2008). Estimation of conditional probabilities with decision trees and an application to fine-grained pos tagging. In *COLING '08 Proceedings of the 22nd International Conference on Computational Linguistics*, volume 1, pages 777–784, Stroudsburg, PA. Association for Computational Linguistics.

Schwarm, S. and Ostendorf, M. (2005). Reading level assessment using support vector machines and statistical language models. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL-05)*, pages 523–530, Ann Arbor, Michigan.

Si, L. and Callan, J. (2001). A statistical model for scientific readability. In *Proceedings of the 10th International Conference on Information and Knowledge Management (CIKM)*, pages 574–576. ACM.

Skut, W., Kreen, B., Brants, T., and Uszkoreit, H. (1997). An annotation scheme for free word order languages. In *Proceedings of the Fith Conference on Applied Natural Language*, Washington, D.C.

Stolcke, A. (2002). SRILM – an extensible language modeling toolkit. In *Proceedings of ICSLP*, volume 2, pages 901–904, Denver, USA.

Vajjala, S. and Meurers, D. (2012). On improving the accuracy of readability classification using insights from second language acquisition. In Tetreault, J., Burstein, J., and Leacock, C., editors, *Proceedings of the 7th Workshop on Innovative Use of NLP for Building Educational Applications (BEA7) at NAACL-HLT*, pages 163—-173, Montréal, Canada. Association for Computational Linguistics.

Vor der Brück, T. and Hartrumpf, S. (2007). A semantically oriented readability checker for german. In *Proceedings of the 3rd Language & Technology Conference*.

Vor der Brück, T., Hartrumpf, S., and Helbig, H. (2008a). A readability checker with supervised learning using deep syntactic and semantic indicators. *Informatica*, 32(4):429—-435.

Vor der Brück, T., Helbig, H., and Leveling, J. (2008b). The readability checker delite. Technical Report Technical Report 345-5/2008, Fakultät für Mathematik und Informatik, FernUniversität in Hagen.

Yang, Y. and Pedersen, J. O. (1997). A comparative study on feature selection in text categorization. In *Proceedings of the Fourteenth International Conference on Machine Learning*, ICML '97, pages 412–420, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Walk-based Computation of Contextual Word Similarity

Kazuo Hara[1] *Ikumi Suzuki*[1] *Masashi Shimbo*[2] *Yuji Matsumoto*[2]
(1) National Institute of Genetics, Mishima, Shizuoka, Japan
(2) Nara Institute of Science and Technology, Ikoma, Nara, Japan
kazuo.hara@gmail.com, suzuki.ikumi@gmail.com, shimbo@is.naist.jp,
matsu@is.naist.jp

ABSTRACT
We propose a new measure of semantic similarity between words in context, which exploits the syntactic/semantic structure of the context surrounding each target word. For a given pair of target words and their sentential contexts, labeled directed graphs are made from the output of a semantic parser on these sentences. Nodes in these graphs represent words in the sentences, and labeled edges represent syntactic/semantic relations between them. The similarity between the target words is then computed as the sum of the similarity of walks starting from the target words (nodes) in the two graphs. The proposed measure is tested on word sense disambiguation and paraphrase ranking tasks, and the results are promising: The proposed measure outperforms existing methods which completely ignore or do not fully exploit syntactic/semantic structural co-occurrences between a target word and its neighbors.

KEYWORDS: contextual word similarity, word sense disambiguation, paraphrase, kernel method.

Proceedings of COLING 2012: Technical Papers, pages 1081–1096,
COLING 2012, Mumbai, December 2012.

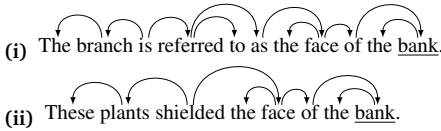
1 Introduction

Word ambiguity poses a great difficulty in natural language processing (Deerwester et al., 1990; Berger and Lafferty, 1999; Navigli, 2009). In document classification, for example, polysemous words may provide spurious evidence for misclassifying documents of different categories into a single category. Synonyms may also cause problems; without the knowledge of two words being synonyms, classifiers might be unable to detect similarity of documents in which they appear.

These problems can be alleviated with the help of contextual similarity (Jurafsky and Martin, 2008, Section 20.7). Polysemous words have identical surface forms, but the contexts in which they appear are often distinct; and even though synonyms have different surface forms, they are expected to appear in similar context.

There has been a volume of work investigating semantic similarity of words in context. However, syntactic or semantic *structure* in the context has not been fully explored. The methods representing context as a *bag of words* or a *bag of n-grams* (Schütze, 1998; Reisinger and Mooney, 2010; Erk and Padó, 2010; Dinu and Lapata, 2010; Giuliano et al., 2009) ignore the underlying syntactic/semantic structure. Recent studies on compositional semantics of words (Mitchell and Lapata, 2008; Erk and Padó, 2008; Thater et al., 2010) take advantage of structure in context, but they only use information of words that directly stand in a predicate-argument relation with the target word.

Consider the following sentences (i) and (ii), shown with the word dependency relations.



In sentence (i), the noun (homonym) *bank* means *"a financial institution"* whereas the one in sentence (ii) means *"the slope of land adjoining a river."*

If these sentences are treated as bags of words, we see that the words unique to (i) and (ii) are respectively {*branch*, *is*, *refer*} and {*plant*, *shield*}, excluding functional words. However, these sets of words are not likely to be sufficient to discriminate the sense of the two *bank*s. The bag-of-*n*-grams representation (Giuliano et al., 2009) collects *n*-grams containing a target word from the context. This representation is still short on distinguishing the senses of the two *bank*s above, as the *n*-grams around them are completely identical up to $n = 5$; i.e., the 5-grams surrounding *bank* are *"the face of the bank"* in both sentences. Recent methods (Mitchell and Lapata, 2008; Erk and Padó, 2008; Thater et al., 2010) exploit direct syntactic/semantic relations with target words, but these methods would still fail in the above examples, because the direct structural neighbors of *bank*s are the same in the two contexts.

On the other hand, syntactic/semantic structural co-occurrences of multiple words, obtained from the dependency relations, seem effective for this example. Consider structural collocations *referred* → *as* → *face* → *of* → *bank* and *shielded* → *face* → *of* → *bank*, which can be obtained if we follow multiple dependency arrows in the dependency structure. The problem is how to compute such co-occurrences efficiently from given syntactic/semantic structures.

Contributions. In this paper, to fully exploit syntactic/semantic co-occurrences of words in contexts, we represent a target word in a context as a *bag of walks* on the *parse graph* (Section 3). In this graph, nodes represent words, and edges represent syntactic/semantic relations between nodes. Such a graph can be readily obtained from the output of state-of-the-art semantic parsers.

We further define the *bag-of-walks* kernel between graph nodes (Section 4). With this kernel, word similarity is calculated as the sum of the similarity of walks starting from target words in their respective parse graphs. Here, the similarity of two walks is the product of the similarity of individual nodes and edges visited during the walks. Since nodes represent words and edges represent syntactic/semantic relations between them, this similarity computation takes into account syntactic/semantic structural co-occurrence of multiple words in context.

We verify effectiveness of the proposed method in two experiments: one on word sense disambiguation and the other on paraphrase ranking (Section 5).

2 Related work

The simplest representation of a context is to regard it as a *bag of words*, or a vector holding the frequency of words co-occurring with the target word in the given context. We call this vector *local context vector* in this paper. Depending on the task, context can be a fixed number of words around the target word, or the sentence or paragraph in which it appears.

We can regard a local context vector as representing a particular meaning the target word conveys in the given context. However, these vectors may be too sparse to provide enough information, especially when the available context is short. Hence, recent studies have focused on the way to make a richer representation of context utilizing information extracted from external resources, such as corpora.

The majority of research along this line utilizes a context-free representation of a word, called *type vector* (Erk and Padó, 2010). The type vector of a word, in its simplest form, is built by aggregating local context vectors made for individual occurrences of the word in the corpus. Thus, it does not represent a specific context of a word, but is a general representation of a word viewed as a type. Still, type vectors can be used as building blocks of *contextualized* feature vectors (Thater et al., 2010) representing the context around the target word.

Schütze (1998) proposed the *word vectors* as one such contextualized feature vector. The word vector of a target word is the sum of the type vectors of the words appearing in its surrounding context. Because it is an aggregate of dense type vectors of different words, we can expect a word vector to hold information richer than a naive, bag-of-words local context vector. More elaborate contextualization has been proposed recently, which combines (e.g., via component-wise vector multiplication) the type vector of the target word with those of the words occurring in the context (Mitchell and Lapata, 2008) in a compositional way. Erk and Padó (2008) and Thater et al. (2010) took a similar compositional approach, but they made type vectors by taking the semantic structure into account, i.e., by counting frequency of words in the corpus that participate in syntactic or semantic relations with the target word.

Erk and Padó (2010) proposed a corpus-based method that does not rely on type vectors. They first make a bag-of-words local context vector $\mathbf{v}$ for a given target word, in a usual manner. They then collect k instances of the target word in a corpus whose contexts are most similar to the one for the target instance (with respect to a predefined similarity measure of contexts). The

local context vectors for these k instances are then added to the original local context vector $\mathbf{v}$ to make it denser. The idea behind this method is that because these k instances appeared in a context similar to that of the target instance, they are likely to have the same sense.

As we have seen above, even the most recent methods do not use syntactic/semantic structure of context at all, or make only limited use of it; namely, only the words that directly participate in a relation (e.g., predicate-argument) with a target word are taken into account.

3 Bag-of-walks representation of word in context

In this section, we introduce the *bag-of-walks* representation of words in context. This representation encodes syntactic/semantic co-occurrences of words around a target word as a collection of walks in a *parse graph*, a graph that can be obtained from the structure output by syntactic/semantic parsers.

We first describe how to make a parse graph from a parser output, and then define a natural random walk model on this graph. The parse graph and the random walk model will be used to define the "bag-of-walks kernels" in Section 4, which allow us to evaluate the similarity of two contexts represented as bags of walks.

For brevity, we assume below that the context is a sentence containing a target word. Smaller fragments (e.g., phrases) can also be used in place of sentences, as long as the relations between words in a fragment can be obtained. For instance, we could use a subtree (corresponding to a phrase) extracted from the dependency tree of an entire sentence.

3.1 Parse graphs

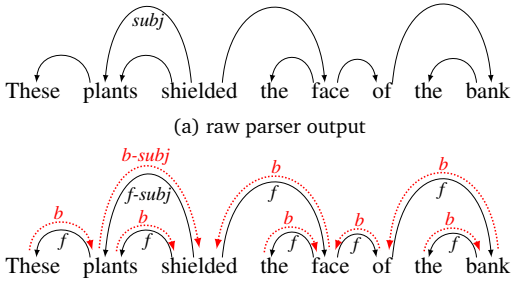
The *parse graph* can be obtained from the output of (semantic) dependency parsers.

Basic dependency parsers output a tree in which nodes are the words in the input sentence and (directed) edges between them represent syntactic head-dependent (dependency) relations. A dependency may be expressed as a directed edge either from the head to the dependent, or from the dependent to the head. The choice is arbitrary, but both head-to-dependent relations and dependent-to-head relations potentially provide useful features for semantic similarity of words. The same should be true for other types of syntactic/semantic relations output by more advanced parsers, such as the *"semantic subject"* of a verb in passive voice. Hence, we represent each individual relation output by the parser as two distinct edges having the same pair of end nodes (words), but with direction opposite to each other. The two edges have the same label representing the type of the relation, but one of them is prefixed *"forward"* (or *"f"* for short) and the other is prefixed *"backward"* (or *"b"*), so that the two edges can be distinguished by the label. Thus for example, if the parser outputs a relation called *"semantic subject"* between two nodes, one edge will be labeled *"forward semantic subject"*, and the other, opposite direction edge will have the label *"backward semantic subject."*

We call a labeled directed graph created in this way a *parse graph*. See Figure 1 for illustration.

3.2 Walks in a graph

We now consider walks in the parse graph starting from the target word. Given a graph G, a walk is a series of nodes and edges, formally defined as follows. Let $\mathrm{Out}(w)$ denote the set of outgoing edges from node w, and $\mathrm{sink}(r)$ denote the sink node of directed edge r. A *walk* $z = (w_0, r_1, w_1, r_2, \ldots, r_t, w_t)$ in graph G is an alternating series of nodes $(w_0, \ldots, w_t)$ and edges



(a) raw parser output

(b) after conversion

Figure 1: Illustration of parse graph construction. (a) The raw dependency structure extracted from the output of the Enju parser. Notice multiple edges from "shielded" to "plants"; the parser outputs a semantic relation named "*subj*" in addition to syntactic dependencies (shown without edge labels). (b) Each relation in the parser output is converted to two edges, to allow "forward" and "backward" (shown as a dotted red arrow) traversal of each relation. In the figure, "*b*" and "*f*" are the shorthands for "backward" and "forward," respectively.

$(r_1, \ldots, r_t)$ in G such that w_{i-1} and w_i are the source and sink of edge r_i; i.e., $r_i \in \mathrm{Out}(w_{i-1})$ and $w_i = \mathrm{sink}(r_i)$ for every $i = 1, \ldots, t$. We call t the *length* of z and denote it by $|z|$. A walk may contain duplicate nodes and edges. Thus it is possible that $w_i = w_j$ or $r_i = r_j$ for some i and j, $j \neq i$.

A walk starting from a target word (node) on a parse graph represents the structural co-occurrence of multiple words around the target word. For example, an entire predicate-argument structure consisting of a predicate p and arguments $\{a_1, a_2, \ldots, a_n\}$ may be represented by a walk $p \to a_1 \to p \to a_2 \to p \to \cdots \to p \to a_n$, traveling between the predicate and each of its arguments alternately; as mentioned above, a walk may visit the same nodes multiple times.

In this way, we represent the context surrounding a target word as a bag of all walks (of bounded length) starting from the target word on a parse graph; we call this a *bag-of-walks* representation of context.

We now introduce a random walk model on a parse graph, as a way to systematically associate a probability (or weight) to each walk in the "bag." In this model, a walk starts from a node corresponding to the target word, and the following process is repeated: (1) Flip a biased coin (with head probability γ). If it comes out heads, terminate the walk. If it comes out tails, proceed to step 2. (2) Choose an edge outgoing from the current node uniformly at random. (3) Follow the edge to the next node. After this process is repeated L times, the walk is terminated, but as step 1 shows, it may also be terminated prematurely at each step with a constant probability γ.

This random walk model assigns probability to each walk of length at most L, starting from the target word in the parse graph. Let w be a node in G, and let $z = (w_0, r_1, w_1, r_2, \ldots, r_{|z|}, w_{|z|})$, be a walk. By the way the random walk model is defined, it is easy to verify that the probability

of taking walk z from node w, written $p(z \mid w)$, is given by

$$p(z \mid w) = \begin{cases} 0, & \text{if } w \neq w_0 \text{ or } |z| > L; \\ \gamma^{I[|z| \neq L]}(1-\gamma)^{|z|} \prod_{i=1}^{|z|} \frac{1}{|\mathrm{Out}(w_{i-1})|}, & \text{otherwise;} \end{cases} \quad (1)$$

where $I[X]$ is an indicator function taking 1 if proposition X holds, or 0 otherwise. Recall that γ is the predefined stopping probability.

As we can see from Eq. (1), walk probability can be tuned by the model parameters L and γ; e.g., the larger the stopping probability γ, the smaller the probability of taking longer walks. These parameters control how much weight should be placed on long-distance dependencies between words, and how far we should look for such dependencies. Section 4 defines the bag-of-walks kernels using the walk probability $p(\cdot)$ as the weight of individual walks.

4 Contextual word similarity via bag-of-walks kernels

This section presents a new method of computing similarity between words occurring in different contexts, which takes syntactic/semantic structural co-occurrences around target words into account. As noted in the previous section, we first build parse graphs for sentences containing target words, using a syntactic or semantic parser. In a parse graph, nodes represent words in the sentence, and edges represent (binary) relations between the words. We then compute the similarity between a pair of target words, represented as nodes in two different parse graphs, using a kernel that defines an inner product between nodes in graphs.

A popular approach to measuring the similarity of structured data is to count shared substructures with the so-called *convolution kernels* (Haussler, 1999; Gärtner, 2003). Our method also takes this approach; to measure the similarity between words in parse graphs, we use a variation of the kernel proposed by Desrosiers and Karypis (2009). This kernel is designed to evaluate the similarity of target nodes in a graph (or possibly in two different graphs) in which nodes and edges are labeled. The basic idea is to compute the number (weight) of identical walks starting from the nodes in respective graphs. Here, two walks are deemed identical if they have the same length and the series of labels along the nodes and edges in the two walks exactly match. We can also relax the requirement of exact matching, and instead allow for varying degree of similarity between different node/edge labels. The resulting kernel computes the node similarity through the degree of similarity of every pairs of (possibly non-identical) walks.

Below, we first give a definition of kernel (or similarity) between two individual walks in the parse graphs (Section 4.1), and then explain how to compute the kernel between two contexts viewed as bags of walks (*bag-of-walks kernel*) in Section 4.2. Under the random walk model presented in Section 3.2, this formulation allows efficient recursive computation without explicitly enumerating all single walks in the graphs.

Let G be a directed graph consisting of a union of two disjoint subgraphs, each of which corresponds to the parse graph for sentences S and S', respectively.

4.1 Walk similarity

We assume two kernel (similarity) functions $K_{\mathrm{word}}(w, w')$ and $K_{\mathrm{rel}}(r, r')$ are at our disposal, which are defined respectively over pairs of nodes (w, w') and pairs of edges (r, r') in G. In parse graphs, nodes represent words, so a natural example for $K_{\mathrm{word}}(w, w')$ is the cosine of

$\mathbf{v}_w$ and $\mathbf{v}_{w'}$, where $\mathbf{v}_w$ is the feature vector (e.g., type vector of Section 2) of word w; for edges, we can simply define $K_{\text{rel}}(r, r') = 1$ if the directed edges r and r' have the same relation label, or 0 otherwise. Using these basic kernels, we define kernel $K_{\text{walk}}(z, z')$ between two walks $z = (w_0, r_1, w_1, r_2, \ldots, r_t, w_t)$ and $z' = (w'_0, r'_1, w'_1, r'_2, \ldots, r'_{t'}, w'_{t'})$. First, if z and z' have different length $t \neq t'$, let $K_{\text{walk}}(z, z') = 0$. For two walks with the same length $t = t'$, we define

$$\begin{aligned} K_{\text{walk}}(z, z') &= K_{\text{word}}(w_0, w'_0) \cdot K_{\text{rel}}(r_1, r'_1) \cdot K_{\text{word}}(w_1, w'_1) \cdot \ \ldots \ \cdot K_{\text{rel}}(r_t, r'_t) \cdot K_{\text{word}}(w_t, w'_t) \\ &= \prod_{i=0}^{t} K_{\text{word}}(w_i, w'_i) \prod_{i=1}^{t} K_{\text{rel}}(r_i, r'_i). \end{aligned} \tag{2}$$

Thus, $K_{\text{walk}}(z, z')$ is the product of the similarity of the pairs of nodes and edges visited along the two walks z and z'.

4.2 Bag-of-walks kernels

Let w and w' be nodes in a graph[1] G. The *bag-of-walks kernels* between w and w', denoted by $K(w, w')$, is defined as a weighted sum of similarity given by K_{walk} between all pairs of walks of length at most L, with each walk starting from w and w', respectively. Here, L is the parameter of the kernel. Formally speaking,

$$K(w, w') = \sum_{\substack{z \in Z(w;L) \\ z' \in Z(w',L)}} p(z \mid w) p(z' \mid w') K_{\text{walk}}(z, z') \tag{3}$$

where $Z(w; L)$ denote the set of all walks of length at most L, starting from w in G. $p(z \mid w)$ is a function that assigns a non-negative weight to walk z. In the rest of the paper, we define $p(z \mid w)$ to be the one given by Eq. (1), i.e., the probability of taking walk z from w according to the random walk model defined in Section 3.2.

4.3 Efficient computation

Naively enumerating all walk pairs (z, z') in Eq. (3) is intractable when L is large, because the number of possible walks grows exponentially with length bound L. However, for the specific weight function $p(z \mid w)$ given by Eq. (1), we can efficiently compute $K(w, w')$ in a recursive manner, as follows.

For $t = 0, 1, \ldots, L$, and any node pair w and w' in G, let $K^{(t)}(w, w')$ be the quantity defined as

$$K^{(t)}(w, w') = \sum_{\substack{z \in Z(w;t) \\ z' \in Z(w';t)}} p(z \mid w) p(z' \mid w') K_{\text{walk}}(z, z'). \tag{4}$$

Eq. (4) has the same form as Eq. (3), except that it is parametrized by the maximum length t of allowed walks. Especially, $K(w, w') = K^{(L)}(w, w')$.

We now show how to compute $K^{(t)}$ recursively over $t = 0, \ldots, L$. For $t = 0$, there exists only one walk pair of length 0, namely, $z^{(0)} = (w)$, $z'^{(0)} = (w')$. Thus we have

$$K^{(0)}(w, w') = K_{\text{walk}}(z^{(0)}, z'^{(0)}) = K_{\text{word}}(w, w'), \tag{5}$$

[1] Note that even if two nodes w and w' appear in different graphs, they can be regarded as nodes in a single graph, namely, the graph union of the two graphs. Hence two nodes can be assumed to belong to a graph G without loss of generality.

where the second equality follows from Eq. (2).

For $t \geq 1$, observe that every walk starting from w will either (i) be terminated immediately with probability γ, or (ii) with probability $(1-\gamma)/|\mathrm{Out}(w)|$, follows an edge $r \in \mathrm{Out}(w)$ and connects to a walk of length at most $(t-1)$ starting from $\mathrm{sink}(r)$, yielding a walk of length at most t as a whole. Recall that $|\mathrm{Out}(w)|$ is the out-degree of nodes w. It follows that

$$K^{(t)}(w,w') = \gamma^2 K_{\mathrm{word}}(w,w') + \frac{(1-\gamma)^2}{|\mathrm{Out}(w)||\mathrm{Out}(w')|} \cdot K_{\mathrm{word}}(w,w') \sum_{\substack{r \in \mathrm{Out}(w) \\ r' \in \mathrm{Out}(w')}} K_{\mathrm{rel}}(r,r') \cdot K^{(t-1)}(\mathrm{sink}(r), \mathrm{sink}(r')). \quad (6)$$

By iteratively computing $K^{(t)}(v,v')$ for all pairs of nodes (v,v') in graph G over $t = 0, 1, \ldots, L$, we eventually obtain the desired quantity $K(w,w') = K^{(L)}(w,w')$, namely, the local structural similarity of nodes w and w' in graph G. The time complexity of this calculation is $O(|G|^2 L)$, where $|G|$ is the number of nodes in graph G. If G comprises two disjoint subgraphs corresponding to the parse graphs for sentence S and S', we can show that the computation time can be reduced to $O(|S||S'|L)$, where $|S|$ and $|S'|$ are the number of words in sentences S and S', respectively; or equivalently, these are the number of nodes in their respective parse graphs. It follows that in practice computation of this bag-of-walks kernel is feasible, because sentences S and S' usually contain only a moderate number of words.

Depending on the structure of the parse graphs and the value of L, we can further save computation. To obtain the value of $K(w_{\mathrm{target}}, w'_{\mathrm{target}}) = K^{(L)}(w_{\mathrm{target}}, w'_{\mathrm{target}})$, we need to compute Eqs. (5) and (6) not for all pairs of words in the sentences, but only for pairs (w,w') such that w and w' are within L steps from the target words w_{target} and w'_{target} in the parse graph.

4.4 Relation to other kernels for structured data

4.4.1 Desrosiers-Karypis kernels

The bag-of-walks kernels are similar to but not identical to the kernels proposed by Desrosiers and Karypis (2009), which also measures the similarity of nodes in the labeled graph using their surrounding structure.

The Desrosiers-Karypis kernels can also be viewed as a special case of Eq (3), but the underlying random walk model $p(\cdot)$ is different. Desrosiers and Karypis use a random walk model that do not cast a bound on the length of walks. In contrast, our model poses a strict upper bound L on the walk length; this model was chosen because co-occurrences comprising hundreds of words are unlikely to be effective for contextual word discrimination. Also, this upper bound reduces computational complexity.

Different random walk models lead to different recursive computations. Observe that the formula for Desrosiers-Karypis kernels (see the unnumbered equation for $\sigma_{u,u'}$ in (Desrosiers and Karypis, 2009, Section 3.3, page 266)) is given as a sum of the probability of generating parallel walks, and every term in the sum is multiplied by a square of stopping probability γ. This is not the case with the bag-of-walks kernels; notice that stopping probability γ is absent from Eq. (5), the base formula for recursive computation.

Another minor difference is that Desrosiers-Karypis kernels do not count the similarity of walks

with zero length. Our kernels take this into consideration, as reflected in Eq. (5) and the first term of Eq. (6).

Both the bag-of-walks kernels and the Desrosiers-Karypis kernels borrow idea from Kashima et al. (2003) who defined a kernel (*marginalized graph kernel*) between two graphs (not graph nodes) on the basis of the marginal probability of parallel random walks taking place in the two graphs. Elegant product graph formulations of various graph kernels can be found in Vishwanathan et al. (2010).

4.4.2 Tree kernels

Tree kernels (Collins and Duffy, 2001; Kashima and Koyanagi, 2002; Moschitti et al., 2008; Croce et al., 2011) efficiently count all common subtrees in two tree-structured data. Although it is tempting to apply these kernels to measure the similarity between the dependency (sub)trees surrounding the target words, it should be noted that our goal is to measure the similarity of particular word pairs in two sentences, and not of the sentences or their entire dependency trees. In particular, tree kernels do not give any special treatment of the target words, and thus they even count subtrees that do not involve a target word at all. Tree kernels are hence not suitable for measuring word similarity. In contrast, the bag-of-walks kernels distinguish target words from the other words in the sentence, as they only count walks starting from the target words. Moreover, tree kernels cannot deal with parser outputs containing semantic relations, as these outputs are in general graphs and not trees.

5 Experiments

In this section, we evaluate the bag-of-walks kernels in two tasks: word sense disambiguation (WSD) and paraphrase ranking. For the WSD task, we use the kernels to construct SVM classifiers. For the paraphrase ranking task in which no training data is provided, we use the kernel value simply as the similarity score used to rank candidate words.

We need to define two basic kernels, K_{word} between words (nodes) and K_{rel} between relations (edges) as the building blocks of the bag-of-walks kernels K (Eq. (3)). Throughout the experiments, we let K_{word} be the cosine between context-independent type vectors for words (see below). For K_{rel}, we use the identity function of edge labels; i.e., $K_{\text{rel}}(r, r')$ is 1 if the labels of edges r and r' are identical, or else it is 0.

Before proceeding to the experimental procedures and results, we describe how we built the type vectors for words. These type vectors are used for both the WSD and paraphrase ranking experiments.

Construction of context-independent type vectors for words. We first parse the written text part of British National Corpus (BNC) with syntactic-semantic parser Enju[2]. The outputs of Enju are then converted to parse graphs. The edges of the resulting parse graph have one of the following labels: "forward dependency", "backward dependency", "semantic subject (forward semantic subject)", and "semantic predicate (backward semantic subject)." We next collect pairs of (stemmed and lemmatized) words such that (i) each word occurs at least 10 times in BNC, and (ii) the pair is linked by a syntactic/semantic relation at least once in the parse graph collection we converted from BNC, but excluding the pairs for which the *pointwise mutual information* (pmi) (Church and Hanks, 1990) between the words are less than 1. Finally, for

[2] `http://www-tsujii.is.s.u-tokyo.ac.jp/enju/index.html`

every word w occurring in this collection, we make a type vector in which each component corresponds to a pair (w', r) of another word w' and the relation type r between w and w', and holds the pmi score of word w and the pair (w', r),

In preliminary experiments, we also tested type vectors made from the words in a fixed contextual window size (i.e., without using parser outputs), but the results were inferior.

5.1 Experiment 1: word sense disambiguation

For the WSD experiment, we apply the bag-of-walks kernels to the Senseval-3 English lexical sample (ELS) task (Mihalcea et al., 2004).

5.1.1 Task and dataset

The Senseval-3 ELS dataset is a collection of instances of polysemous target words and the contextual paragraphs in which they appear, consisting mostly of several sentences. Each target instance is annotated with one or more gold standard senses selected from a sense inventory (also distributed with the dataset). The dataset comes with predefined training/test splits, and the task goal is to predict a sense for each of the 3944 test instances of 57 polysemous words: 1807 instances for 20 nouns, 1978 for 32 verbs, and 159 for 5 adjectives.

A standard approach for this task is to represent contextual paragraphs as bag-of-words local context vectors. It then predicts the sense of the target word according to rules constructed from training data, usually with a machine learning technique. Recently, to further improve the WSD performance, Giuliano et al. (2009) built a kernel that measures similarity of n-gram collocations containing target words, and combined it with the cosine of local context vectors[3] In this WSD experiment, following Giuliano et al. (2009), we also combine the bag-of-walks kernels with the cosine of local context vectors, and evaluate the performance.

5.1.2 Experimental procedure

WSD with the bag-of-walks kernels consists of four steps: (1) computing the cosine of local context vectors computed from the contextual paragraphs, (2) computing bag-of-walks kernels on parse graphs output by a parser, (3) combining kernels computed in Steps 1 and 2, and (4) sense prediction by support vector machines (SVMs) using the combined kernels. We detail each step below.

Step 1. Compute the cosine of local context vectors. We construct a local context vector, for each target instance according to a standard procedure for WSD (Mihalcea, 2004; Navigli, 2009). Specifically, we treat a contextual paragraph simply as a bag-of-words. After removing stop words[4], we make a local context vector in which components are the weighted frequencies (tf-idf) of words in the "bag." Then we construct a matrix holding cosine between every pairs of local context vectors. Because cosine matrices are positive semi-definite, this matrix can be regarded as a kernel matrix.

Step 2. Compute bag-of-walks kernels. We compute a bag-of-walks kernel between instances of a target word as follows. For each target instance, we pick up a sentence containing the target word from the paragraph given as a context, and then construct a parse graph by

[3] The cosine of local context vectors is called *bag-of-words kernel* in Giuliano et al. (2009).

[4] We use a list of English stop words available from the on-line appendix to (Lewis et al., 2004).

parsing the sentence with the Enju parser. Then we compute the bag-of-walks kernel K using the recursive formula (6) in Section 4.3, for all pairs of target instances. As described in the beginning of this section, K_{word} in Eq. (2) is the cosine similarity between context-independent type vectors, and $K_{\text{rel}}(r, r')$ is the identify kernel of edge labels. Finally, we normalize the obtained bag-of-walks kernel matrix K, so that the (i, j) component of the resulting kernel matrix becomes $K(i, j)/\sqrt{K(i, i)K(j, j)}$.

Step 3. Combine two kernels. In a way similar to Giuliano et al. (2009), we compute a composite kernel by simply adding together the two kernel matrices obtained in Steps 1 and 2.

Step 4. Train SVMs and predict senses of the test data We train multi-class SVMs[5] with the Senseval-3 ELS training data, using the composite kernels obtained in Step 3. After tuning the parameters for each of the polysemous target words by five-fold cross validation on the training set, we train multi-class SVMs with entire training set. The tuned parameters are the walk length bound L and the stopping probability γ of the bag-of-walks kernels, and SVM's soft-margin parameter C. Finally, the trained SVMs are used to predict the sense of the test instances.

5.1.3 Compared methods

We use the most frequent sense prediction as the first baseline. It totally ignores the context, and always predicts the most frequent sense in the training data. This baseline is helpful to evaluate the difficulty of the WSD tasks.

As the second baseline, we use the cosine of local context vectors. Note again that we are interested in how much the performance improves when the bag-of-walks kernels are combined with this simple similarity measure.

We also compare our method with the one proposed by Giuliano et al. (2009), which is also kernel-based. Like bag-of-walks kernels, their kernel takes multi-word co-occurrences with a target word into account, but these co-occurrences are taken in terms of (gap weighted) n-gram collocations; i.e., their kernel counts the overlap of n-gram sequences surrounding the two target words. In contrast, bag-of-walks kernels compute multi-word co-occurrence in the syntactic/semantic structure present in parse graphs. Hence, the comparison of bag-of-walks kernels and Giuliano et al.'s kernel allows us to assess the effectiveness of using syntactic/semantic structure to measure word co-occurrence, rather than using n-grams.[6]

5.1.4 Evaluation

We evaluate predicted senses for test data by F1 score, computed with the scoring script distributed with the Senseval-3 ELS dataset.

5.1.5 Results

Table 1 shows the performance (F1 scores) of the compared methods. The F1 score of Giuliano et al. (2009) is taken from their paper. In total ("ALL"), the proposed method outperforms the

[5] We use the $SVM^{multiclass}$ package: `http://svmlight.joachims.org/svm_multiclass.html`

[6] Giuliano et al. (2009) combined cosine of local context vectors with not only the n-gram collocation kernels, but also "domain kernels." The domain kernels exploit external lexical knowledge sources and singular value decomposition to infer the domain of words. Since our focus is on the effective way of using word co-occurrence in local context, we compare our proposed method only with n-gram collocation kernels (combined with baseline bag-of-words kernels).

Method	ALL	Noun	Verb	Adjective
most frequent sense	55.2	54.2	56.5	49.7
cosine of local context vectors	63.6	64.1	60.3	46.5
proposed method	**71.0**	**72.1**	**71.7**	**50.3**
Giuliano et al. (2009)	69.7	-	-	-

Table 1: F1 scores on the WSD experiment.

baselines and Giuliano et al. (2009)'s method.

These results indicate the effectiveness of using syntactic/semantic structural co-occurrences of multiple words for the WSD tasks.

Table 1 also shows the F1 scores when the test data was broken down by part-of-speech (noun, verb, and adjective). In all parts-of-speech, the proposed method outperformed the baseline. Giuliano et al. (2009) do not report break-down results of their method on individual parts-of-speech, so the corresponding columns are left blank.

The stopping probability γ, and walk length bound L of the bag-of-walks kernels, obtained by cross-validation on the training data, were $\gamma = 0.08$ and $L = 3.0$ on averaged over all the 57 target words. The average value of L for each part-of-speech was $L = 3.6$ for noun, $L = 2.5$ for verb, and $L = 4.2$ for adjective. Thus in all cases, we have $L > 1$, so we conclude that it is worth considering structural co-occurrence of multiple words, not just single words that are directly connected to target words (which corresponds to $L = 1$).

5.2 Experiment 2: ranking paraphrases

In the second experiment, we evaluate bag-of-walks kernels on the task of ranking paraphrases.

5.2.1 Task and dataset

In the paraphrase ranking task, the system is given a target word and the list of its paraphrase candidates. Also given is an instance of the target word with the (sentential) context it appears, and the task goal is to rank the paraphrase candidates by their appropriateness in the light of the given context. No training data is provided, so the system is allowed to use external resources.

Following (Erk and Padó, 2008; Thater et al., 2010; Erk and Padó, 2010; Dinu and Lapata, 2010), we adapt the SemEval-2007 English lexical substitution (ELS) dataset (McCarthy and Navigli, 2007) for a paraphrase ranking task. The dataset consists of 205 target words (59 nouns, 54 verbs, 57 adjectives, and 35 adverbs), each with at most 10 instances of sentential contexts. For each target instance, gold standard paraphrases are annotated and ranked by annotators' votes. Notice that although we use this dataset to organize a paraphrase ranking task, the task setting is different from the original SemEval-2007 ELS task. In SemEval-2007, paraphrase candidates are not provided, and hence the system is required to generate them before ranking. Here in the paraphrase ranking task, we focus solely on ranking candidate paraphrases, which are predefined and given. To make this predefined list of candidate paraphrases from the SemEval-2007 ELS dataset, we pool all the gold standard paraphrases in the dataset for each target word. Now the goal is to rank these candidates for each target instance occurring in a specific context.

Notice that because we use BNC to collect contexts for candidate paraphrases (see Section 5.2.2), paraphrases occurring less than 100 times in the corpus are removed from the candidate lists; a similar filtering was done in (Thater et al., 2010). After this filtering, the average number of candidates for a target word becomes 14.0 in total (13.5 for nouns, 17.6 for verbs, 14.4 for adjectives, and 8.6 for adverbs).

5.2.2 Experimental procedure

Step 1. Collect candidate instances. Given an instance of a target word (i.e., target word in a specific sentential context), we rank its paraphrase candidates according to their contextual similarity to the target instance. To this end, we need the contexts in which candidate words appear, but candidates are provided without context. Hence, for each candidate word, we collect 100 sentential contexts from BNC.

Step 2. Compute bag-of-walks kernels. After collecting sentential contexts for candidate words in Step 1, we convert these sentences into the corresponding parse graphs using the Enju parser and the post-processing described in Section 3.1. The contextual sentences for target instances, which come from SemEval-2007 data, are also converted to parse graphs in the same way. We then apply the bag-of-walks kernels on these graphs to compute the kernel value $K(w, w')$ between a target instance w and each candidate instance w'. The similarity score of w' relative to w is then given by $K(w, w')/\sqrt{K(w, w), K(w', w')}$. In the bag-of-walks kernel computation, we used the same basic kernels K_{word} and K_{rel} as those used for the WSD experiment.

Step 3. Rank candidates. For a given target instance, we compute the score of each candidate word by averaging the similarity scores between the target instance and (a hundred) candidate instances, which we calculated in Step 2. The candidates are then ranked by their scores.

5.2.3 Evaluation

Following previous work (Erk and Padó, 2008; Thater et al., 2010; Erk and Padó, 2010; Dinu and Lapata, 2010), we evaluate the rankings output by each system by *generalized average precision* (GAP) (Kishida, 2005).

5.2.4 Compared methods

The baseline in this task is to rank candidates according to cosine similarity with a target instance using type vectors. Note that since type vectors are context-independent, rankings obtained from this baseline become identical for all instances of a target word regardless of the difference in context in which they appear.

We also compare the performance of bag-of-walks kernels against those reported in two previous studies (Erk and Padó, 2010; Dinu and Lapata, 2010). These studies conducted paraphrase ranking for all target words in the SemEval-2007 dataset, just like this experiment. However, notice that these results are just for reference, because detailed experimental settings, such as the way they built gold candidates for each target word, are probably not the same with ours.

5.2.5 Results

Table 2 shows the performance (GAP score) of the compared methods. For bag-of-walks kernels, we did not tune the parameters since no training data was provided. So the scores displayed

Method	ALL	Noun	Verb	Adjective	Adverb
cosine similarity	44.6	44.0	36.3	45.7	57.0
bag-of-walks kernel ($L = 1$)	46.7	45.2	39.1	47.5	60.1
bag-of-walks kernel ($L = 2$)	47.2	45.5	39.4	**48.7**	60.0
bag-of-walks kernel ($L = 3$)	**47.5**	45.2	40.6	48.6	**60.7**
bag-of-walks kernel ($L = 4$)	47.3	45.6	40.2	**48.7**	59.5
bag-of-walks kernel ($L = 5$)	47.4	45.3	40.8	48.6	59.6
bag-of-walks kernel ($L = 6$)	47.3	45.5	**40.9**	48.1	59.2
bag-of-walks kernel ($L = 7$)	47.0	45.3	40.7	47.6	59.0
bag-of-walks kernel ($L = 8$)	46.8	**45.9**	40.2	47.4	57.7
bag-of-walks kernel ($L = 9$)	46.6	**45.9**	39.9	46.9	57.9
(Erk and Padó, 2010)*	38.6	41.4	38.4	37.5	-
(Dinu and Lapata, 2010)*	42.9	-	-	-	-

Table 2: GAP scores on the paraphrase ranking experiment. Note that the results for the previous studies (*) are just for reference, as the detailed task setting is different from this experiment. Scores for the proposed methods are obtained with $\gamma = 0$.

are those when the walk termination probability γ is set to zero, because it gave the best results. According to Table 2, all the bag-of-walks kernels with maximum walk length L from 1 to 9 outperform the baseline cosine similarity in total ("ALL") as well as when the data is split by part-of-speech (noun, verb, adjective, and adverb). Also, better scores are obtained when $L \geq 3$, indicating that it is effective to take the structural co-occurrences consisting of more than two words into consideration.

Conclusion

We have proposed a new measure of semantic similarity between words in context. It captures structural collocation between the target words and multiple words in the context simultaneously. The proposed measure applies the bag-of-walks kernels, a variation of Desrosiers and Karypis' kernel between graph nodes, to the syntactic/semantic graph output by semantic parsers.

In the experiments on word sense disambiguation and paraphrase ranking, we verified that higher-order relations between words are effective; setting the maximum walk length of $L = 3$ steps achieved the highest performance in these tasks. And in the paraphrase ranking task, if the target words are limited to nouns only, walk length as long as $L = 9$ achieved the highest score.

While we tested our method only on English data, it does not rely on any language-specific features, and hence should work on any languages with sufficiently accurate dependency parsers. Such highly-accurate parsers are now available for many languages, as evidenced by the success of the CoNLL-X shared task (Buchholz and Marsi, 2006).

References

Berger, A. and Lafferty, J. (1999). Information retrieval as statistical translation. In *Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '99)*, pages 222–229, Berkeley, California, USA.

Buchholz, S. and Marsi, E. (2006). CoNLL-X shared task on multilingual dependency parsing. In *Proceedings of the 10th Conference on Natural Language Learning (CoNLL-X)*, pages 149–164, New York, NY, USA.

Church, K. W. and Hanks, P. (1990). Word association norms, mutual information, and lexicography. *Computational Linguistics*, 16(1):22–29.

Collins, M. and Duffy, N. (2001). Convolution kernels for natural language. In *Advances in Neural Information Processing Systems 14 (NIPS '01)*, pages 625–632. MIT Press.

Croce, D., Moschitti, A., and Basili, R. (2011). Structured lexical similarity via convolution kernels on dependency trees. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing (EMNLP '11)*, pages 1034–1046, Edinburgh, UK.

Deerwester, S., Dumais, S. T., Furnas, G. W., Landauer, T. K., and Harshman, R. (1990). Indexing by latent semantic analysis. *Journal of the Society for Information Science*, 41:391–407.

Desrosiers, C. and Karypis, G. (2009). Within-network classification using local structure similarity. In *Proceedings of the European Conference on Machine Learning and Knowledge Discovery in Databases (ECML-PKDD '09): Part I*, Lecture Notes in Artificial Intelligence 5781, pages 260–275, Bled, Slovenia. Springer-Verlag.

Dinu, G. and Lapata, M. (2010). Measuring distributional similarity in context. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing (EMNLP '10)*, pages 1162–1172, Cambridge, Massachusettes, USA.

Erk, K. and Padó, S. (2008). A structured vector space model for word meaning in context. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing (EMNLP '08)*, pages 897–906, Honolulu, Hawaii, USA.

Erk, K. and Padó, S. (2010). Exemplar-based models for word meaning in context. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics (ACL '10), Short Papers*, pages 92–97, Uppsala, Sweden.

Gärtner, T. (2003). A survey of kernels for structured data. *SIGKDD Explorations*, 5:49–58.

Giuliano, C., Gliozzo, A. M., and Strapparava, C. (2009). Kernel methods for minimally supervised WSD. *Computational Linguistics*, 35(4):513–528.

Haussler, D. (1999). Convolution kernels on discrete structures. Technical Report UCS-CRL-99-10, University of California at Santa Cruz, Santa Cruz, CA, USA.

Jurafsky, D. and Martin, J. H. (2008). *Speech and Language Processing*. Pearson Prentice Hall, 2nd edition.

Kashima, H. and Koyanagi, T. (2002). Kernels for semi-structured data. In *Proceedings of the 19th International Conference on Machine Learning (ICML '02)*, pages 291–298, Sydney, Australia.

Kashima, H., Tsuda, K., and Inokuchi, A. (2003). Marginalized kernels between labeled graphs. In *Proceedings of the 20th International Conference on Machine Learning (ICML '03)*, pages 321–328, Washington, DC, USA. AAAI Press.

Kishida, K. (2005). Property of average precision and its generalization: an examination of evaluation indicator for information retrieval experiments. Technical Report NII-2005-014E, National Institute of Informatics, Tokyo, Japan.

Lewis, D. D., Yang, Y., Rose, T. G., and Li, F. (2004). RCV1: a new benchmark collection for text categorization research. *Journal of Machine Learning Research*, 5:361–397.

McCarthy, D. and Navigli, R. (2007). Semeval-2007 task 10: English lexical substitution task. In *Proceedings of the 4th International Workshop on Semantic Evaluations (SemEval-2007)*, pages 48–53, Prague, Czech Republic.

Mihalcea, R. (2004). Co-training and self-training for word sense disambiguation. In Ng, H. T. and Riloff, E., editors, *Proceedings of the 8th Conference on Computational Natural Language Learning (CoNLL '04)*, pages 33–40, Boston, Massachusetts, USA.

Mihalcea, R., Chklovski, T., and Kilgarriff, A. (2004). The Senseval-3 English lexical sample task. In Mihalcea, R. and Edmonds, P., editors, *Proceedings of the Third International Workshop on the Evaluation of Systems for the Semantic Analysis of Text (Senseval-3)*, pages 25–28, Barcelona, Spain.

Mitchell, J. and Lapata, M. (2008). Vector-based models of semantic composition. In *Proceedings of the 46th Annual Meeting of the Association of Computational Linguistics: Human Language Technologies (ACL '08)*, pages 236–244, Columbus, Ohio, USA.

Moschitti, A., Pighin, D., and Basili, R. (2008). Tree kernels for semantic role labeling. *Computational Linguistics*, 34(2):193–224.

Navigli, R. (2009). Word sense disambiguation: A survey. *ACM Computing Surveys*, 41:10:1–10:69.

Reisinger, J. and Mooney, R. J. (2010). Multi-prototype vector-space models of word meaning. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL-HLT '10)*, pages 109–117, Los Angeles, California, USA.

Schütze, H. (1998). Automatic word sense discrimination. *Computational Linguistics*, 24:97–123.

Thater, S., Fürstenau, H., and Pinkal, M. (2010). Contextualizing semantic representations using syntactically enriched vector models. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics (ACL '10)*, pages 948–957, Uppsala, Sweden.

Vishwanathan, S. V. N., Schraudolph, N. N., Kondor, R., and Borgwardt, K. M. (2010). Graph kernels. *Journal of Machine Learning Research*, 11:1201–1242.

Flexible Japanese Sentence Compression by Relaxing Unit Constraints

Jun Harashima Sadao Kurohashi
Graduate School of Informatics, Kyoto University
Yoshida-honmachi, Sakyo-ku, Kyoto, 606-8501, Japan
harashima@nlp.ist.i.kyoto-u.ac.jp, kuro@i.kyoto-u.ac.jp

ABSTRACT
Sentence compression is important in a wide range of applications in natural language processing. Previous approaches of Japanese sentence compression can be divided into two groups. *Word-based methods* extract a subset of words from a sentence to shorten it, while *bunsetsu-based methods* extract a subset of bunsetsu (where a bunsetsu is a text unit that consists of content words and following function words). Basically, bunsetsu-based methods perform better than word-based methods. However, bunsetsu-based methods have the disadvantage that they cannot drop unimportant words from each bunsetsu because they have to follow constraints under which each bunsetsu is treated as a unit. In this paper, we propose a novel compression method to overcome this disadvantage. Our method relaxes the constraints using Lagrangian relaxation and shortens each bunsetsu if it contains unimportant words. Experimental results show that our method effectively compresses a sentence while preserving its important information and grammaticality.

TITLE AND ABSTRACT IN JAPANESE

ユニット制約の緩和による柔軟な日本語文圧縮

文圧縮は，自然言語処理の様々なアプリケーションにおいて重要である．日本語文に対する既存の圧縮手法は二種類に分けられる．単語ベースの手法は文から単語集合を選出し，圧縮文とする．一方，文節ベースの手法は文から文節集合を選出し，圧縮文とする．基本的には後者の方が良く機能する．しかし，文節ベースの手法は，文節をユニットとして扱うという制約があるため，個々の文節を圧縮できない．本稿では，この欠点を克服する新しい圧縮手法を提案する．提案手法はラグランジュ緩和を用いて上の制約を緩和し，各文節を圧縮する．実験の結果，提案手法によって原文の情報を多く保持する文法的な圧縮文を生成できることが分かった．

KEYWORDS: sentence compression, Lagrangian relaxation.

KEYWORDS IN JAPANESE: 文圧縮, ラグランジュ緩和.

Proceedings of COLING 2012: Technical Papers, pages 1097–1112,
COLING 2012, Mumbai, December 2012.

1 Introduction

Sentence compression is the task of shortening a sentence while preserving its important information and grammaticality. This task is important in a wide range of applications such as automatic summarization (Jing, 2000; Lin, 2003; Zajic et al., 2007), subtitle generation (Vandeghinste and Pan, 2004), and displaying text on small screens (Corston-Oliver, 2001).

In this paper, we propose a novel compression method for a Japanese sentence. Like other languages, Japanese uses sentences composed of words. However, we can also say that a Japanese sentence is composed of *bunsetsu*. Bunsetsu is a text unit that consists of one or more content words and possibly one or more function words. For example, consider the following sentence.[1]

(1) *nihon to kanada no kokusai kyoudou kenkyuu guru-pu ga hakken shita*
Japan CNJ Canada GEN international collaborative research group NOM discover did

(An international collaborative research group between Japan and Canada made a discovery)

This sentence is composed of four bunsetsu: "*nihon to*", "*kanada no*", "*kokusai kyoudou kenkyuu guru-pu ga*", and "*hakken shita*". As seen in this example, a Japanese sentence can be viewed as a bunsetsu sequence as well as a word sequence.

This characteristic of the Japanese language has led researchers to take two compression approaches: *word-based methods* and *bunsetsu-based methods*. Word-based methods view a source sentence as a word sequence and generate a compressed sentence by selecting a subset of words from the source sentence (Hori and Furui, 2004; Hirao et al., 2009). However, the methods do not take account of bunsetsu, and it is thus difficult to generate grammatical compressions. For example, if only content words (or only function words) in a bunsetsu are selected, the grammaticality of the corresponding part in the compressed sentence would be poor.

We can avoid this problem using bunsetsu-based methods. Bunsetsu-based methods view a source sentence as a bunsetsu sequence and generate a compressed sentence by selecting a subset of bunsetsu from the source sentence (Takeuchi and Matsumoto, 2001; Oguro et al., 2002; Yamagata et al., 2006; Nomoto, 2008). Bunsetsu-based methods treat each bunsetsu as a unit. Thus, the methods do not suffer from the above problem and they can generate compressions that are more grammatically correct than those generated by word-based methods.

However, bunsetsu-based methods have a disadvantage in that they cannot shorten each bunsetsu in a source sentence. More precisely, when there is a compound noun in a bunsetsu and the noun contains unimportant words, bunsetsu-based methods cannot drop those words from the noun. Consider the above sentence (1) as an example. The third bunsetsu "*kokusai kyoudou kenkyuu guru-pu ga*" contains a compound noun "*kokusai kyoudou kenkyuu guru-pu*" (international collaborative research group). Suppose that we want to drop the word "*kokusai*" (international) from the noun and to shorten the bunsetsu to "*kyoudou kenkyuu guru-pu ga*". However, bunsetsu-based methods cannot perform such flexible word selection because they have to treat each bunsetsu as it is.

[1] This paper uses the abbreviations NOM (nominative), ACC (accusative), DAT (dative), ALL (allative), GEN (genitive), CMI (comitative), CNJ (conjunction), and TOP (topic marker).

In this paper, we propose a novel compression method to overcome the above disadvantage. As described above, the disadvantage results from a constraint for each bunsetsu under which the bunsetsu has to be treated as a unit (hereafter called the *unit constraint*). If we ignore unit constraints, we may be able to avoid the problem. However, if we do so, we again suffer from the problem of word-based methods (i.e., we will not generate grammatical compressions). We therefore do not ignore or adhere to unit constraints, but relax them using Lagrangian relaxation. That is, the proposed method basically follows the constraints and treats each bunsetsu as a unit. However, if a bunsetsu contains unimportant words, our method violates its unit constraint and drops those words from the bunsetsu. In this paper, we formulate this idea using integer linear programming (ILP) and report the effectiveness through experiments.

2 Word-based Method

We first describe word-based methods in detail. Although several word-based methods have been proposed (Hori and Furui, 2004; Hirao et al., 2009), the basic idea behind the methods is the same. We explain the idea in detail and discuss the advantage and disadvantage of word-based methods.

2.1 Idea

In word-based methods, a source sentence is viewed as a word sequence. Let w_i $(i = 1, \ldots, I)$ denote a word in a source sentence. The basic idea underlying word-based methods is that the compression is a subset of words with the maximum importance in a source sentence. Through ILP, this idea is formulated as follows.

Sentence Compression (Word-based Formulation)

$$\text{maximize} \quad \sum_{i=1}^{I} x_i \, Score(w_i) \tag{1}$$

$$\text{subject to} \quad \sum_{i=1}^{I} x_i \, Length(w_i) \leq L \tag{2}$$

$$x_i = 0 \text{ or } 1 \quad (i = 1, \ldots, I) \tag{3}$$

where x_i denotes a decision variable of w_i that is 1 if w_i is contained in a compressed sentence, and otherwise 0. $Score(w_i)$ represents the importance of w_i and $Length(w_i)$ represents the length of w_i. L is a predefined maximum length of a compressed sentence. According to Eq. (1), the optimal subset of words in a source sentence is selected as a compressed sentence. In addition, according to Eq. (2), the length of the compressed sentence shall be not more than L.

2.2 Advantage and Disadvantage

The advantage of word-based methods is that the methods can more freely select important words in a source sentence than bunsetsu-based methods. This is because word-based methods do not take account of bunsetsu in a source sentence and are not limited to unit constraints.

However, due to the freeness, word-based methods have the disadvantage that they tend to generate ungrammatical compressions. As described in Section 1, if only content words (or

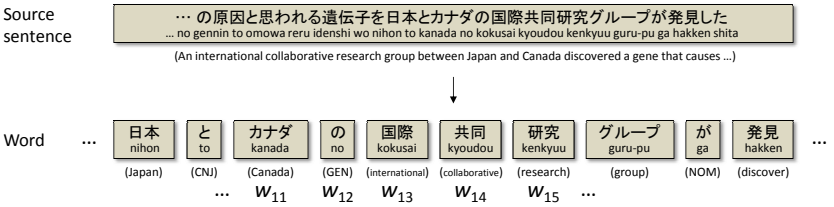


Figure 1: Word-based method. A source sentence is segmented into words and a subset of words in the sentence is selected as the compressed sentence.

only function words) in a bunsetsu are selected, the grammaticality of the corresponding part in the compressed sentence would be poor. Consider the sentence in Figure 1. There is, for example, the bunsetsu "*kanada no*" in the sentence (although word-based methods do not take account of it). If we select only the word "*kanada*" (Canada) from the bunsetsu and do not select the word "*no*" (GEN), the corresponding part of the compressed sentence would not make sense. As just described, it is difficult to generate grammatical compressions using word-based methods.

3 Bunsetsu-based Method

Like word-based methods, most previous bunsetsu-based methods (Oguro et al., 2002; Yamagata et al., 2006; Nomoto, 2008) are based on the same idea. In this section, we explain the idea and discuss the advantages and disadvantage of bunsetsu-based methods.

3.1 Idea

Word-based methods view a source sentence as a word sequence, while bunsetsu-based methods view the sentence as a bunsetsu sequence. Let b_j $(j = 1, \ldots, J)$ denote a bunsetsu in a source sentence. The basic idea underlying bunsetsu-based methods is that the compression is a subset of bunsetsu with the maximum importance in a source sentence. Through ILP, this idea is formulated as follows.

Sentence Compression (Bunsetsu-based Formulation)

$$\text{maximize} \quad \sum_{j=1}^{J} y_j \, Score(b_j) \tag{4}$$

$$\text{subject to} \quad \sum_{j=1}^{J} y_j \, Length(b_j) \leq L \tag{5}$$

$$y_j = 0 \text{ or } 1 \quad (j = 1, \ldots, J) \tag{6}$$

where y_j denotes a decision variable of b_j that is 1 if b_j is contained in a compressed sentence, and otherwise 0. $Score(b_j)$ represents the importance of b_j and $Length(b_j)$ represents the length of b_j. According to Eq. (4), the optimal subset of bunsetsu in a source sentence is selected as a compressed sentence. In addition, according to Eq. (5), the length of the compressed sentence shall be not more than L.

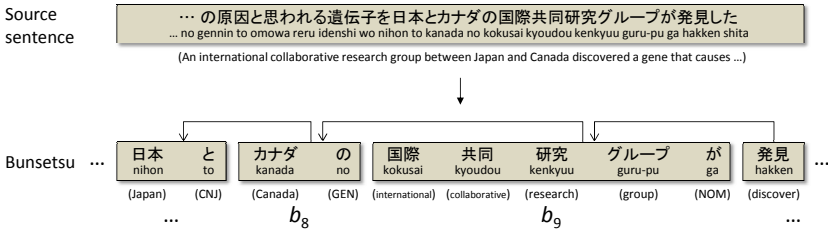


Figure 2: Bunsetsu-based method. A source sentence is segmented into bunsetsu and a subset of bunsetsu in the sentence is selected as the compressed sentence. Bunsetsu-based methods have the advantage in that they are able to use dependency information.

3.2 Advantages and Disadvantage

One advantage of bunsetsu-based methods is that the methods can generate compressions that are more grammatical than compressions generated by word-based methods. Bunsetsu-based methods select each bunsetsu in a source sentence just as it is. Therefore, the methods do not suffer from the problem of word-based methods (see also Section 2.2).

Bunsetsu-based methods have the another advantage in that they are able to use dependency information in a source sentence. In Japanese, a dependency relation is generally defined between not a pair of words but a pair of bunsetsu. Consider the source sentence in Figure 2. In the sentence, there is the example that bunsetsu b_8 depends on bunsetsu b_9. Bunsetsu-based methods can use this information by adding the following simple constraint to the above formulation.

$$\text{subject to} \quad y_8 \leq y_9 \tag{7}$$

This constraint ensures that if b_8 is contained in a compressed sentence, b_9 is also contained in the sentence. In this way, bunsetsu-based methods can easily use dependency information in a source sentence. On the other hand, there is a word-based method that defines dependency relations between words in a source sentence and uses the information (Hori and Furui, 2003). However, as described above, a dependency relation is generally defined between a pair of bunsetsu. In the method, bunsetsu dependencies in a source sentence and complex rules are necessary to define the word dependencies, and it is not easy to use the information.

On the other hand, as described in Section 1, the disadvantage of bunsetsu-based methods is that they cannot shorten each bunsetsu in a source sentence. More precisely, when there is a compound noun in a bunsetsu and the noun contains unimportant words, bunsetsu-based methods can not drop those words from the noun. Consider again the sentence in Figure 2. Bunsetsu b_9 contains a compound noun "*kokusai kyoudou kenkyuu guru-pu*" (international collaborative research group). Suppose that we want to drop the word "*kokusai*" (international) from the noun and to shorten the bunsetsu to "*kyoudou kenkyuu guru-pu ga*". However, bunsetsu-based methods cannot perform such flexible word selection because they are limited to unit constraints.

4 Proposed Method

Bunsetsu-based methods basically perform better than word-based methods, especially in terms of the grammaticality of a compressed sentence. However, bunsetsu-based methods have the disadvantage that they cannot shorten each bunsetsu in a source sentence. In this section, we describe a novel compression method that overcomes this disadvantage.

4.1 Idea

The point of our method is to relax unit constraints responsible for the disadvantage. Under the constraints, we have to treat each bunsetsu as a unit. If we ignore the constraints, we may be able to avoid the problem. However, if we do so, we again suffer from the problem of word-based methods (i.e., we will not generate grammatical compressions). Therefore, we select a set of bunsetsu, each containing unimportant words, and relax their unit constraints. Note that each bunsetsu that contains a compound noun (e.g., b_9 in Figure 3) is selected as a bunsetsu that may contain unimportant words. Conversely, we do not shorten each bunsetsu that does not contain a compound noun (e.g., b_8 in Figure 3) because such a bunsetsu has only one content word and does not need to be shortened.

First, let us rewrite the bunsetsu-based formulation in Section 3.1. Using w_i and x_i instead of b_j and y_j, the formulation can be rewritten as follows.

Sentence Compression (Bunsetsu-based Formulation 2)

$$\text{maximize} \quad \sum_{i=1}^{I} x_i \, Score(w_i) \tag{8}$$

$$\text{subject to} \quad \sum_{i=1}^{I} x_i \, Length(w_i) \leq L \tag{9}$$

$$x_i = 0 \text{ or } 1 \quad (i = 1, \ldots, I) \tag{10}$$

$$x_{First(b_j)} = x_{First(b_j)+1} = \cdots = x_{Last(b_j)} \quad (w_{First(b_j)}, \ldots, w_{Last(b_j)} \in b_j, \; j = 1, \ldots, J) \tag{11}$$

where $First(b_j)$ represents a function that returns the index of the first word in b_j, while $Last(b_j)$ returns that of the last word in b_j (e.g., in Figure 3, $First(b_9) = 13$ and $Last(b_9) = 17$). In addition, we set $Score(b_j) = \sum_{w_i \in b_j} Score(w_i)$ and $Length(b_j) = \sum_{w_i \in b_j} Length(w_i)$.

The notable aspect of the above formulation is Eq. (11), which is the set of unit constraints. Equation (11) ensures that if we select one word from a bunsetsu in a source sentence, we also select the other words from the bunsetsu. Likewise, if we do not select one word from a bunsetsu, we must also not select the other words from the bunsetsu.

The proposed method does not ignore or adhere to unit constraints but relaxes them. To do this, we use Lagrangian relaxation, which is a classical technique for combinatorial optimization. The technique moves problematic constraints into the objective function and penalizes the function if those constraints are not satisfied. Using the technique, we remove unit constraints for each bunsetsu that contains a compound noun.

Let B_{CN} denotes a subset of bunsetsu, each containing a compound noun. In addition, let us decompose a unit constraint for b_j in B_{CN} (i.e., $x_{First(b_j)} = x_{First(b_j)+1} = \cdots = x_{Last(b_j)}$) into

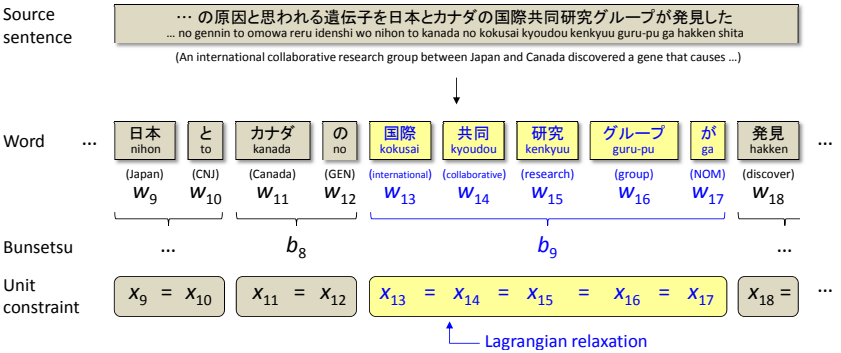


Figure 3: Proposed method. Bunsetsu b_9 contains a compound noun "*kokusai kyoudou kenkyuu guru-pu*" (international collaborative research group). The proposed method relaxes the unit constraint for the bunsetsu using Lagrangian relaxation.

a set of constraints: $x_{First(b_j)} = x_{Last(b_j)}$, $x_{First(b_j)+1} = x_{Last(b_j)}$, …, and $x_{Last(b_j)-1} = x_{Last(b_j)}$. Our new formulation is given below.

Sentence Compression (Proposed Formulation)

$$\text{maximize} \quad \sum_{i=1}^{I} x_i \, Score(w_i) + \sum_{b_j \in B_{CN}} \sum_{w_i \in b_j \setminus w_{Last(b_j)}} \mu_{i,Last(b_j)}(x_i - x_{Last(b_j)}) \tag{12}$$

$$\text{subject to} \quad \sum_{i=1}^{I} x_i \, Length(w_i) \leq L \tag{13}$$

$$x_i = 0 \text{ or } 1 \quad (i = 1, \ldots, I) \tag{14}$$

$$x_{First(b_j)} \leq x_{First(b_j)+1} \leq \cdots \leq x_{CLast(b_j)} = \cdots = x_{Last(b_j)}$$
$$(w_{First(b_j)}, \ldots, w_{Last(b_j)} \in b_j, \; j = 1, \ldots, J) \tag{15}$$

where $\mu_{i,Last(b_j)}$ is a Lagrangian multiplier provided for a constraint $x_i = x_{Last(b_j)}$. $CLast(b_j)$ in Eq. (15) returns the index of the last content word in b_j (e.g., in Figure 3, $CLast(b_9) = 16$).

A notable aspect is the second term in Eq. (12). For a bunsetsu b_j in B_{CN} (e.g., b_9 in Figure 3), the term penalizes the objective function if a decision variable of a word in b_j (denoted as x_i) is not equal to that of the last word in b_j (denoted as $x_{Last(b_j)}$). For example, if $x_i = 0$ and $x_{Last(b_j)} = 1$, the term penalizes the objective function. Thus, the proposed method basically treats words in b_j as a unit similarly to bunsetsu-based methods. However, now there is no constraint under which $x_{First(b_j)} = x_{First(b_j)+1} = \cdots = x_{Last(b_j)}$. In other words, although we have to consider the penalty, we can set a different value for each decision variable. Suppose that w_{13} in b_9 in Figure 3 have little importance, while the rest words in b_9 have great importance. Unlike bunsetsu-based methods, the proposed method can set $x_{13} = 0$ and the rest decision variables to 1.

Equation (15) represents a constraint that sets the order of preference of the selection of words in a bunsetsu. As described above, for each bunsetsu in B_{CN}, each decision variable

Algorithm 1 Solve the proposed formulation.

for $b_j \in B_{CN}$ **do**
  for $w_i \in b_j \backslash w_{Last(b_j)}$ **do**
    $\mu^{(0)}_{i,Last(b_j)} \leftarrow 0$
  end for
end for
for $t \in \{1, \ldots, T\}$ **do**
  $\boldsymbol{x}^{(t)} \leftarrow \arg\max_{\boldsymbol{x}}$ Eq. (12) (Note that $\mu^{(t-1)}_{i,Last(b_j)}$ is used as $\mu_{i,Last(b_j)}$ in Eq. (12)
  for $b_j \in B_{CN}$ **do**
    for $w_i \in b_j \backslash w_{Last(b_j)}$ **do**
      $\mu^{(t)}_{i,Last(b_j)} \leftarrow \mu^{(t-1)}_{i,Last(b_j)} - \alpha^{(t)}(x^{(t)}_i - x^{(t)}_{Last(b_j)})$
    end for
  end for
end for
return $\boldsymbol{x}^{(T)}$

can take a different value from other variables. However, care must be taken in setting each variable. More precisely, latter words in a bunsetsu generally should not be dropped before the earlier words are dropped. One reason for this is that function words are located in the latter part of a bunsetsu. Another is that former words within a Japanese compound noun basically modify the latter words (i.e., the latter words are syntactically more important than the former words). We thus add a constraint to our formulation under which we prioritize latter words in a bunsetsu.

We set $x_{CLast(b_j)} = \cdots = x_{Last(b_j)}$ in the latter part of Eq. (15). The purpose is to treat function words in a bunsetsu in B_{CN} as a unit and select at least one content word from the bunsetsu. Furthermore, using this equation, we can retain a unit constraint for each bunsetsu that does not contain a compound noun (i.e., for such bunsetsu, Eq. (15) is the same as Eq. (11)).

Of course, there are exceptions to Eq. (15), especially proper nouns. For example, if we drop the first word from the compound noun "*murayama tomiichi syusyou*", an unlikely noun "*tomiichi syusyou*" would be generated ("*murayama*", "*tomiichi*", and "*syusyou*" meaning Murayama, Tomiichi, and prime minister, respectively). In this case, we need to recognize the family name ("*murayama*"), the last name ("*tomiichi*"), and the title ("*syusyou*") and drop the words in the following order: last name, family name, and title. In our experiments described in the following section, we handle this exception for person names. However, we do not handle exceptions about other proper nouns such as organization names. We leave this for our future work.

Finally, we present an algorithm to solve our formulation in Algorithm 1. In the algorithm, T denotes the number of iterations and $\alpha^{(t)}$ denotes a parameter that determines a step size to update each Lagrangian multiplier (see (Korte and Vygen, 2008) for detail). Using this algorithm, each multiplier is updated and a subset of words in a source sentence is selected so that a compression produced by our method is as similar to that produced by bunsetsu-based methods as possible. However, as described in the previous paragraphs, if bunsetsu contain unimportant words, our method prioritizes to violate their unit constraints and drop the unimportant words.

4.2 Advantages

Compared with word-based and bunsetsu-based methods, the proposed method has at least three advantages. First, our method can generate compressions that are more grammatical than compressions generated with word-based methods. This is because our method is loosely based on bunsetsu-based methods and basically treats each bunsetsu in a source sentence as it is.

Second, unlike word-based methods, our method can easily use dependency information in a source sentence. This is again because our method is loosely based on bunsetsu-based methods. For example, bunsetsu b_8 in Figure 3 depends on bunsetsu b_9 (see also Figure 2). We can use this information employing the following constraint.

$$\text{subject to} \quad x_{12} \leq x_{17} \tag{16}$$

That is, we introduce a constraint between the last words of the bunsetsu. In this way, when b_8 is contained in a compressed sentence, we can ensure that b_9 is also contained in the sentence regardless of whether the bunsetsu are shortened.

Third, unlike bunsetsu-based methods, our method can shorten a bunsetsu in a source sentence. Since bunsetsu-based methods are limited by unit constraints, they have to treat each bunsetsu as a unit. Thus, even if there are unimportant words in a bunsetsu, the methods do not drop those words from the bunsetsu. On the other hand, our method relaxes unit constraints using Lagrangian relaxation. Thus, our method has the ability to drop unimportant words from a bunsetsu, even though it is loosely based on bunsetsu-based methods.

5 Experiments

In this section, we report two experiments conducted to evaluate the proposed method.

5.1 Test Set

There is no standard test set for Japanese sentence compression. We therefore constructed a test set to evaluate the proposed method. The construction process is as follows.

First, we extracted 240 sentences from Kyoto University Text Corpus (Kurohashi and Nagao, 1998),[2] a parsed corpus of Mainichi Shimbun 1995. More precisely, we extracted sentences that satisfied all of the following three conditions. (1) The sentence was a lead sentence (the first sentence of an article). Lead sentences are often used in experiments for sentence compression because they can be compressed without consideration of their context. (2) The length of the sentence was not too short and not too long. We employed the number of characters in a sentence as the sentence length and extracted sentences not shorter than 51 characters and not longer than 100 characters. (3) The sentence did not contain parentheses. This condition was considered because we found that even human subjects often could not compress content in parentheses (typically speech). From the 409 extracted sentences that satisfied these three conditions, we randomly selected 240 sentences for our experiments.

For each of the 240 sentences, two subjects produced compressed versions. The compression ratio was set to 0.7. For example, if the length of a sentence was 100 characters, each subject was asked to produce a compressed sentence whose length was not longer than 70 characters.

[2] http://nlp.ist.i.kyoto-u.ac.jp/EN/index.php?Kyoto%20University%20Text%20Corpus

Avg. # of characters per source sentence	75.1
Avg. # of words per source sentence	44.6
Avg. # of bunsetsu per source sentence	14.5
Avg. # of bunsetsu with a compound noun per source sentence	5.3

Table 1: Statistics of our test set.

Finally, from the 240 groups of a source sentence and its two compressed versions, we randomly selected 160 groups as our test set. We used the remaining 80 groups as a development set to tune the proposed method. The statistics of our test set are given in Table 1.

5.2 Methods

In our experiments, we compared the outputs of the following methods.

WORD (RANDOM) Word-based method. This method randomly selected a subset of words from a source sentence as the compressed sentence.

WORD Word-based method described in Section 2.1. The optimal subset of words in a source sentence was selected using ILP.

BNST (RANDOM) Bunsetsu-based method that randomly selected a subset of bunsetsu from a source sentence as the compressed sentence.

BNST Bunsetsu-based method described in Section 3.1. The optimal subset of bunsetsu in a source sentence was selected using ILP.

BNST w/ DPND Bunsetsu-based method. We added dependency constraints to BNST.

PROP Proposed method described in Section 4.1. Using Lagrangian relaxation, unimportant words were dropped from a compound noun in a bunsetsu.

PROP w/ DPND Proposed method. We added dependency constraints to PROP.

HUMAN Human compression. For each source sentence in our test set, one of the two compressed sentences produced by subjects were randomly selected.

For WORD, BNST, BNST w/ DPND, PROP, and PROP w/ DPND, we used lp_solve (a mixed ILP solver[3]). In addition, we set $Score(w_i)$ and $Score(b_j)$ as follows. First, from articles in the newspaper Mainichi Shimbun from 1991 to 2002, we extracted pairs of a lead sentence and a title that could be viewed as a source sentence and its pseudo compression. Note that articles in 1995 were excluded because they overlapped with our test set. Moreover, we viewed a lead sentence and a title as a source sentence and its pseudo compression if the lead sentence contained more than 80% of content words in the title. We then calculated the rate of occurrence of a word in the titles to that in the lead sentences. For example, if a word appeared 50 times in the titles and 100 times in the lead sentences, the rate of occurrence of the word was 0.5. We used this rate as $Score(w_i)$ and set $Score(b_j) = \sum_{w_i \in b_j} Score(w_i)$.

Furthermore, we set $\alpha^{(t)} = \alpha^{(0)}/t$ for PROP and PROP w/ DPND. Note that $\alpha^{(0)}$ was set to 0.02 according to our development set. T was set to 100.

[3] lpsolve.sourceforge.net/5.5

	ROUGE 1	ROUGE 2
WORD (RANDOM)	0.690	0.409
WORD	0.736	0.540
BNST (RANDOM)	0.679	0.529
BNST	0.745	0.615
BNST w/ DPND	0.772†	0.653†
PROP	0.751	0.616
PROP w/ DPND	0.796†‡	0.671†‡

Table 2: Information content in a compressed sentence. † and ‡ mark statistically significant improvement over BNST and BNST w/ DPND with $p < 0.01$, respectively.

5.3 Information Content of a Compressed Sentence

In the first experiment, we examined how well our method performed in preserving important information in a source sentence. Using the methods described in the previous section, we compressed each source sentence in our test set. The compression ratio was set to 0.7. We then computed Recall-Oriented Understudy for Gisting Evaluation (ROUGE) scores (Lin, 2004) for each of the methods. That is, we measured the *n*-gram overlap between the outputs of each method and those of human subjects. In the calculation of ROUGE, stopwords were not removed. In addition, each word in a compression was normalized using the Japanese morphological analyzer JUMAN (Kurohashi et al., 1994).

Table 2 gives the results. We can see that PROP significantly outperformed WORD, especially in terms of ROUGE 2. The reason for this is that unlike WORD, PROP basically selected words in each bunsetsu in a source sentence as a unit.

When dependency constraints were not considered, PROP performed better than BNST. However, the performance differences between them were small and only the difference in ROUGE 1 was statistically significant (Wilcoxon signed-rank test, $p < 0.05$). On the other hand, when dependency constraints were considered, PROP significantly outperformed BNST. This time, the differences in ROUGE 1 and ROUGE 2 were both statistically significant ($p < 0.01$).

We found that the differences between PROP w/ DPND and BNST w/ DPND were due to the number of available bunsetsu that each method could select. More precisely, PROP w/ DPND selected 4.6% more (shortened) bunsetsu in a source sentence than BNST w/ DPND. Suppose that bunsetsu b_j is located at a deep node in a dependency tree of a source sentence (i.e., b_j depends on $b_{j'}$ and $b_{j'}$ depends on $b_{j''}$ and ... depends on b_J). To select b_j, both methods first have to select from $b_{j'}$ to b_J owing to the dependency constraints. However, since there is a length constraint, it is usually difficult to select b_j at such a deep node even if b_j contains important words. However, PROP w/ DPND has more chance of selecting b_j than BNST w/ DPND. This is because PROP w/ DPND can make room to select b_j by dropping unimportant words from other bunsetsu. In this way, PROP w/ DPND selected more bunsetsu that contained important words and achieved higher performance than BNST w/ DPND.

In contrast, when dependency constraints were not considered, the number of (shortened) bunsetsu that PROP selected was not so different from the number of that BNST selected (the difference was 2.4%). This is because PROP ignored dependency constraints and almost greedily selected bunsetsu that were composed of many important words. In other words, PROP had less opportunity to drop unimportant words. As a result, the differences between the performances of the two methods were not so large.

	Grammaticality
WORD	1.50
BNST	2.23
BNST w/ DPND	4.15
PROP	2.18
PROP w/ DPND	4.14
HUMAN	4.85

Table 3: Grammaticality of a compressed sentence. The score ranges from 1 to 5: 1 (very poor), 2 (poor), 3 (average), 4 (good), and 5 (very good).

5.4 Grammaticality of a Compressed Sentence

In the second experiment, we examined how well our method performed in producing grammatical compressions. We randomly selected 50 source sentences from our test set and obtained the outputs of six methods for those sentences. Note that the six methods were WORD, BNST, BNST w/ DPND, PROP, PROP w/ DPND, and HUMAN. Then, for each of the 50 source sentences, we presented the six outputs to five subjects and asked them to rate the outputs in terms of grammaticality. The subjects were all native Japanese speakers and did not include the two subjects who constructed our test set. They were told that all outputs were automatically generated. For each of the source sentences, the order of the outputs was randomized.

Table 3 presents the results. From the table, we can confirm that PROP produces compressions that are more grammatically correct than the compressions produced by WORD. This is because PROP was loosely based on bunsetsu-based methods and basically selected each bunsetsu as a unit.

For the same reason, PROP achieved comparable performance with BNST. Note that the performance of PROP was slightly worse than that of BNST. This is because words that should not be dropped from a bunsetsu were dropped by PROP. For example, PROP shortened the bunsetsu "*unyu syou wa*" to "*shou wa*" ("*unyu*", "*syou*", and "*wa*" mean transport, ministry, and TOP, respectively). In Kyoto University Text Corpus, which we used in our experiments, "*shou*" (ministry) was tagged as a noun. However, unlike usual nouns, the word cannot be located at the beginning of a bunsetsu. This is because in Japanese, the word has a strong suffix nature. Thus, we must not drop "*unyu*" (transport) from the bunsetsu and we should treat "*unyu*" (transport) and "*syou*" (ministry) as a unit. We found that most errors arising when employing our method related to such words, which could be viewed as both a noun and suffix (e.g., "*kai*" (meeting), "*jin*" (-ese)). As a future work, we need to properly handle these words.

PROP achieved good performance when we added dependency constraints to the method. The score of PROP w/ DPND was 4.14. This result indicates that the grammaticality of the compressions produced by PROP w/ DPND was generally good. The reason why the performance of PROP w/ DPND was slightly worse than that of BNST w/ DPND is the same as the reason described in the previous paragraph. As seen in the first experiment, dependency constraints were also effective in preserving important information. Thus, we can say that there is no reason for not using the constraints in sentence compression.

To verify that the grading was consistent, we computed correlation coefficients between every pair of our five subjects (i.e., between subjects 1 and 2, 1 and 3, ..., and 4 and 5). Consequently, we found that the average of the coefficients was 0.45 and those coefficients were all statistically significant (t-test, $p < 0.01$). These suggest that the grading was consistent.

Source sentence
中米ホンジュラスの/軍司令官は/１５日、/４０年間に/わたり/管轄してきた/警察部隊を/文民の/指揮下に/置くと/発表した。[52]
(On the 15th, an army commander of Honduras, located in Central America, announced that the police divisions that the army had controlled for 40 years would be placed under civilian control.)

WORD
中米ホンジュラスの軍司令官は日、４０年間に管轄警察部隊を文民の指揮下にと [36]
Ungrammatical sentence

BNST w/ DPND
中米ホンジュラスの/軍司令官は/警察部隊を/文民の/指揮下に/置くと/発表した。[34]
(An army commander of Honduras, located in Central America, announced that the police divisions would be placed under civilian control.)

PROP w/ DPND
ホンジュラスの/軍司令官は/１５日、/警察部隊を/文民の/指揮下に/置くと/発表した。[36]
(On the 15th, an army commander of Honduras announced that the police divisions would be placed under civilian control.)

HUMAN
ホンジュラスの/軍司令官は/１５日、/警察部隊を/文民の/指揮下に/置くと/発表した。[36]
(On the 15th, an army commander of Honduras announced that the police divisions would be placed under civilian control.)

Figure 4: Example of the outputs. For explanation purposes, we inserted slashes between bunsetsu in each of the outputs except for WORD. The values in square bracket denote sentence lengths.

5.5 Example of the Outputs

Figure 4 gives an example of the output of our method (PROP w/ DPND). For comparison, we also show the outputs of WORD, BNST w/ DPND, and HUMAN. In this example, the source sentence comprises 52 characters. Thus, each method was required to produce a compressed sentence whose length was not more than 36 characters ($52 \times 0.7 = 36.4$).

In this example, WORD produced a completely ungrammatical compression. In contrast, BNST w/ DPND produced a good compression, which was grammatical and preserved much information in the source sentence. However, in the output of HUMAN, the first bunsetsu “中米ホンジュラスの” in the source sentence was shortened to “ホンジュラスの” (“中米”, “ホンジュラス”, and “の” mean Central America, Honduras, and GEN, respectively). BNST w/ DPND could not perform such an operation because the method has to treat each bunsetsu as it is.

On the other hand, PROP w/ DPND could shorten the bunsetsu by dropping word “中米”, which had less importance, from the bunsetsu ($Score$(“中米”) was 0.119). Additionally, using the room that was made by dropping the word, the method could add another bunsetsu “１５日、” to the compression (“１５”, “日”, and “、” mean 15, day, and comma, respectively). Although there are other bunsetsu that contain compound nouns (e.g., “軍司令官は”), our method did not drop any words from those bunsetsu and selected them as a unit similarly to BNST w/ DPND (“軍”, “司令” + “官”, and “は” mean army, commander, and TOP, respectively). This is because the words in the bunsetsu had great importance (e.g., $Score$(“軍”) was 0.378). In this way, PROP w/ DPND could produce the same compression as HUMAN.

6 Related Work

Sentence compression has been widely studied since the early 2000s. For the English language, Jing used multiple knowledge resources to decide which phrases in a source sentence to remove (Jing, 2000). Knight and Marcu modeled a generative process of a source sentence based on a noisy-channel framework and generated a compressed sentence using the model (Knight and Marcu, 2002). Turner and Charniak presented semi-supervised and unsupervised variants of the Knight and Marcu's model (Turner and Charniak, 2005). McDonald employed a discriminative model to learn which words in a source sentence should be dropped (McDonald, 2006). Clarke and Lapata recasted previous methods as ILP and extended those with various constraints (Clarke and Lapata, 2008). Our work differs from these efforts in that we focus on Japanese sentence compression.

For the Japanese language, previous compression methods can be divided into two groups: word-based methods and bunsetsu-based methods. Hori and Furui proposed a word-based method to summarize speech (Hori and Furui, 2004). They extracted a set of important words from an automatically transcribed sentence. Hirao et al. also proposed a word-based method (Hirao et al., 2009). They extended Hori and Furui's method using novel features. Unlike these methods, our method is loosely based on bunsetsu-based methods, and thus easily generates grammatical compressions.

Previous bunsetsu-based methods are given below. Takeuchi and Matsumoto used a support vector machine to acquire rules for dropping unimportant bunsetsu in a source sentence (Takeuchi and Matsumoto, 2001). Oguro et al. and Yamagata et al. defined varying importance of bunsetsu and dependency and extracted the optimal subset of bunsetsu from a source sentence (Oguro et al., 2002; Yamagata et al., 2006). Nomoto generated candidates for a compression by removing bunsetsu from a source sentence and selected the best candidate using a conditional random field (Nomoto, 2008). Our work differs from these efforts in that our method has the ability to drop unimportant words from a bunsetsu.

In our method, we used Lagrangian relaxation to relax unit constraints. Lagrangian relaxation is a well known technique for combinatorial optimization and it has recently been successfully applied to various natural language processing tasks (Koo et al., 2010; M.Rush et al., 2010; Chang and Collins, 2011; M.Rush and Collins, 2011). However, to the best of our knowledge, this is the first work to use the technique for sentence compression.

Conclusions and Future Work

We presented a novel compression method for a Japanese sentence. The proposed method was loosely based on bunsetsu-based methods. Thus, unlike word-based methods, it could easily produce grammatical compressions. Additionally, using Lagrangian relaxation, the proposed method relaxed constraints that troubled bunsetsu-based methods. In this way, unlike bunsetsu-based methods, our method could shorten each bunsetsu if it contained unimportant words. Experimental results showed that the proposed method could preserve more information in a source sentence than word-based and bunsetsu-based methods. Furthermore, we confirmed that our method could produce grammatical compressions similarly to bunsetsu-based methods.

In future work, as described in Section 4.1, we plan to explore a technique to handle proper nouns such as organization names. Additionally, as described in Section 5.4, we need to develop a method to handle words that can be viewed as both a noun and suffix.

References

Chang, Y.-W. and Collins, M. (2011). Exact Decoding of Phrase-Based Translation Models through Lagrangian Relaxation. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing (EMNLP 2011)*, pages 26–37.

Clarke, J. and Lapata, M. (2008). Global Inference for Sentence Compression: An Integer Linear Programming Approach. *Journal of Artificial Intelligence Research*, 31:399–429.

Corston-Oliver, S. (2001). Text Compaction for Display on Very Small Screens. In *Proceedings of the Workshop on Automatic Summarization at the 2nd Meeting of the North American Chapter of the Association for Computational Linguistics (WAS 2001)*, pages 89–98.

Hirao, T., Suzuki, J., and Isozaki, H. (2009). A Syntax-Free Approach to Japanese Sentence Compression. In *Proceedings of the 47th Annual Meeting of the Association for Computational Linguistics and the 4th International Joint Conference on Natural Language Processing (ACL-IJCNLP 2009)*, pages 826–833.

Hori, C. and Furui, S. (2003). A New Approach to Automatic Speech Summarization. *IEEE Transactions on Multimedia*, 5(3):368–378.

Hori, C. and Furui, S. (2004). Speech Summarization: An Approach through Word Extraction and a Method for Evaluation. *IEIEC Transactions on Information and Systems*, E87-D(1):15–25.

Jing, H. (2000). Sentence Reduction for Automatic Text Summarization. In *Proceedings of the 6th Applied Natural Language Processing Conference (ANLP 2000)*, pages 310–315.

Knight, K. and Marcu, D. (2002). Summarization beyond sentence extraction: A probabilistic approach to sentence compression. *Artificial Intelligence*, 139(1):91–107.

Koo, T., M.Rush, A., Collins, M., Jaakkola, T., and Sontag, D. (2010). Dual Decomposition for Parsing with Non-Projective Head Automata. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing (EMNLP 2010)*, pages 1288–1298.

Korte, B. and Vygen, J. (2008). *Combinatorial Optimization: Theory and Algorithms*. Springer Verlag.

Kurohashi, S. and Nagao, M. (1998). Building a Japanese Parsed Corpus while Improving the Parsing System. In *Proceedings of the 1st International Conference on Language Resources and Evaluation (LREC 1998)*, pages 719–724.

Kurohashi, S., Nakamura, T., Matsumoto, Y., and Nagao, M. (1994). Improvements of Japanese Morphological Analyzer JUMAN. In *Proceedings of the International Workshop on Sharable Natural Language*, pages 22–28.

Lin, C.-Y. (2003). Improving Summarization Performance by Sentence Compression — A Pilot Study. In *Proceedings of the 6th International Workshop on Information Retrieval with Asian Languages (IRAL 2003)*, pages 1–8.

Lin, C.-Y. (2004). ROUGE: A Package for Automatic Evaluation of Summaries. In *Proceedings of the Workshop on Text Summarization Branches Out (WAS 2004)*, pages 74–81.

McDonald, R. (2006). Discriminative Sentence Compression with Soft Syntactic Evidence. In *Proceedings of the 11th Conference of the European Chapter of the Association for Computational Linguistics (EACL 2005)*, pages 297–304.

M.Rush, A. and Collins, M. (2011). Exact Decoding of Syntactic Translation Models through Lagrangian Relaxation. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics on Human Language Technologies (ACL-HLT 2011)*, pages 72–82.

M.Rush, A., Sontag, D., Collins, M., and Jaakkola, T. (2010). On Dual Decomposition and Linear Programming Relaxations for Natural Language Processing. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing (EMNLP 2010)*, pages 1–11.

Nomoto, T. (2008). A Generic Sentence Trimmer with CRFs. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics on Human Language Technologies (ACL-HLT 2008)*, pages 299–307.

Oguro, R., Sekiya, H., Morooka, Y., Takagi, K., and Ozeki, K. (2002). Evaluation of a Japanese Sentence Compression Method Based on Phrase Significance and Inter-Phrase Dependency. In *Proceedings of the 5th International Conference on Text, Speech and Dialogue (TSD 2002)*, pages 27–32.

Takeuchi, K. and Matsumoto, Y. (2001). Acquisition of Sentence Reduction Rules for Improving Quality of Text Summaries. In *Proceedings of the 6th Natural Language Processing Pacific Rim Symposium (NLPRS 2001)*, pages 447–452.

Turner, J. and Charniak, E. (2005). Supervised and Unsupervised Learning for Sentence Compression. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL 2005)*, pages 290–297.

Vandeghinste, V. and Pan, Y. (2004). Sentence Compression for Automated Subtitling: A Hybrid Approach. In *Proceedings of the Workshop on Text Summarization Branches Out (WAS 2004)*, pages 89–95.

Yamagata, K., Fukutomi, S., Takagi, K., and Ozeki, K. (2006). Sentence Compression Using Statistical Information About Dependency Path Length. In *Proceedings of the 9th International Conference on Text, Speech and Dialogue (TSD 2006)*, pages 127–134.

Zajic, D., J.Dorr, B., Lin, J., and Schwartz, R. (2007). Multi-Candidate Reduction: Sentence Compression as a Tool for Document Summarization Tasks. *Information Processing and Management Special Issue on Summarization*, 43(6):1549–1570.

Approximating theoretical linguistics classification in real data: the case of German *nach* particle verbs

Boris HASELBACH[1] *Kerstin* ECKART[1] *Wolfgang* SEEKER[1]
Kurt EBERLE[2,1] *Ulrich* HEID[3,1]
(1) IMS, Universität Stuttgart, Germany
(2) Lingenio GmbH, Heidelberg, Germany
(3) IWIST, Universität Hildesheim, Germany
haselbbs@ims.uni-stuttgart.de, eckartkn@ims.uni-stuttgart.de,
seeker@ims.uni-stuttgart.de, eberle@ims.uni-stuttgart.de,
heid@ims.uni-stuttgart.de

ABSTRACT
Testing a theory against real world data can sometimes be helpful in figuring out the shortcomings of your current theory. In this paper, we test a theory about the syntax-semantics interface of German *nach*-particle verbs against data from a web corpus in order to see if we can use our automatic NLP machinery to corroborate the predictions of the theory. We use state-of-the-art parsers to automatically annotate our data with the features predicted by the theory and then apply a standard clustering approach to approximate the *nach*-particle verb classes of the theory. The results of our experiment not only help us highlighting the more problematic parts of the theory but also teach us about the strengths and weaknesses of our automatic analysis tools.

TITLE AND ABSTRACT IN ANOTHER LANGUAGE, L_2 (OPTIONAL, AND ON SAME PAGE)

Corpusbasierte Überprüfung einer semantischen Klassifikation deutscher *nach*-Partikelverben

Um Unzulänglichkeiten einer Theorie auszumachen ist es mitunter vonnöten, Hypothesen gegen echtes Textmaterial abzugleichen. In diesem Beitrag soll diskutiert werden, wie Vorhersagen einer Theorie zum syntaktischen und semantischen Verhalten deutscher *nach*-Partikelverben gegen Netztexte abgeglichen werden können und wie dabei eine automatische Textverarbeitung unterstützend zum Tragen kommt. Es werden Parser des letzten Stands der Forschung verwendet um die Daten mit den von der Theorie vorhergesagten Merkmalen zu annotieren bevor ein standardisiertes Clustering-Verfahren angewandt wird um die theoretischen *nach*-Partikel-Verb-Klassen nachzubilden. Die Resultate des Experiments unterstreichen nicht nur Problemfälle der Theorie sondern zeigen auch die Stärken und Schwächen der automatischen Analyse.

KEYWORDS: particle verbs, syntax-semantics interface, German, web data, parser combination.

KEYWORDS IN L_2: Partikelverben, Syntax-Semantik-Schnittstelle, Deutsch, Netztexte, Parser-Kombination.

Proceedings of COLING 2012: Technical Papers, pages 1113–1128,
COLING 2012, Mumbai, December 2012.

1 Introduction

We test a theoretically well-motivated hypothesis about the syntax-semantics interface of German *nach*-particle verbs (*nach*-PV) on data from the "real world", to gain insight into the strengths and weaknesses of the theory as well as the limitations of our NLP-methodology. The theory we want to test is a word-syntactic approach to *nach*-PVs implemented in a generative framework. The analysis of five different readings of *nach* in combination with a verb makes different predictions on the argument structure of the complex verb, in particular with respect to dative and marginally to accusative arguments: the verb semantics (especially the *nach* particle) determines the argument structure of the verb. In order to test this assumption, we will turn the argumentation around: to what extent can we use automatically gained syntactic information to recreate the *nach*-PV classes. For this, we automatically annotate large amounts of data with their syntactic structure and identify the argument structure of each verb. Based on this information, we then apply a standard clustering technique to recreate the *nach*-PV classes proposed by the theory. As syntactic indicators for the clustering, dative and accusative arguments seem to be ideal candidates, as they are partially in the scope of the theoretical description. But we also incrementally add further features of potential relevance for the *nach*-PV reading classification: the form of arguments, prepositional phrases (form of preposition, governed case, form of embedded object), adverbials (adverbs and predicative adjectives), as well as clausal objects.[1].

Although automatic parsers are high quality and reliable tools nowadays, their performance degrades when applied to unrestricted all-domain data. Since we work on web text, we apply two very different parsers, which we then combine for a more reliable annotation. For a small subset of the *nach*-PV data, we use a manually created gold standard for dative and accusative arguments. Evaluating the parsers against this gold standard enables us to pinpoint the advantages of each parser and develop a combination scheme.

The paper is structured as follows. In Section 2, we introduce the theory about the *nach*-PV classes and their related argument structures. In Sections 3 to 5, we present the study in which we apply the theoretical approach to the real world data. In Section 3, we present the data we use: a German web corpus, a small gold standard, and a manual classification of all *nach*-PV lemmas from the corpus according to the theory from Section 2. The gold standard is for the evaluation and combination of the parsers, which extract syntactic features related to the *nach*-PVs (Section 4). The classification of *nach*-PV lemmas is used for the evaluation of the clustering based on the extracted syntactic features (Section 5). We conclude by discussing the lessons learned from each step of this study.

2 Phenomena and linguistic modeling

The German verb particle *nach* (≈ 'after') shows a range of different meanings. Haselbach (2011) provides a partial classification of *nach* into five readings that behave differently with respect to licensing a dative argument:

1. ⊕DAT: direction reading (DIR)
2. ⊕DAT: copy manner reading (MAN)
3. ⊖DAT: copy creation reading (CRE)
4. ⊖DAT: once-more/restitution reading (OMR)
5. ⊖DAT: continuation reading (CONT)

This classification does not cover yet e.g. an intensifying reading such as in *nachdenken* ('to reflect') or a prove/check reading such as in *nachprüfen* ('to recheck').

[1]For similar work on the German verb particle *an* (≈ 'on') see Springorum et al. (2012)

2.1 Modeling at the syntax-semantics interface

(Haselbach, 2011) provides a syntactico-semantic modeling of the five readings of *nach* in terms of Discourse Representation Theory (e.g. Kamp and Reyle, 1993; Roßdeutscher and Kamp, 2010), combined with word-syntactic principles from Distributed Morphology (Halle and Marantz, 1993). Haselbach (2011) implements them by means of the extended VP-shell hypothesis (Larson, 1990). His syntactico-semantic modeling comes close to Nicol's (2002) implementation who argues that verb particles are spelled out instances of functional heads in the verbal domain. Precisely, Haselbach (2011) argues that (i) *nach* either represents the head of a functional projection *w*P which is above VP and projects a dative argument in its specifier, cf. (1-a); or (ii) it is the realization of the head of a functional projection *x*P, which is also above VP however does not project a dative argument in its specifier, cf. (1-b).

(1) a. [$_{wP}$ DP$_{DAT}$ [$_{w'}$ *w*=“*nach*” VP]]
b. [$_{xP}$ *x*=“*nach*” VP]

The idea is that *nach* adds a second eventuality to the semantics by presupposition. The functional difference between *w* and *x* is that *w*, i.e. if a dative is present in the structure, allows *nach* to access event properties in the underlying VP, whereas *x* allows *nach* to access state properties in the underlying VP. Accessing event properties here means that properties of the event in the VP are assigned to the event presupposed by *nach*. Accessing event properties can either be the direction (class DIR) or the manner (class MAN) of an event. By presupposing a second **event** in the semantics, a slot for a further argument is created in the specifier of the functional projection headed by *nach*, i.e. in Spec,*w*P. This argument, which surfaces as dative, is interpreted as a participant in the presupposed event. Accessing state properties, on the other hand, means that properties of a state, if present, within the VP are assigned a presupposed **state**. These can be result or progression state properties. A result state property can either be the existence of an object, i.e. for creation verbs (class CRE), or predicational states contributed by a deadjectival or denominal verb (class OMR). Progressive state properties are stative run-time properties of an event, i.e. the state that can be described by means of the event taking place (class CONT). In these cases, no additional argument slot is semantically present, and thus no dative is licensed.

2.1.1 Five readings of *nach* and the dative

Distinguishing two groups of *nach* reading (plus vs. minus dative), Haselbach (2011) claims that *nach* expressing the meaning “following NP$_{DAT}$” (**directional**: DIR; cf. (2-a)) and *nach* expressing the meaning “do such as NP$_{DAT}$” (**manner**: MAN; cf. (2-b)) take a dative argument.

(2) a. Der Hund rannte dem Hasen nach.
the dog ran the.DAT hare after
“The dog ran after the hare.”
b. Das Mädchen betete der Mutter (den Psalm) nach.
the girl prayed the.DAT mother (the.ACC psalm) after
“The girl copied the mother's praying (of the psalm).”

As opposed to the two readings of *nach* in the context of which a dative is present, there are three readings of *nach* where no dative is present. The first reading is the creation, or copy object reading which can be paraphrased as “making a copy of Y” (**creation**: CRE; cf. (3-a)). The second one is the **once-more/restitution** reading (OMR), which itself has two sub-meanings: a repetitive (once-more) and a restitution reading; cf. (3-b). Haselbach (2011) groups these two readings together as it is not *nach* that is liable for the semantic distinction,

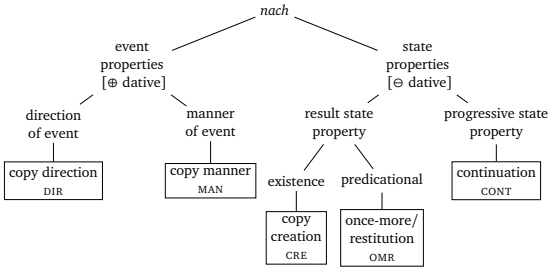


Figure 1: Classification of *nach* readings

but it is considered to be a discourse effect of how the eventualities emerging in the derivation are ordered temporally. The third reading without dative is the **continuation** reading (CONT; cf. (3-c)). In this reading, *nach* conveys a meaning that can be roughly paraphrased as "do something longer than expected". Figure 1 gives an overview of all readings.

(3) a. Der Junge baute den Eiffelturm nach.
the boy built the.ACC Eiffel Tower after
"The boy made a copy of the Eiffel Tower."
b. Der Schmied schärfte das Messer nach.
the blacksmith sharpened the.ACC knife after
"The blacksmith re-sharpened the knife."
c. Die Banane reifte nach.
the banana ripened after
"The banana continued ripening."

2.1.2 Five readings of *nach* and the accusative

Haselbach (2011) stays agnostic about the role of direct objects (i.e. accusative NPs) with respect to the readings of *nach*. With the DIR class, an accusative object may be present or absent, the latter being preferred, cf. (4).

(4) a. Der Hund rannte dem Hasen nach.
the dog ran the.DAT hare after
"The dog ran after the hare."
b. Der Lausbub warf ihr den Ball nach.
the scallywag threw her.DAT the.ACC ball after
"The scallywag threw the ball after her."

For the MAN reading, we encounter both predicates with and without an accusative object, as in (5). Thus, an accusative object does not correlate with the MAN-reading.

(5) Homer tanzte Marge (den Tango) nach.
Homer danced Marge.DAT (the.ACC tango) after
"Homer copied Marge's dancing (of the tango)."

For both the CRE and OMR classes, an accusative object seems to be obligatory. If no dative is present (which is fundamental for these readings), the accusative in (6-a) for CRE, and (7) for OMR cannot be left out. Thus, we consider the direct object to be crucial for these two classes. Replacing the accusative by a dative argument, cf. (6-b), also leads to grammaticality. Then, *nach* is considered to belong to the MAN class.

(6) a. Die Oma strickte *(die Mütze) nach.
the grandmother knitted the.ACC cap after
"The grandmother copied the cap by knitting."
b. Die Oma strickte der Uroma nach.
the grandmother knitted the.DAT great granny after
"The granny copied the great granny's knitting manner."

(7) Der Opa salzte *(die Suppe) nach.
the grandfather salted the.ACC soup after
"The grandfather added more salt to the soup."

The continuation reading of *nach* is not compatible with an accusative argument. This is because this reading seems to correlate with verbs where the verb-internal argument surfaces as subject, i.e. with anti-causative verbs such as in (8). In combination with a verb such as *reifen* ('to ripen') as in (8), neither an accusative, nor a dative, nor both can be present.

(8) Die Banane reifte (*dem Apfel) (*den Pfirsich) nach.
the banana ripened the.DAT apple the.ACC peach after
"The banana continued ripening."

2.1.3 MAN and CRE: one or two classes?

As we already saw in example (6-a) in Section 2.1.2, the *nach*-PV readings MAN and CRE are quite close. Now one could ask whether these two classes are really distinct. Note that the *nach*-PV classes discussed by Haselbach (2011) are interpretation-driven, i.e. in the context of a particular verb, the particle *nach* can evolve a certain interpretation with respect to the nature of the verb. This does not mean that each and every verb does only allow one interpretation. On the contrary, there are many verbs in the context of which *nach* exhibits several of the readings presented above. For example, the complex verb *nachtanzen* (derived from *tanzen*, 'to dance') shows at least three readings: (i) a DIR reading ("follow someone dancing"), (ii) a MAN reading ("copy somebody's dancing manner"), and (iii) a CRE reading ("copy a dance"). DIR and MAN require a dative, i.e. the "somebody" followed or copied, while CRE does not. Nevertheless, the interpretations of MAN and CRE are quite close to each other; verbs expressing the manner of process, such as *stricken* ('to knit', i.e. to produce with wool), *sprechen* ('to speak', i.e. to produce speech), or even *tanzen* ('to dance', i.e. to produce a dance), can also be interpreted as verbs of creation if used with an accusative object (mostly incremental theme, cf. Dowty 1991). Thus, we expect that these two classes are empirically not easily ascertainable.

2.2 Indicators of semantic readings observable in corpus data

In principle, dative and accusative could function as indicators for an automatic approximation of the *nach*-PV readings. However, they are problematic as there is no 1-to-1 relation between indicators and readings, cf. Table 1: datives seem to be better indicators (separating DIR/MAN from the rest) than accusatives, which are unevenly distributed. However, the discriminative power of dative is weakened by the closeness of MAN and CRE, cf. Section 2.1.3. Given this situation, we will check to what extent the use of further features will improve the result of an automatic classification. We will use, incrementally, details of the dative and accusative arguments, PPs, adverbials, as well as clausal complements.

3 Data

As our real world data, we use a fragment of a cleaned version of the German web corpus deWaC (Baroni and Kilgarriff, 2006). We automatically extracted 270k sentences containing at least one *nach*-PV.

indicator	DIR	MAN	CRE	OMR	CONT
dative	+	+	−	−	−
accusative	(−)	?	+	+	−

Table 1: Argument structure indicators for *nach*-PV classes

3.1 Manual gold standard for syntactic criteria for *nach*-PV

From this 270k *nach* sub-corpus, we extracted 270 sentences containing *nach* verbs (i) of each reading (i.e. DIR, MAN, CRE, OMR, and CONT) and (ii) of the frequency ranges high, middle, and low frequency. The set of 270 sentences functions as the gold standard, which we use to evaluate the automatic dependency parsers.[2]

Each *nach*-PV token was annotated by five annotators: two linguistically trained, three untrained. The annotation contains the dative argument and/or the accusative argument of each *nach*-PV token, if present. The linguistically untrained annotators only indicated the presence of a dependent dative or accusative argument. The linguistically trained annotators marked the extension of the argument under consideration, i.e. they annotated the entire textual string. Table 2a shows the inter-annotator agreement. The identification of dative and accusative (Boolean feature) is feasible even for untrained annotators (substantial agreement[3] on all annotators); the linguists even achieved an almost perfect agreement on the extension of the arguments. For the gold standard, the longest extension possible, i.e. a dependent NP containing all modifiers such as adjoined relatives clauses, etc., was taken.

3.2 Interaction of indicators with syntactic constructions

The syntactic features dative and accusative interact with certain "argument-structure-changing" constructions found in the corpus, which will impact on parsing. Basically, we encounter two types of construction that interfere with the argument structure of the verb: constructions that "reduce" the argument structure at the surface and constructions that seemingly "extend" it.

Argument structure reduction. The diatheses passive and middle, as well as null instantiations[4] (NI) "reduce" the argument structure of the verb. In passives (eventive, stative, impersonal, etc.) the subject of the verb is demoted and the direct object surfaces as subject, marked with nominative case. Eventive passives which are indicated with the auxiliary *werden* are identifiable, however other types of passives such as adjectival passives, which are identical with predicative adjective constructions, cannot be identified by the parsers. Middles, e.g. (9), behave similar to the passive with respect to subject and object. In middles, the internal object, which is marked with accusative in the standard active form, surfaces as nominative. The true external subject is demoted. However, middles are usually not identifyed by automatic parsers.

(9) Die Rumba tanzt sich leicht nach.
the rumba dances REFL easy after
"The rumba dances easily."

Another issue that arises is the pragmatically context-driven demotion of arguments. In a sentence such as (10-a), a direct object is clearly expected for verb *schärfen* ('to sharpen') to

[2] In the original gold standard set, there were 277 sentences. However, seven did not contain a *nach*-PV. The parser erroneously identified *nach* and verb as a *nach*-PV; e.g. *Falten Sie das Papier der Länge nach* ('Fold the paper lengthwise').

[3] Significance according to Landis and Koch (1977).

[4] Cf. Fillmore et al. (2003).

predicate over. However, in operating instructions for things that need to be sharp and that are operated manually, e.g. a knife or a pair of scissors, it seem perfectly fine to leave out the direct object. Similarly, in (10-b), the theory would predict a dative argument of *nachtanzen*, which seems to be correct if the *nachtanzen*-clause would occur in isolation. However in this example, the dative can be omitted easily as the reference of the dancing manner can be identified locally in the same sentence with the dance instructor.

(10) a. Wenn Sie mehr Druck ausüben müssen, sofort Ø nachschärfen!
if you more pressure exert must immediately Ø.ACC after-sharpen
"If you have to exert more pressure, re-sharpen (the knife) immediately!"
b. Der Tanzlehrer tanzte genau vor und ich tanzte Ø nach.
the dance instructor danced exactly before and I danced Ø.DAT after
"The dance instructor showed how to dance and I followed his dancing manner."

Argument structure extension. On the other side, there are constructions that seemingly "extend" the argument structure: ACI-constructions (*Accusativus cum infinitivo*), dative benefactives, and accusative temporal adverbials increase the number of accusative- or dative-marked NPs recognized by a parser. These constructions mentioned above do not extend the argument structure of the verb, however they change the observable amount of argument-like phrases by raising the subject of an embedded clause to the object of the matrix clause (ACI-construction as in (11)), or by adding a noun phrase that could erroneously be identified as an argument of the verb (dative benefactive as in (12-a) and accusative temporal adverbial as in (12-b)).

(11) Ich höre die Glocke nachklingen.
I hear the.ACC bell after-sound
"I hear the bell linger on."

(12) a. Die Oma strickte dem Baby die Mütze nach.
the grandmother knitted the.DAT baby the cap after
"The grandmother made a knitted copy of the cap for the baby."
b. Die Banane reifte eine Woche nach.
the banana ripened a.ACC week after
"The banana continued ripening for one week."

Table 2b shows the agreement of the linguistically trained annotators on the valency-changing constructions and their frequencies in the gold standard, where they affect 32.49 % of the *nach*-PV instances. Dative benefactives and accusative temporal adverbials are to be added.

a.

indicator	κ (all annot.)	κ (trained annot.)
accusative	0.699	0.967
dative	0.869	0.985

b.

phenomenon	κ	frequency
passive	0.971	67
middle	1.000	4
ACI	1.000	4
NI	0.774	15

Table 2: Inter-annotator agreement on a. indicators and b. valency-changing constructions

3.3 Manual *nach*-PV classification

To obtain a gold standard of the *nach*-PV lemmas with respect to the five *nach* readings described above, three annotators, who were familiar with the classes by Haselbach (2011), manually classified 475 *nach*-PV lemmas that where extracted from the corpus. Each annotator decided for each lemma (without context) if it exhibits one particular *nach*-reading, e.g. DIR. If at least two annotators labeled a lemma with a reading, the lemma was accounted to the *nach*-reading (majority decision). Multiple labels for one lemma were allowed (reflecting polysemy).

Table 3 shows the verb classes and verb class combinations of the *nach*-PV lemmas identified by the annotators. 246 of 475 *nach*-PV lemmas were classified as belonging to at least one of the five classes described by Haselbach (2011). Table 3 also illustrates that a manual classification of the *nach*-PV lemmas without context is a non-trivial task. Albeit the majority of *nach*-PV lemmas annotated with at least one of Haselbach (2011)'s classes clearly belong to one class (cf. rows 1 to 5 in Table 3) – or at least, as expected, to the mixed class {MAN,CRE} – there are many verbs that show different *nach* readings. This, of course, is not fatal for the theory, on the contrary, it is expected. However, for the automatic identification this might pose a problem. The ranks 1 to 5 (more than 10 lemma types) mostly cover single classes. Under the top-5 there is only one complex class, the MAN/CRE-class, which is expected as discussed in Section 2.1. Examples of the classes that are classified as expected are: *nachfetten* ('to regrease') or *nachsalzen* ('to add more salt') for OMR; *nachlaufen* ('to run after sb.') or *nachblicken* ('to gaze after sb.') for DIR; *nachbacken* (lit.: 'after'+'bake', to copy sth. by baking it/to copy sb.'s baking of sth.) for {MAN,CRE}; or *nachbluten* (lit.: 'after'+'bleed', to continue bleeding) for CONT.

Nevertheless, the verbs in the tail of list, i.e. from rank 6 on, seem randomly distributed. This shows that an *ad hoc* classification of the verbs without context is difficult because annotators might come up spontaneously with a reading that is rather coerced than predicted, or the other way around they might miss a reading because it is less common but perfectly grammatical. The unbalanced distribution of the verbs in the lower part of the list therefore also indicate that there might be verbs in the upper, apparently clear part of the table that might also be coerced to a particular reading of *nach*, and it might also have happened by accident that the annotators did not come up with a particular reading.[5]

rank	reading class	frequency	rank	reading class	frequency
1	OMR	67		{MAN,CRE,OMR}	6
2	DIR	58	9	{DIR,CONT}	5
3	{MAN,CRE}	26	10	{CRE,OMR}	4
4	MAN	20	11	{DIR,CRE}	3
5	CONT	19		{DIR,OMR}	3
6	{MAN,OMR}	8	12	{DIR,OMR,CONT}	2
	{DIR,MAN,CRE}	8		CRE	2
7	{DIR,MAN}	7	13	{MAN,CRE,CONT}	1
8	{OMR,CONT}	6		{DIR,MAN,CRE,CONT}	1
				overall	246

Table 3: Manual *nach*-PV classification

Additionally, the inter-annotator agreement of the classification is poor, cf. Table 4. The probability that all three annotators agreed on a particular class is not bad (P), however, as the distribution of the *nach*-readings over the lemmas is rather unbalanced the probability of an accidental match with the binary features is high (Pe). Thus κ is poor.

4 Tools and analyses of corpus data

To identify different classes of *nach*-PVs based on the corpus data, we make use of two dependency parsers and a relational database infrastructure (cf. Eckart et al., 2010), to extract

[5] An example in the data set is *nachlabern* (lit.: 'after'+'babble', paraphrase: derogatively reciting sth. in a babbling manner), which is labeled as exclusively belonging to the MAN class. A careful scrutiny however shows that *nachlabern* can also occur with the CRE reading.

nach-reading	P	Pe	κ
DIR	0.889	0.689	0.642
MAN	0.716	0.642	0.205
CRE	0.865	0.774	0.403
OMR	0.755	0.626	0.346
CONT	0.876	0.811	0.344

Table 4: Inter-annotator agreement on *nach* reading wrt. lemmas

syntactic features appearing with the *nach*-PVs. We chose two parsers based on different concepts to complement one another, i.e. by taking the individual strengths of each parser into account when extracting indicators from their results.

The **Bohnet-Parser** (BP), cf. Bohnet (2010), is a data-driven state-of-the-art dependency parser. It makes use of a rich feature model and a second order maximum-spanning-tree algorithm (McDonald and Pereira, 2006). The parser also includes its own processing pipeline containing statistical lemmatization, part-of-speech tagging, and morphological tagging on an already tokenized input. The output structure are non-projective dependency trees in the tabular representation format of CoNLL 2009's shared task. Regarding labeled syntactic accuracy in this shared task, the Bohnet-Parser was the second best system and the best system for English and German. The model we utilized in our experiments was trained on a dependency version of the German TiGer treebank (Brants et al., 2002), as described in Seeker and Kuhn (2012).

FSPar (FP), cf. Schiehlen (2003), is a rule-based dependency parser based on the approach by Abney (1996) of partial parsing by finite state cascades. FSPar also processes its own internal pipeline, which includes lexically informed tokenizing and lemmatizing and part-of-speech tagging with the TreeTagger, cf. Schmid (1994). Not only the tokenizing step but also the parsing makes use of a large integrated lexical knowledge base, e.g. including named entities. FSPar generates underspecified dependency graphs. Underspecification is applied with respect to head selection as well as dependency labels. If the attachment to a head is ambiguous for a specific token or if the head is part of a coordination or is a combined verb form, more than one head token is given. Multiple or underspecified dependency labels occur either because of mulitple head possiblilties or because of ambiguous dependency relations.

4.1 Parsing results: recognizing the indicators

Both parsers were evaluated against the test set described in Section 3.1. Three evaluation criteria were applied, each time taking all 277 sentences into account. The first evaluation is on *nach*-PV recognition, including cases where a sentence contains more than one or no *nach*-PV. The sentences that contain no *nach*-PV are difficult for the parsers as they contain the token *nach* and a verb for which the *nach* could denote a separable verb particle but does not so in the sentence under analysis. The other two criteria are the recognition of dative and accusative arguments, where in a correct case the parser extracted an argument of the expected case, and its head is contained in the argument string represented in the gold standard.

Table 5 shows the results of the evaluation. For the Bohnet-Parser, verbs are identified as *nach*-PVs, in case of *nach* being a part of the lemma, or having a token *nach* being the dependent of the verb, where the dependency label or the part-of-speech tag identified a separable verb particle. Accusative and dative arguments are identified by the respective dependency label on a relation having the *nach*-PV as its head.

Bohnet-Parser's *nach*-PV recognition suffers from the fact that the statistical lemmatizer sometimes produces a wrong lemma, even if the right token is recognized as a *nach*-PV. Dative recognition outperforms accusative recognition because dative arguments are morphologically marked more clearly. So although the Bohnet-Parser recognizes a dative only with a recall of 56.67, if it does annotate one, it is mostly correct with a precision of 87.18.

	Bohnet-Parser				FSPar		
	prec	rec	f1		prec	rec	f1
NPV recognition	93.62	95.65	94.62		97.53	100.0	98.75
dative recognition	87.18	56.67	68.69	upper bound	61.11	91.67	73.33
				lower bound	50.00	75.00	60.00
				chance	52.22	78.33	62.67
accusative recognition	53.12	67.11	59.30	upper bound	46.21	88.16	60.63
				lower bound	26.90	51.32	35.29
				chance	32.41	61.84	42.53

Table 5: Bohnet-Parser and FSpar on 277-sentences gold standard

Nach-PV recognition has a recall of 100.0 for FSPar, which is due to the facts that (i) we used the parser to identify sentences containing at least one *nach*-PV, so FSPar introduced all sentences including the sentences which contain no *nach*-PV and (ii) FSPar makes use of a huge lexicon so the correct lemma does not have to be generated. Due to the underspecified output of FSPar, three values are given for its argument evaluations. The first value results from a credulous interpretation of the underspecified output, constituting an *upper bound*, which takes all cases into account, where the right annotation could be derived from the underspecified one. The second value results from a strict interpretation and introduces a *lower bound*, as only those cases have been counted as correct where the right annotation was the only annotation by FSPar. The third value, called *chance* in the table, was calculated by randomly choosing an annotation from those proposed by FSPar.

4.2 More reliability by voting-based combination

To extract reading indicators from the large data set, we make use of our findings on the test set. The quality of the accusative and dative argument recognition is seen as an approximation of the overall parsing quality relative to the task of indicator identification. To extract more reliable indicators, we combine the results of FSPar and the Bohnet-Parser. By this approach, we trust in the hypothesis that the combined result exceeds the best single result, which has been confirmed for a set of speech recognition systems by Fiscus (1997), for a set of constituency-based parsers by Henderson and Brill (1999) and for a set of dependency parsers by Zeman and Žabokrtský (2005), among others. As our combination is task specific, we do not make use of complex heuristics or approaches handling the complete parse, but define some extraction rules based on the evaluation results from Section 4.1.

Tables 6 show the combination rules which we applied in the indicator extraction. Due to the underspecification representation of FSPar the combination rules distinguish between the two binary features "*nach*-PV has an argument of case X", where X is in {DAT,ACC}, and the feature denoting the argument form. Tables 6a and b show the combination rules for the binary features. $\oplus$DAT and $\oplus$ACC denote that the parser identified an argument in the respective case. For FSPar, this is split up in cases where the output of FSPar is not underspecified, i.e. where there was only a single result (*s*) and cases where the respecive annotation was one possibility

in a set of multiple annotations (*m*). ⊖DAT and ⊖ACC denote that the parser did not identify an argument of the respective case. In cases where both parsers agree, no further rule has to be applied. Where they disagree, we decided for the following rules depending on the results of the evaluation from Section 4.1:

a.

FP \ BP	⊕DAT	⊖DAT
⊕DAT **s**	⊕DAT	⊕DAT
m	⊕DAT	⊕DAT
⊖DAT	⊕DAT	⊖DAT

b.

FP \ BP	⊕ACC	⊖ACC
⊕ACC **s**	⊕ACC	⊖ACC
m	⊕ACC	⊕ACC
⊖ACC	⊖ACC	⊖ACC

c.

FP \ BP	⊕DAT	⊕ACC	0
⊕DAT **s**	⊕DAT	⊕DAT	⊕DAT
⊖ACC	⊕DAT	⊕DAT	0
⊕ACC **s**	⊕ACC	⊕ACC	⊕ACC
⊖DAT	⊕DAT	⊕ACC	0
⊕DAT ⊕ACC	⊕DAT	⊕ACC	0
0	⊕DAT	⊕ACC	0

Table 6: Combination rules for a. datives and b. accusatives per annotation and c. per argument

As the precision of Bohnet-Parser's dative recognition is high (87.18), we take the result of the Bohnet-Parser, whenever it identifies a dative. Regarding recall however, FSPar's result exedes the Bohnet-Parser even in the lower bound and has a very high value for the upper bound (91.67) which is an important value for the binary feature. Therefore we only decide for ⊖DAT in cases where both parsers agree on that.

For arguments in accusative case, the only high value in the evaluation was the upper bound recall of FSPar (88.16). This is not surprising, as arguments in accusative case are highly case syncretistic in German. We decided thus to let FSPar overwrite the decision of the Bohnet-Parser in the cases where an argument in accusative case was among the results of FSPar or where FSPar did not recognize an argument in accusative case at all. In cases where the Bohnet-Parser did not identify an argument in accusative case, but FSPar did so explicitly, i.e. the single case, we do not let FSPar overwrite because the values of FSPar's lower bound, which take exactly those single cases into account, are very low ($f1 = 35.29$).

Table 6 shows the rules we utilized in extracting the argument form. There we have the cases that the Bohnet-Parser specifies a form to be an argument in dative case (⊕DAT), in accusative case (⊕ACC), or neither (0). FSPar identifies a form as single dative (⊕DAT, *s*), as underspecified dative but from a representation of which no accusative can be derived (⊕DAT, ⊖ACC), as single accusative (⊕ACC, *s*), as underspecified accusative but no dative (⊕ACC, ⊖DAT), as underspecified between at least dative and accusative (⊕DAT, ⊕ACC), or as none of these (0).

Again we trust the dative recognition of each parser, which is only overwritten in two cases: (i) FSPar rules out a dative in favor of a single accusative and (ii) Bohnet-Parser neither annotates accusative nor dative and FSPar states the dative in a multiple result. These decisions deviate from the ones in the binary features. The latter decision takes into account that now a particular argument has to be decided upon and identifiying the respective dative argument drops FSPar to the chance value. The first decision is a more debatable one, and serves to balance the decisions for accusative arguments which are overall more frequent. The same applies for the other cases, in which FSPar proposes an accusative, while the Bohnet-Parser does not. If FSPar proposes ⊕DAT and ⊕ACC, or neither of them, we opt for the Bohnet-Parser. For all other clustering features, we use the Bohnet-Parser output as long as FSPar does not contradict it.

5 Automatic clustering of *nach*-PV lemmas into semantic classes

5.1 Experimental setup

As input data, we use the 270k sentences extracted from the SDeWaC corpus, which are automatically parsed by the Bohnet-Parser and FSPar. From these automatic syntactic structures we then extract five different types of features starting with the presence of dative and accusative arguments. In order to see the effect of features beyond argument structure, as described in Section 2.2, we successively add four other types of features: (i) the word form of the dative/accusative arguments, (ii) a combination of preposition form, case value, and prepositional object form for each prepositional dependent of the verb, (iii) the word form of adverbials depending on the verb, and (iv) the presence of clausal objects.

We use Ward's algorithm (Ward, 1963) to produce the clustering of the verbs in our gold standard with the number of output clusters set to 18.[6] In order to evaluate our clusters, we use the v-measure proposed in Rosenberg and Hirschberg (2007), which is defined as the harmonic mean of homogeneity and completeness. Both metrics are defined based on entropy of the clusters, where homogeneity measures the distribution of gold standard classes within each cluster and completeness measures the distribution of clusters within each gold standard class.[7]

features (added up)	homogeneity		completeness		v-measure	
	BP	FSPar	BP	FSPar	BP	FSPar
	combined		combined		combined	
dat,acc	32.96	37.62	28.20	30.11	30.39	33.45
	38.23		30.23		33.76	
⊕ datform,accform	33.24	33.53	30.27	28.19	31.68	30.62
	36.04		29.82		32.46	
⊕ pp-form-case-pobjform	34.40	35.18	30.65	29.56	32.42	32.13
	36.08		31.32		33.53	
⊕ adverbials	35.78	37.77	33.56	32.11	34.64	34.71
	41.76		34.89		38.02	
⊕ clausal objects	39.56	39.31	35.18	33.30	37.24	36.05
	41.49		35.09		38.02	

Table 7: Clustering: features extracted by Bohnet-Parser, FSPar, and their combination

5.2 Results and evaluation against human gold standard

Table 7 shows the result of the clustering broken down for the individual parsers as well as their combination. Focusing on the results for the Bohnet-Parser, we see that all features successively add to the overall performance. This shows that the additional features also contribute information to the formation of verb classes. If we check the results for FSPar and the combined feature extraction, we see a drop in performance when the word forms of the dative and accusative arguments are added, which is then compensated by the other three features which improve performance. The highest value is achieved if the combined feature

[6] Verbs can be in more than one of the theoretically predicted classes. Since the clustering algorithm does not allow for an instance to be in more than one cluster, we assume each combination of theoretical readings to be one class in the clustering. We however only consider those that appear in the gold standard and we also remove all verbs that belong to neither of the five *nach*-PV classes.

[7] The ideal case for homogeneity occurs when there is only one single class in each cluster whereas for completeness, the ideal case occurs when there is only one cluster for each of the gold standard classes.

extraction is applied. The reason for the drop for the first features is the ambiguous output of FSPar, that possibly adds more word forms to the features for the clustering than are actually correct. This is due to the fact that oftentimes FSPar gives more than one possibility for the dative or accusative argument and it is not possible to automatically choose between them without first disambiguating them. In the combined system, the drop is much smaller than for FSPar alone, because we can use the Bohnet-Parser to restrict the options that FSPar offers.

nach-reading	frequency	homogeneity	completeness	v-measure
CONT	10	78.53	66.30	71.90
{MAN,CRE}	24	42.82	58.45	49.43
OMR	51	45.35	44.82	45.08
MAN	16	28.02	56.85	37.54
DIR	51	37.05	37.19	37.12
{MAN,CRE,CONT}	1	65.95	100.00	79.48
{DIR,MAN,CRE,CONT}	1	44.05	100.00	61.16
{DIR,OMR,CONT}	2	45.17	86.83	59.43
{DIR,MAN}	4	44.32	78.85	56.75
CRE	1	39.61	100.00	56.75
{DIR,CRE}	2	40.59	86.83	55.32
{DIR,MAN,CRE}	5	44.53	73.28	55.40
{DIR,CONT}	3	38.95	79.12	52.20
{DIR,OMR}	3	36.65	79.12	50.09
{CRE,OMR}	3	33.17	86.74	47.99
{OMR,CONT}	5	31.96	79.36	45.56
{MAN,OMR}	7	31.72	72.20	44.08
{MAN,CRE,OMR}	4	25.58	73.66	37.98
overall	193	41.49	35.09	38.02

Table 8: Clustering: detail analysis: Bohnet-FSPar combination and all features

The other finding from the evaluation is the rather low overall performance. With a best value of 38.02 % v-measure it seems that we simply cannot recreate the classes that our theory predicts. However, it is worth taking a closer look at the results for the individual classes, since it turns out that the clustering quality varies greatly with the class that we are trying to produce. Table 8 shows the results of the clustering using the combined system, broken down for the individual classes. The table also splits the classes into high frequency and low frequency classes, showing the bigger classes first. As can be seen from the results, there are five classes that contain at least 10 lemmata, the biggest of them containing 51. It turns out that for three of them, we get much higher results than expectable from the average score (up to 35 % better for the CONT class). For the second and third class, we get results a little below 50 %, but for the MAN and DIR class, we get results slightly below the average. For the smaller classes with size less than 10, we mostly get results higher than the average, but these classes are too small to be really conclusive. In summary, we find that while the clustering works reasonably well for three of the bigger classes, the results are unsatisfactory for the other two. An explanation for the inferior performance on the MAN class can be found in the theory: as we discussed already in Section 2.1.3, the classes MAN and CRE are very closely related and could be considered one class since it is often not easy to distinguish the two meanings. The clustering seems to have similar problem singling them out. The performance on the DIR class can to a certain extent be explained by the argument structure of these verbs. As we show in Table 1, a DIR verb can have an accusative argument even though it would normally be avoided. That means that some of the features

that are available to the clustering are less informative than for other classes. As a point in case consider the CONT class, which comes out very nicely in the clustering. As shown in Table 1, this class has the most distinct argument distribution compared to all other classes.

One should also keep in mind that we are working with a tool chain of automatic tools that itself has several drawbacks, which influence the performance of the clustering. This includes the quality of the parsers, which although being state-of-the-art are still far from being perfect, and also the clustering algorithm, which can make incorrect decisions. Finally, one needs to take into account that our gold standard is not optimal because of the difficulties annotating our five verb classes.

Results and interpretation of the experiments: lessons learned

Theoretical results. The *nach*-PV class we were able to identify most precisely is the continuation class (CONT) for which no dative or accusative is predicted, cf. also Table 1. As assumed in Section 2.1.3, we saw that the theoretically motivated *nach*-PV classes MAN and CRE empirically collapse. We can conclude that argument structure, among others, is a fairly good indicator for automatically identifying the combined class {MAN,CRE}. However for the individual classes MAN and CRE, as well as for the directional class DIR, this is not the case. Here, more research is needed to pin down clear criteria for the selection of these classes.

Technological results. Regarding the NLP-methodology, our findings mainly address three topics: the quality of the single parsers, the combination of the parsing results and the application of a small high qualitiy sample as an approximation for the complete data set. We started with two single parsers, each evaluated against a small gold standard for *nach*-PV argument structures. Both parsers fell below expectations. The results of the state-of-the-art Bohnet-Parser decreased due to the unrestricted all-domain data. And FSPar, which preserves underspecified structures, did not reach an f-score higher than 73.33 even in the upper bound. While already the identification of *nach*-PVs was difficult due to sentences containing no *nach*-PV and due to complex lemmas, the creation of the gold standard also showed a lot of valency-changing constructions, most of them being difficult to annotate for the parsers. As we expect the gold standard to be representative for the whole data set, these difficult constructions should be found there in a similar distribution. This definitely has an impact on the features used in the clustering and therefore on the clustering results. So even by applying the best tools available their performace leaves much room for improvement. Nevertheless the parser combination showed the expected effect as the v-measure for the features from parser combination always exceeds the best single result. This also supports the applicabillity of the gold standard as approximation of the data set, as the combination rules were based on it.

Outlook. To benefit from this findings, we intend to add more parsers to the result combination and for example apply a majority voting scheme. While the two parsers we utilized were applicable right away, it might seem a good idea to complement our parsers with parsers that can be adapted to the task. Furthermore, one could try to apply other clustering techniques, e.g. fuzzy clustering to give greater emphasis on the fact that one lemma can be found in more than one class. Concerning the low agreement on the *nach*-PV lemma classification, we think that one could improve this by either taking more annotators into account or by classifying the *nach*-PV lemmas in expected and unexpected (coerced) contexts for each reading, and then measure their acceptability. This would rather approximate the conjectured continuum-like character of the distribution of the *nach* readings over the verb lemmas.

Acknowledgments

This work was funded by the Deutsche Forschungsgemeinschaft (DFG) via Projects B3, B4 and D8 of the SFB 732 "Incremental Specification in Context".

References

Abney, S. (1996). Partial parsing via finite-state cascades. *Natural Language Engineering*, 2(4):337–344.

Baroni, M. and Kilgarriff, A. (2006). Large linguistically-processed web corpora for multiple languages. In *Proceedings of the 11th Conference of the European Chapter of the Association for Computational Linguistics (EACL) 2006*, pages 87–90, Trento.

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

Brants, S., Dipper, S., Hansen, S., Lezius, W., and Smith, G. (2002). The TIGER Treebank. In *Proceedings of the Workshop on Treebanks and Linguistic Theories*, Sozopol.

Dowty, D. (1991). Thematic proto-roles and argument selection. *Language*, 67(3):547–619.

Eckart, K., Eberle, K., and Heid, U. (2010). An infrastructure for more reliable corpus analysis. In Calzolari, N., Choukri, K., Maegaard, B., Mariani, J., Odijk, J., Piperidis, S., Rosner, M., and Tapias, D., editors, *Proceedings of the Workshop on Web Services and Processing Pipelines in HLT: Tool Evaluation, LR Production and Validation (LREC 2010)*, pages 8–14, Valletta. European Language Resources Association (ELRA).

Fillmore, C. J., Johnson, C. R., and Petruck, Miriam R. L. (2003). Background to FrameNet. *International Journal of Lexicography*, 16(3):235–250.

Fiscus, J. G. (1997). A post-processing system to yield reduced word error rates: Recognizer output voting error reduction (ROVER). In *Proceedings of the IEEE Workshop on Automatic Speech Recognition and Understanding*, pages 347–354.

Halle, M. and Marantz, A. (1993). Distributed Morphology and the pieces of inflection. In Hale, K. and Keyser, S. J., editors, *The View from Building 20: Essays in Linguistics in Honor of Sylvain Bromberger*, volume 24 of *Current Studies in Linguistics*, pages 111–176. MIT Press, Cambridge, MA.

Haselbach, B. (2011). Deconstructing the German verb particle *nach* at the syntax-semantics interface. In Baunaz, L., Bentea, A., and Blochowiak, J., editors, *Generative Grammar in Geneva (GG@G) 7*, pages 71–92. Department of Linguistics, University of Geneva, Geneva.

Henderson, J. C. and Brill, E. (1999). Exploiting diversity in natural language processing: Combining parsers. In *Proceedings of the Fourth Conference on Empirical Methods in Natural Language Processing (EMNLP-99)*, pages 187–194, College Park, Maryland.

Kamp, H. and Reyle, U. (1993). *From Discourse to Logic. Introduction to Modeltheoretic Semantics of Natural Language, Formal Logic and Discourse Representation Theory*, volume 42 of *Studies in Linguistics and Philosophy*. Kluwer Academic Publishers, Dordrecht/Boston/London.

Landis, J. R. and Koch, G. G. (1977). The measurement of observer agreement for categorical data. *Biometrics*, 33(1):159–174.

Larson, R. K. (1990). Double object revisited: Reply to Jackendoff. *Linguistic Inquiry*, 21(4):589–632.

McDonald, R. T. and Pereira, F. C. N. (2006). Online learning of approximate dependency parsing algorithms. In *EACL 2006, 11st Conference of the European Chapter of the Association for Computational Linguistics, Proceedings of the Conference, April 3-7, 2006, Trento, Italy*. The Association for Computer Linguistics.

Nicol, F. (2002). Extended VP-shells and the verb-particle construction. In Dehé, N., McIntyre, A., Jackendoff, R., and Urban, S., editors, *Verb-Particle Explorations*, volume 1 of *Interface Explorations*, pages 165–190. Mouton de Gruyter, Berlin.

Rosenberg, A. and Hirschberg, J. (2007). V-Measure : A conditional entropy-based external cluster evaluation measure. In *Empirical Methods of Natural Language Processing (EMNLP) 2007*, number June, pages 410–420, Prague, Czech Republic. Association for Computational Linguistics.

Roßdeutscher, A. and Kamp, H. (2010). Syntactic and semantic constraints on the formation and interpretation of *-ung*-nouns. In Rathert, M. and Alexiadou, A., editors, *The Seamntics of Nominalizations across Languages and Frameworks*, volume 22 of *Interface Explorations*, pages 169–214. Mouton de Gruyter, Berlin.

Schiehlen, M. (2003). A cascaded finite-state parser for German. In *Proceedings of the 10th Conference of the European Chapter of the Association for Computational Linguistics (EACL) 2003*, pages 163–166, Budapest.

Schmid, H. (1994). Probabilistic part-of-speech tagging using decision trees. In *International Conference on New Methods in Language Processing*, pages 44–49, Manchester.

Seeker, W. and Kuhn, J. (2012). Making ellipses explicit in dependency conversion for a german treebank. In Chair), N. C. C., Choukri, K., Declerck, T., Doğan, M. U., Maegaard, B., Mariani, J., Odijk, J., and Piperidis, S., editors, *Proceedings of the Eight International Conference on Language Resources and Evaluation (LREC'12)*, Istanbul, Turkey. European Language Resources Association (ELRA).

Springorum, S., Schulte im Walde, S., and Roßdeutscher, A. (2012). Automatic classification of German *an* particle verbs. In *Proceedings of the 8th International Conference on Language Resources and Evaluation*, pages 73–80, Istanbul, Turkey.

Ward, J. H. (1963). Hierarchical grouping to optimize an objective function. *Journal of the American statistical association*, 58(301):236–244.

Zeman, D. and Žabokrtský, Z. (2005). Improving parsing accuracy by combining diverse dependency parsers. In *Proceedings of the Ninth International Workshop on Parsing Technology*, pages 171–178, Vancouver, British Columbia. Association for Computational Linguistics.

Bridging the Gap between Intrinsic and Perceived Relevance in Snippet Generation

Jing He *PabloDuboue* *Jian-YunNie*
DIRO, University of Montreal
hejing@iro.montreal.ca, dubouep@iro.umontreal.ca, nie@iro.umontreal.ca

ABSTRACT
Snippet generation plays an important role in a search engine. Good snippets provide users a good indication on the main content of a search result related to the query and on whether one can find relevant information in it. Previous studies on snippet generation focused on selecting sentences that are related to the query and to the document. However, resulting snippet may look highly relevant while the document itself is not. A missing factor that has not been considered is the consistency between the perceived relevance by the user in reading the snippet and the intrinsic relevance of the document. This factor is important to avoid generating a seemingly relevant snippet for an irrelevant document and vice versa. In this paper, we incorporate this factor in a snippet generation method that imposes the constraint that the snippet of a more relevant document should also be more relevant to the query. We derive a set of pairwise preferences between sentences from relevance judgments. We then use this set to train a gradient boosting decision tree to model a sentence scoring function used in snippet generation. Compared to the existing snippet generation methods and to the snippets generated by a commercial search engine, our snippets are more consistent with the true relevance of the documents. When the snippets are incorporated into a document ranking function, we also observe a significant improvement in retrieval effectiveness. This study shows the importance to generate snippets indicating the right level of relevance to the search results.

KEYWORDS: Search Engine Snippet, Query-biased Summarization, Document Retrieval.

Proceedings of COLING 2012: Technical Papers, pages 1129–1146,
COLING 2012, Mumbai, December 2012.

1 Introduction

The quality of a search engine is not only determined by its document ranking function, but also by the way the search results presented to the user (Turpin et al., 2009). In particular, good snippets can help the user select relevant documents to click on. Snippets can be considered as a type of extractive summary of the Web pages. It is thus important that a snippet reflects the main part of the document content related to the query topic. This is the traditional role that one assigns to a snippet. To this end, two main categories of approaches have been proposed for snippet generation: sentence retrieval (Sanderson, 1998; Han et al., 2000; Ko et al., 2007; Park and An, 2010; Daumé and Marcu, 2006) and query-biased document summarization (Conroy et al., 2006; Tombros and Sanderson, 1998; Varadarajan and Hristidis, 2005; Otterbacher et al., 2005; Wang et al., 2007; Kanungo et al., 2009).

The sentence retrieval approach finds the most relevant sentences (or text fragments) from a document and uses them as the snippet. One drawback of this method is that it does not consider the fidelity of the sentence to the content of the document, i.e., the extracted sentences may not reflect the main content of the document. There is a high risk of producing false positive snippets leading the user to click on an irrelevant document.

To solve this problem, query-biased document summarization aims to summarize the main content of a document for a specific query. The methods in this category consider both the query and the document. However, an implicit assumption in this method is that the document to be summarized is relevant to the given query. As a matter of fact, the training data used to train the extraction method usually consists of queries and relevant documents. The resulting method could reflect well the relevant part of a relevant document, but may also tend to generate "relevant" snippets for irrelevant documents.

The missing factor in the previous approaches to snippet generation is the consistency between the relevance that the user perceives in reading the snippet (perceived relevance) and the real relevance of the document (intrinsic relevance). The higher-than-intrinsic perceived relevance of snippets leads users to click on irrelevant documents (false positive), while the reverse case leads to not clicking on relevant documents (false negative).

In this paper, we propose a snippet generation method that tries to produce a snippet reflecting the right level of intrinsic relevance of the document. The basic idea is to define a sentence ranking function for a query and a document in such a way that the snippet of a document with a higher intrinsic relevance has a higher perceived relevance. We cast the sentence ranking problem in a learning-to-rank framework (Liu, 2011) with the above constraint. A set of training data is derived from a TREC dataset in which the intrinsic relevance of documents is known. The perceived relevance of a snippet is approximated by a similarity score to the query. A set of features will be defined to reflect different aspects such as the sentence's fidelity to the document and relevance to the query. Finally, we define a cost function for the learning, and propose to use a gradient boosting decision tree model to minimize the cost of the training data set.

When a snippet can reflect the intrinsic relevance of a document, it could be useful to use it to help document ranking. Although some previous studies have examined this by replacing the documents by their snippets for the purpose of increasing efficiency (Sakai and Sparck-Jones, 2001; Wasson, 2002), the results are not conclusive as to whether combining snippets with the

document content can help document ranking. In this paper, we will also incorporate snippets into the document ranking function.

We evaluate the snippet generation methods by comparing the manually judged perceived relevance of the generated snippets to the intrinsic relevance of the corresponding documents, and find that our snippets can better reflect the intrinsic relevance. On F1 score, it improves 14.0% compared to a search engine's snippets, and 7.7% compared to the query-biased summaries. The experiment on document retrieval also shows that our snippets, when incorporated into the document function, are useful in boosting document retrieval.

The remainder of the paper is organized as follows: we first present related work in Section 2. Our snippet generation method is described in Section 3. The experiments are shown in Section 4 and we conclude the paper in Section 5.

2 Related Work

A snippet is usually used as a surrogate for the content of the document in the search engine, and it helps the user determine whether the document contains relevant information (Tombros and Sanderson, 1998; White et al., 2002, 2003).

2.1 Snippet Generation Methods

There are two main categories of methods for generating snippets: sentence retrieval and query-biased summarization.

The sentence retrieval approach finds the most relevant sentences from a document using an IR method, and takes them as a snippet (Sanderson, 1998; Han et al., 2000; Ko et al., 2007; Park and An, 2010; Daumé and Marcu, 2006). One drawback of the approach is that it does not consider the fidelity of sentences to documents, i.e., their relation with the main content of the document. As a consequence, a snippet of an irrelevant document may consist of marginal sentences from the document and appear highly relevant. This problem can be partly tackled by the query-biased summarization method described below.

Query-biased document summarization approaches aim to summarize the content of a document around a specific query. In these approaches, both the document and query are considered in the snippet generation. (Conroy et al., 2006) used query terms and significant terms to select the sentences as snippets. (Tombros and Sanderson, 1998) also considered features such as a sentence's position (title, opening sentence, head, etc.) in weighting. Some work models a document as a graph, in which a sentence is presented as a vertex and the relation between the sentences is presented as a weighted edge. The goal is then to select a minimal spanning tree (Varadarajan and Hristidis, 2005) or a set of central sentences using PageRank-like algorithm as a snippet (Otterbacher et al., 2005).

The query-biased summarization problem can also be thought as a problem of ranking sentences according to their goodness to be a summary, so the learning-to-rank methods such as SVM classifier, SVM-rank (Wang et al., 2007), GBDT (Metzler and Kanungo, 2008) and GBRank (Kanungo et al., 2009) can be used. To build the training data for these methods, human subjects are asked to generate the manual snippet for documents or to judge the snippet's goodness. However, a "good" snippet in this case can reflect the content of a document, but may fail to help users distinguish relevant and irrelevant documents. In this paper, we will incorporate a criterion of consistency between the perceived relevance of the snippets and the intrinsic relevance of the documents in the learning-to-rank framework.

2.2 Snippet Evaluation

It is important to measure the quality of a snippet. The ultimate measure is to put snippet directly in a search task and evaluate how well they can help accomplish the task (Tombros and Sanderson, 1998; White et al., 2003). However, this method is very expensive and not reusable, and the utility measure is influenced by both the retrieval performance and the quality of snippets. Most work targeting the construction of indicative summaries relies on task-based evaluation where the summaries are evaluated as surrogates of the document content (Murray et al., 2009; Kushniruk et al., 2002). Some researchers also proposed to evaluate these two components together (Turpin et al., 2009; He et al., 2010).

Since the snippet generation problem is usually cast as a query-biased summarization problem, evaluation methods in summarization are also used for snippets. In this method, it is assumed that there is a gold-standard summary, and the automatically generated snippet can be compared to it (Wang et al., 2007; Bando et al., 2010). However, (Bando et al., 2010) found this method usually overestimates the quality of the snippets. Alternatively, some work evaluated a snippet by its judged goodness (Kanungo et al., 2009) or readability (Kanungo and Orr, 2009). Users' interaction with the search engine such as clickthrough, dwelling time and eye tracking are also used to evaluate the quality of the snippets (Savenkov et al., 2011; Cutrell and Guan, 2007).

For the task of search, we argue that the quality of a snippet should be primarily evaluated by whether the *perceived relevance* from it can reflect the *intrinsic relevance* of the corresponding document. In SUMMAC evaluation, (Mani et al., 1999) compared the relevance judgments on snippets and on documents, and found the users can make reasonable relevance based on snippets. In the INEX 2011 snippet retrieval task, a measure was used to explicitly examine the consistency between the perceived relevance of the snippets and the intrinsic relevance of the documents (Trappett et al., 2012). However, this measure was only used in the evaluation and none of the participating system used it in the snippet generation process. We believe that we are among the first researchers to use this criterion to train a snippet generator.

2.3 Using Snippets for Retrieval

Since a summary of a document can be considered as a surrogate for the document, it can be used in the document retrieval. Some previous work (Sakai and Sparck-Jones, 2001; Wasson, 2002) tested the utility of snippets for retrieval by replacing the document content by a summary, finding that it can achieve similar precision but worse recall. In this paper, we also test the use of snippets by applying them during the document retrieval. Instead of replacing the document content, we combine the snippet with the document content. This will lead to a higher retrieval effectiveness.

3 Generating Informative Snippets

In this section, we present the method for generating snippets whose relevance is consistent to the documents' relevance. We will first define the problem, and then propose to address this problem in a learning-to-rank framework.

3.1 Problem Definition

(Rose et al., 2007) found that text choppiness and sentence truncation is not good for the readability of search results, and it is better to use complete sentences in snippets. Following

this observation, we define snippet generation as a process of extracting a subset of complete sentences from a document.

A good snippet should of course reflect the main content of the document related to the query topic. It should also provide a good indication on how the document could be relevant, or how two documents compare with respect to relevance. In other words, the comparison between the perceived relevance of two documents should be consistent with their intrinsic relevance. Let's use $R_d(q,d)$ and $R_s(q,S)$ to denote respectively the intrinsic relevance of a document d to a query q and the perceived relevance of the document's snippet S to the query q. The consistency can be defined as follows: Given a query q and a document pair (d_1,d_2) with intrinsic relevance $R_d(q,d_1) > R_d(q,d_2)$, a snippet pair (S_1,S_2) is consistent with the relevance of the documents if and only if $R_s(q,S_1) > R_s(q,S_2)$.

Compared to previous approaches, our additional problem is to find a snippet generation function g^* so that the snippet pairs generated are as consistent with the relevance of the documents as possible. Formally, we need to find a snippet generator g^* that can maximize the expected ratio of the snippet pairs with consistent perceived relevance:

$$g^* = \arg\max_g \{\mathrm{E}_{q,d_1,d_2}[R_s(q,g(q,d_1) > R_s(q,g(q,d_2))|R_d(q,d_1) > R_d(q,d_2)]\} \quad (1)$$

In this paper, a set of judged queries will be used to train g^*. That is, for the training queries, we have document relevance judgments $R_d(q,d)$. The perceived relevance of a snippet $R_s(q,g(q,d))$ will be approximated by a similarity score between the query and the snippet. Even though we used the similarity between queries and snippets to approximate the perceived relevance, the same formalism can use real perceived relevance assessed by human subjects, an approach we might pursue in future work.

We use the learning-to-rank framework to address this problem due to its capability to utilize many different features. The previous work on snippet generation showed that pairwise learning methods perform better than pointwise methods (Kanungo et al., 2009; Wang et al., 2007), and that the methods based one gradient boosting decision tree (GBDT) outperform SVM based methods (Metzler and Kanungo, 2008). In this work, we use GBDT to generate the snippets.

3.2 Gradient Boosting Decision Tree (GBDT)

GBDT is a state-of-the-art learning-to-rank method, which can be learned in a pointwise or pairwise manner (Zheng et al., 2007; Friedman, 2001, 2002; Li et al., 2007; Kanungo et al., 2009; Metzler and Kanungo, 2008).

For pointwise learning, the training data containing N samples can be presented as $\{(x_i,y_i)|x_i \in \mathbb{X}, y_i \in \mathbb{Y}\}_{i=1}^N$, where x_i is a set of extracted features of an item, and y_i is the value of the independent variable, e.g., relevance of a document in the information retrieval task. A ranking function can produce a score for a feature vector $f : \mathbb{X} \to \mathbb{Y}$. Given a loss function $L(y,f(x))$ (we use least-squares loss) defined on the predicted value $f(x)$ and the real value y, the goal is to learn a function f^* in a function class $\mathbb{F}$ that can minimize the sum of loss function on the training dataset, i.e.,

$$f^* = \arg\min_{f\in\mathbb{F}} \sum_{i=1}^N L(y_i,f(x_i)) = \arg\min_{f\in\mathbb{F}} \sum_{i=1}^N \frac{1}{2}(y_i - f(x_i))^2$$

It is an optimization problem, and it can be solved using gradient descent. At iteration k the function $f^{(k)}$ can be updated by $f^{(k+1)}(x) = f^{(k)}(x) - \alpha_k \nabla L(f^{(k)}(x))$, where $\nabla L(f^{(k)}(x))$ is the gradient of $f^{(k)}$ at the point x, and α_k is a coefficient that can be set by line search or at a predefined value.

The gradient descent is implemented in GBDT as follows: At iteration k, we calculate the negative gradient $y_i - f^{(k)}(x_i)$ for each point in the training data, and these points as well as their negative gradients form a new gradient training data $\{x_i, y_i - f^{(k)}(x_i)\}_{i=1}^N$. We can fit a regression decision tree for this gradient training data. For a new point x, we can predict its gradient by the decision tree.

When GBDT is used in pairwise learning (Zheng et al., 2007), the training data is a set of ordered paired items: $\{(x_{i1}, x_{i2}) | y_{i1} > y_{i2}\}_{i=1}^N$. It is expected to produce the predicted values whose pairwise ranking is consistent with that in the training data. We can define a least-squares loss function as follows

$$L(f) = \sum_{i=1}^{N} \frac{1}{2} (\max\{0, f(x_{i2}) - f(x_{i1}) + \sigma\})^2, \sigma > 0 \tag{2}$$

In this formula, the loss is zero when $f(x_{i1}) \geq f(x_{i2}) + \sigma$. A small positive value σ is used to prevent from learning a constant function which produces an identical value for all the points.

Similarly, we can use gradient descent method to solve the problem of optimizing $L(f)$ in this pairwise setting. The negative gradient of one pair of items in the training data can be calculated as:

$$-\nabla L(f^{(k)}(x_{i1}), f^{(k)}(x_{i2})) = \begin{pmatrix} \max\{0, f^{(k)}(x_{i2}) - f^{(k)}(x_{i1}) + \sigma\} \\ \max\{0, f^{(k)}(x_{i1}) - f^{(k)}(x_{i2}) - \sigma\} \end{pmatrix} \tag{3}$$

That is, for one sample of pair of items, we add two samples in the gradient training data $(x_{i1}, \max\{0, (f^{(k)}(x_{i2}) - f^{(k)}(x_{i1}) + \sigma)\})$ and $(x_{i2}, \max\{0, (f^{(k)}(x_{i1}) - f^{(k)}(x_{i2}) - \sigma)\})$, aiming to increase $f^{(k+1)}(x_{i1})$ if $(f^{(k)}(x_{i1}), f^{(k)}(x_{i2}))$ is inconsistent with (y_{i1}, y_{i2}).

3.3 Learning GBDT for Ranking Sentences

In this section, we will propose a method to rank the sentences using the GBDT model.

3.3.1 Training Data

We need training data to learn an informative snippet generator, i.e., $R_d(q, d)$ for a set of queries $\{q\}$ and a set of documents $\{d\}$. Ideally, we would also like to have $R_s(q, S)$ for a snippet S, but this is usually unavailable given the large number of possible snippets for a document. In our implementation, we approximate the perceived relevance by the query-sentence similarity.

3.3.2 Loss Function

Since our objective function is defined on preferences, we can transform the relevance judgments in the training data to a preference data set: $\{(q, d_1, d_2) | q \in Q, d_1 \in D_q, d_2 \in D_q, R_d(q, d_1) > R_d(q, d_2)\}$, where D_q is the set of retrieved documents for q. For a snippet generation function g, we can get a pair of snippets for a given sample of the training data:

$(g(q,d_1),g(q,d_2))$. The perceived relevance is $(R_s(q,g(q,d_1)),R_s(q,g(q,d_2)))$. Similarly to the general GBDT pairwise loss function (Eq 2), we can define the loss function here as

$$L(g)=\sum_{q,d_1,d_2}\frac{1}{2}(\max\{0,R_s(q,g(q,d_2))-R_s(q,g(q,d_1))+\sigma\})^2,\sigma>0 \quad (4)$$

3.3.3 Sentence Scoring Function

It is difficult to train the snippet generator function g directly. Instead, we define a sentence scoring function, and then derive the snippet generator function from the sentence scoring function. A sentence scoring function can be defined as $f:\mathbb{X}\rightarrow\mathbb{R}$, where the input is a vector of extracted features about a sentence s in a document d for a query q, and the output is a predicted score for the sentence. Given the sentence scoring function, the snippet generator function g can be applied to a document d and a query q by: (1) ranking the sentences $s\in d$ by their scores, and (2) selecting the top-n sentences as the snippet up to some predetermined length. Thus, the problem of learning a snippet generation function g can be cast as learning a sentence scoring function f.

3.3.4 Learning Method

We use the GBDT method to learn a sentence scoring function in an iterative manner. At iteration k, we have the sentence scoring function $f^{(k)}$ and the derived snippet generator function $g^{(k)}$. In each iteration, there are two cases:

- Case 1: If the perceived relevance of the snippet of document d_1 is larger than that of d_2, the sentence ranking function should not be modified;
- Case 2: If the perceived relevance of the snippet about document d_2 is larger, then we should modify the sentence ranking function so that the snippet of d_1 becomes more relevant and that of d_2 less relevant;

In Case 2, the modification of sentence ranking function f is achieved by adding new sentence preference pairs into the training set. In particular, for document d_1, we should rank higher the sentences that are more relevant than the snippet of d_2:

$$S^{(k)+}_{q,d_1}=\{s|s\in d_1',R_s(q,s)>R_s(q,g^{(k)}(q,d_2))$$

In our implementation, d' is a subset of d_1 composed by the sentences that contain at least one non-stop query term. As such, we have a preference dataset about d_1:

$$\{(s_1,s_2)|s_1\in S^{(k)+}_{q,d_1},s_2\in g^{(k)}(q,d_1)\}$$

Therefore, the gradient training set for the sentence scoring function can be defined as:

$$(x_1,f(x_2)-f(x_1)+\sigma),(x_2,f(x_1)-f(x_2)-\sigma)$$

where x_1 and x_2 are feature vectors for s_1 and s_2 respectively. Similarly, for the sentences in d_2, we should create

$$\{(s_1,s_2)|s_1\in S^{(k)-}_{q,d_2},s_2\in g^{(k)}(q,d_2)\}$$

where

$$S_{q,d_2}^{(k)-} = \{s | s \in d_2', R_s(q,s) < R_s(q, g^{(k)}(q,d_1))$$

The above process for each iteration has a complexity of $O(|Q| \cdot |D|^2 \cdot |d|)$, where $|Q|$ is the number of queries, $|D|$ is the number of retrieved documents, and $|d|$ is the number of sentences in a document. This complexity is high. We use the more efficient stochastic GBDT method (Friedman, 2002). The algorithm is presented in Algorithm 1.

Input: Parameters: N, M, σ, α
Training Data: Query Set Q, Document Set $\{D_q | q \in Q\}$, Intrinsic Relevance Judgment $\{R_d(q,d) | q \in Q, d \in D_q\}$, Perceived Relevance Judgment $\{R_s(q,s) | q \in Q, s \in d, d \in D_q\}$.
Output: Sentence Scoring Function: f
begin
  Initial $f^{(1)}$ as a constant function
  for $k = 1, \ldots, N$ **do**
    GD = {}
    for $j = 1, \ldots, M$ **do**
      Sample a query $q \in Q$, a pair of documents $d_1, d_2 \in D_q, s.t. R_d(q,d_1) > R_d(q,d_2)$
      Generate snippets $g^{(k)}(q,d_1)$, $g^{(k)}(q,d_2)$
      if $R_s(q, g^{(k)}(q,d_1) > R_s(q, g^{(k)}(q,d_2)$ **then**
        Sample a sentence $s_{11} \in g^{(k)}(q,d_1)$ and a sentence $s_{12} \in d_1 - g^{(k)}(q,d_1)$
        GD.add($\{(x_{11}, 0), (x_{12}, 0)\}$)
        Sample a sentence $s_{21} \in g^{(k)}(q,d_2)$ and a sentence $s_{22} \in d_2 - g^{(k)}(q,d_2)$
        GD.add($\{(x_{21}, 0), (x_{22}, 0)\}$)
      else
        Sample a sentence $s_{11} \in S_{q,d_1}^{(k)+}$ and a sentence $s_{12} \in g^{(k)}(q,d_1)$
        GD.add($\{(x_{11}, f(x_{12}) - f(x_{11}) + \sigma), (x_{12}, f(x_{11}) - f(x_{12}) - \sigma)\}$)
        Sample a sentence $s_{21} \in S_{q,d_2}^{(k)-}$ and a sentence $s_{22} \in g^{(k)}(q,d_2)$
        GD.add($\{(x_{21}, f(x_{22}) - f(x_{21}) + \sigma), (x_{22}, f(x_{21}) - f(x_{22}) - \sigma)\}$)
      end
    end
    Fit a regression decision tree $t^{(k)}$ to the gradient training data GD
    Update the sentence scoring function $f^{(k+1)} = f^{(k)} + \alpha t^{(k)}$ and the corresponding snippet generator function $g^{(k+1)}$
  end
  $f = f^{(N+1)}$
end

Algorithm 1: Training GBDT Sentence Scoring Function

3.3.5 Features

We use four categories of features in the learning methods (see Table 1): Query-Sentence Relevance (QSR), Document-Sentence Fidelity Features (DSF), Sentence Informativeness Features (SI), Query-Document Relevance Features (QDR). The features about the relevance between the query and the document (QDR) are not commonly used in query-biased summarization.

Category	Description	Features
QSR	the relevance between the query and the sentence	- Cosine similarity of TFIDF vectors
DSF	the fidelity of the sentence's content to the document	- Cosine similarity of TFIDF vectors - Number of significant words - Is the sentence the title/heading?
SI	how much information contained in the sentence	- KL-divergence to the collection model - Averaged IDF of the words
QDR	the relevance between the document and the query	- Cosine similarity of TFIDF vectors - BM25 score

Table 1: Features

Since our goal is to generate snippets that can reflect the query-document relevance, these features can help in snippet generation.

4 Experiments

We carried out two experiments to test our snippet generation method. The first experiment compared the perceived relevance of the generated snippets with the intrinsic relevance of the documents. The second experiment use the generated snippets in document retrieval.

4.1 Experiment Setup

4.1.1 Dataset

We use the dataset of TREC Web track 2010 and 2011. There are 100 queries in the dataset (98 of them have relevance judgments), and these queries were extracted from search engine logs. For each query, we were given a short query string, a detailed description and a list of description about the subtopics. We use short queries in the study. The document collection is ClueWeb09B, which contains 428M documents. The dataset also contains relevance judgments. The relevance judgment for a (query, document) pair is in grades {0,1,2,3}, with larger values indicating higher relevance. In our experiment, we took the 50 queries of TREC 2011 for training the sentence ranking function, and the other 48 queries of TREC2010 as test. The information of the training dataset is shown in Table 2. The test dataset will be further pruned due to lack of some baseline results and selection for judgment, discussed in Section 4.1.2 and Section 4.1.3.

Statistics	Dataset		
	Training	Test	Pruned Test
#query	50	48	47
#document	13081	15849	754
#sentence/doc	64.2	66.4	51.1
#word/sentence	10.1	10.3	12.0

Table 2: Experimental Dataset

4.1.2 Tested Snippet Generating Methods

Two baselines are used in our experiment: a commercial search engine (SE) and a query-biased summarizer (SUM). The use of a commercial search engine is to see how other snippet generation methods compare to the current state-of-the-art search engines. In order to collect the snippets for a query-document pair, we send a search query containing the query string and the site of the document's URL. For example, for the query "horse hooves" and the document whose URL is "http://www.snopes.com/military/statue.asp", we construct a query "horse hooves site:www.snopes.com". We then look for the result whose URL is exactly same as the document's URL. The content of some Web pages may change, and search engine generates the snippet based on a document with different content. So we only keep the documents whose content is very similar (cosine value > 0.95) to that in ClueWeb09B dataset.

For many (query, document) pair, we cannot find the snippets generated by the search engine. For a fair comparison, we only use the (query, document) pairs whose snippets can be found in the search engine in our evaluation, and the statistics of this pruned test dataset is shown in Table 2. We also limited the length of generated snippets comparable to the search engine's snippets, the average snippet length of SE, SUM, our method is 153, 130, 142, respectively.

For query-biased snippets, we used the MEAD summarizer (Radev et al., 2004). MEAD clusters sentences and chooses sentences most similar to the centroid of the cluster, discarding sentences too similar to already picked sentences. We also can use an external query file as an extra feature for ranking. For our snippet generation method, we approximate the perceived relevance of the snippet by the query-sentence similarity. Specifically, we use cosine similarity between the TFIDF vectors of sentences and queries. There are four parameters in the learning method (see Algorithm 1): iteration number N, sample size M for each iteration, the predefined cost for identical predicted relevance σ and the shrinking parameter α. We tune these parameters by five-fold cross validation on the training dataset and get $(N = 200, M = 10,000, \sigma = 0.01, \alpha = 0.2)$.

4.1.3 Judgments

For earh query in the dataset, we randomly select up to 20 Web pages and generate the snippets for them (Some topics may have less than 20 documents).

We ask human assessors to judge the perceived relevance of these snippets. For each query, the assessors were given the query string, the description and a list of subtopics. For each snippet, the assessors were asked whether they thought the document behind the snippet can provide relevant information for the query. Each snippet is judged as "relevant" or "irrelevant". Every snippet and query pair was judged first by two assessors, and a third assessor was invited to judge if the previous two assessors disagree. The final judgment used for the evaluation is determined by voting. We found the agreement rate of the first two assessors quite high (86.7%), and the Cohen's kappa value is 0.699.

4.1.4 Measures

We use the evaluation measures used in the INEX 2011 snippet retrieval task (Trappett et al., 2012). Denote the number of true positive, false positive, true negative and false negative snippets for all the test queries are TP, FP, TN, FN respectively. Recall (R) and negative recall (NR) reflect the percent of the relevance/irrelevant documents that can be detected by their snippets. The geometric mean score (GM) of these values are used as the primary measure

in INEX 2011 snippet retrieval task. In addition, the precision (P), F-1 values (F1) and the accuracy (ACC) are used for the evaluation. All measured used are shown in Table 3

Measure	Definition
Precision	$P = \frac{TP}{TP+NP}$
Recall	$R = \frac{TP}{TP+FN}$
F1 Rcore	$F1 = \frac{R \cdot P}{R+P}$
Negative Recall	$NR = \frac{TN}{FP+TN}$
GM	$GM = \sqrt{R \cdot NR}$
Accuracy	$ACC = \frac{TP+TN}{TP+FP+TN+FN}$

Table 3: Definition of Measures

4.2 Results

The results are shown in Table 4. Each row corresponds to one measures, and each column one snippet generation method. For the our snippet generation methods, we also show in the parentheses the improvement rate compared to the baselines (SE and SUM). We found that the scores of our method are higher than both the search engine's snippets and the query-biased document summarization on most measures with an exception on recall. The query-biased summarization method also performs better than the search engine's snippet.

Measure	Snippet Generation Method		
	SE	SUM	Our Method
P	0.412	0.506	0.531(28.9%,4.9%)
R	0.605	0.539	0.596(-1.5%,10.6%)
F1	0.493	0.522	0.562(14.0%,7.7%)
ACC	0.617	0.693	0.714(15.7%,3.0%)
NR	0.622	0.764	0.782(25.7%,2.4%)
GM	0.613	0.642	0.676(10.3%,5.3%)

Table 4: Evaluation Results for the Snippet Generation Methods

For the snippets generated by the search engine, its recall value is the highest of the methods tested, but its precision score is the lowest. In other words, the snippet generator of search engines tends to select sentences with higher perceived relevance, and this way attract the users to click on the snippets. Specifically, 30.8% of the documents within the test collection are relevant, but 44.8% of the snippets from the search engine are assessed as relevant, compared to 33.0% and 34.5% from SUMM and our method, respectively.

For the snippets generated by SUM and our method, the negative recall value is much larger than that of the search engine. This means that those methods can better reflect irrelevant search restuls. Both our method and SUM methods perform worse than SE for the recall metric, but the our method's performance doesn't drop so much (-1.5% and not significant worse) compared to the others.

As an illustration, Table 5 shows two examples of generated snippets. The second column in the table is the labeled intrinsic/perceived relevance. SUM method generates a false-negative

snippet in the first example, and SE and SUM generate false positve snippets in the second example. In both cases, the snippets generated by our method can better indicate if the document is relevant.

Query	-	horse hooves (about horse hooves, care and disease)
Doc	1	clueweb09-en0009-75-38689 (horse hooves cleaning)
SE	1	Cleaning your horses hooves is easy ... Cleaning your horse's hooves regularly will give you a chance to monitor the health of the foot,check the shoes...
SUM	0	Put a halter on your horse and secure him in cross ties or tether him to a rail using a quick release knot. If your horse doesn't pick up his foot...
Our Method	1	How To Clean Your Horses Hooves. This procedure can be used on foals, along with tapping the hoofpick on the bottom of the hoof to prepare the young...
Query	-	discovery channel store (locations, information and produdcts of Discovery Channel stores)
Doc	0	clueweb09-en0011-53-07816 (about a company rebranding for a product by Discovery Channel)
SE	1	Discovery Channel Stores-Ultimate Science Tools 2005 ReBrand 100: merit Order of information below: Before Image, Rebrand Images, Summary...
SUM	1	Challenge-As Discovery Channel changed the face of science and nature programing, Discovery Channel Stores altered the landscape of educational...
Our Method	0	Result-Delivering profit margins that only proprietary products can, the line brands Discovery Stores as the destination for quality, vibrant...

Table 5: Two Examples of Snippets

We have used four categories of features in our experiment in Section 3.3.5. Here we examine the importance of these features. The importance of each category of feature can be evaluated by testing the loss after removing a category of features. If the loss increases much after removing a category of features, it indicates that these features are important for learning. The results are shown in Figure 1. It shows that these four categories of features are all useful for learning our snippet generator. In particular, the features that reflect the relevance between queries and documents (RQD) should be taken into account in generating snippets.

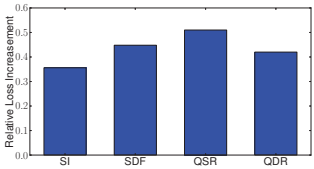


Figure 1: Feature Importance in Our Snippet Generating Method

4.3 Using Snippets for Document Retrieval

In this section, we investigate the role of snippets in document retrieval. We have found that the our snippets can help the users to determine the intrinsic relevance of the documents. Then a natural question is whether the snippets can also be useful to help a search engine determine the relevant documents.

In this experiment, we use a language modeling approach to IR, in which the document model is extended by the snippets as follows:

$$Pr(t|\theta_{d+S}) = \alpha Pr(t|\theta_{d_{\text{LM}}}) + \beta Pr(t|\theta_{S_{\text{LM}}}) + (1-\alpha-\beta)Pr(t|\theta_C)$$

where $\theta_{d_{\text{LM}}}$ and $\theta_{S_{\text{LM}}}$ are maximum likelihood model for the document and the snippet respectively, θ_C is the collection language model for smoothing, α and β are smoothing parameters. These parameters are tuned by grid search in our experiment. Given the new document language model, we use KL-divergence to score the document for a query:

$$KL(\theta_q||\theta_{d+S}) = \sum_t Pr(t|\theta_q) \log \frac{Pr(t|\theta_q)}{Pr(t|\theta_{d+S})}$$

The retrieval performance of the different snippet generation methods is shown in Table 6. The numbers in the parentheses show the improvement compared to retrieval based on the document model without combining with the snippet (NO). Since the snippets of only a subset of documents can be obtained from the search engine, we use the relevance judgments of this subset only in our evaluation. We also test the statistical significance using paired t-test, and † and ‡ indicate the difference is significant with p-value 0.1 and 0.05 respectively.

We can see that the snippets by SUM and our methods can significantly improve the retrieval effectiveness. This result clearly indicates that good snippets can be used to help document ranking. The improvement on top-ranked documents (ERR@10 and NDCG@10) is higher than MAP. This means that these methods have a higher impact on top retrieval results. On the other hand, we did not observe a significant difference when the snippets of the search engine are used. Overall, the rank of the retrieval performance with different snippet generation methods is the same as the perceived relevance accuracy: our method > SUM > SE.

However, we did not observe a significant difference between SUM and our method in IR, as in the experiment on perceived relevance. A possible reason is that both methods generate snippets that are related to the query and to the main content of the document, and the subtle difference is perceived by humans, but not by the simple retrieval model based on bag of words. It is possible that using a more sophisticated retrieval model, the difference between the two methods can be materialized on retrieval effectiveness. This is part of our future work.

Our experiment is different from that of (Sakai and Sparck-Jones, 2001; Wasson, 2002) in which snippets were used as replacement of documents. Their results showed that the retrieval recall decreases while precision remains in doing this. This is predictable as snippets are much shorter than documents, thus contain less less information. However, the short segment of text selected in a snippet is usually highly related to the query. They may provide useful position information about the document content, which can hardly emerge if we look at the whole content of the document. This may be the reason why snippets helped in our experiment on IR.

5 Conclusion and Future Work

In this paper, we addressed the problem of generating informative snippets for search results. A snippet is deemed informative not only because it reflects the part of content of the document related to the query topic, but also because it provides useful indication about the document's relevance. Generating a snippet that is perceived relevant for an irrelevant document is misleading, so is the opposite situation. We specifically addressed the problem of consistency

Measure	Snippet Generation Method			
	NO	SE	SUM	Our Method
MAP	0.328	0.328(0.0%)	0.335(2.1%†)	0.336(2.4%†)
ERR@10	0.109	0.112(2.8%)	0.114(4.6%†)	0.118(8.3%‡)
NDCG@10	0.290	0.293(1.0%)	0.304(4.8%‡)	0.309(6.5%‡)

Table 6: Retrieval Performance of Taking the Snippet as a Field

between the perceived relevance of the snippet and the intrinsic relevance of the document. To our knowledge, this is the first time that such a criterion is incorporated into a snippet generation process.

In this paper, we cast the snippet generation problem as the one to learn a sentence ranking function. The training data are pairwise sentence preference pairs, which are created according to the consistency between their perceived relevance and the real relevance of the documents. We used gradient boosting decision trees to model the sentence ranking function. Our experiments showed two facts: 1) Our method that generates snippets by considering the consistency criterion can provide better indication on the relevance of the documents to users; 2) The snippets of search results can provide useful information for document ranking.

We have explored one possible avenue to generate information snippets using a pairwise learning-to-rank method. Other methods could be explored in the future, such as SVM-rank, etc. Our focus in this paper was on consistency, and we used a simple method to determine the candidate sentences to be included into the training data. It may be better to select sentence candidates by also considering their fidelity to the document and relevance to the query.

On the use of snippets in document ranking, we used a bag-of-word approach, which failed to cope with the subtle differences between snippets. Those differences may emerge in the retrieval results if we use more sophisticated retrieval methods, e.g., by considering the possible relations between terms and by considering more complex units such as phrases. These are the problems that we will address in our future work.

References

Bando, L. L., Scholer, F., and Turpin, A. (2010). Constructing query-biased summaries: a comparison of human and system generated snippets. In *Proceedings of the third symposium on Information interaction in context*, IIiX '10, pages 195–204, New York, NY, USA. ACM.

Conroy, J. M., Schlesinger, J. D., and O'Leary, D. P. (2006). Topic-focused multi-document summarization using an approximate oracle score. In *Proceedings of the COLING/ACL on Main conference poster sessions*, COLING-ACL '06, pages 152–159, Stroudsburg, PA, USA. Association for Computational Linguistics.

Cutrell, E. and Guan, Z. (2007). What are you looking for?: an eye-tracking study of information usage in web search. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, CHI '07, pages 407–416, New York, NY, USA. ACM.

Daumé, III, H. and Marcu, D. (2006). Bayesian query-focused summarization. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, ACL-44, pages 305–312, Stroudsburg, PA, USA. Association for Computational Linguistics.

Friedman, J. (2001). Greedy function approximation: a gradient boosting machine. *Annals of Statistics*, pages 1189–1232.

Friedman, J. (2002). Stochastic gradient boosting. *Computational Statistics & Data Analysis*, 38(4):367–378.

Han, K.-S., Baek, D.-H., and Rim, H.-C. (2000). Automatic text summarization based on relevance feedback with query splitting (poster session). In *Proceedings of the fifth international workshop on on Information retrieval with Asian languages*, IRAL '00, pages 201–202, New York, NY, USA. ACM.

He, J., Shu, B., Li, X., and Yan, H. (2010). Effective time ratio: A measure for web search engine with document snippet. In *AIRS '10: proceeding of the Sixth Asia Information Retrieval Societies Conference*.

Kanungo, T., Ghamrawi, N., Kim, K. Y., and Wai, L. (2009). Web search result summarization: title selection algorithms and user satisfaction. In *Proceedings of the 18th ACM conference on Information and knowledge management*, CIKM '09, pages 1581–1584, New York, NY, USA. ACM.

Kanungo, T. and Orr, D. (2009). Predicting the readability of short web summaries. In *WSDM '09: Proceedings of the Second ACM International Conference on Web Search and Data Mining*, pages 202–211, New York, NY, USA. ACM.

Ko, Y., An, H., and Seo, J. (2007). An effective snippet generation method using the pseudo relevance feedback technique. In *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '07, pages 711–712, New York, NY, USA. ACM.

Kushniruk, A. W., Kan, M.-Y., McKeown, K., Klavans, J., Jordan, D., LaFlamme, M., and Patel, V. L. (2002). Usability evaluation of an experimental text summarization system and three search engines: implications for the reengineering of health care interfaces. In *Proc AMIA Symp. 2002*, pages 420–424.

Li, P., Burges, C. J. C., and Wu, Q. (2007). Mcrank: Learning to rank using multiple classification and gradient boosting. In *NIPS*.

Liu, T.-Y. (2011). *Learning to Rank for Information Retrieval*. Springer.

Mani, I., House, D., Klein, G., Hirschman, L., Firmin, T., and Sundheim, B. (1999). The tipster summac text summarization evaluation. In *Proceedings of the ninth conference on European chapter of the Association for Computational Linguistics*, EACL '99, pages 77–85, Stroudsburg, PA, USA. Association for Computational Linguistics.

Metzler, D. and Kanungo, T. (2008). Machine learned sentence selection strategies for query-biased summarization. In *SIGIR Learning to Rank Workshop*.

Murray, G., Kleinbauer, T., Poller, P., Becker, T., Renals, S., and Kilgour, J. (2009). Extrinsic summarization evaluation: A decision audit task. *TSLP*, 6(2).

Otterbacher, J., Erkan, G., and Radev, D. R. (2005). Using random walks for question-focused sentence retrieval. In *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, HLT '05, pages 915–922, Stroudsburg, PA, USA. Association for Computational Linguistics.

Park, S. and An, D. U. (2010). Automatic query-based personalized summarization that uses pseudo relevance feedback with nmf. In *Proceedings of the 4th International Conference on Uniquitous Information Management and Communication*, ICUIMC '10, pages 61:1–61:7, New York, NY, USA. ACM.

Radev, D., Allison, T., Blair-Goldensohn, S., Blitzer, J., Çelebi, A., Dimitrov, S., Drabek, E., Hakim, A., Lam, W., Liu, D., Otterbacher, J., Qi, H., Saggion, H., Teufel, S., Topper, M., Winkel, A., and Zhang, Z. (2004). MEAD - a platform for multidocument multilingual text summarization. In *LREC 2004*, Lisbon, Portugal.

Rose, D. E., Orr, D., and Kantamneni, R. G. P. (2007). Summary attributes and perceived search quality. In *Proceedings of the 16th international conference on World Wide Web*, WWW '07, pages 1201–1202, New York, NY, USA. ACM.

Sakai, T. and Sparck-Jones, K. (2001). Generic summaries for indexing in information retrieval. In *Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '01, pages 190–198, New York, NY, USA. ACM.

Sanderson, M. (1998). Accurate user directed summarization from existing tools. In *Proceedings of the seventh international conference on Information and knowledge management*, CIKM '98, pages 45–51, New York, NY, USA. ACM.

Savenkov, D., Braslavski, P., and Lebedev, M. (2011). Search snippet evaluation at yandex: lessons learned and future directions. In *Proceedings of the Second international conference on Multilingual and multimodal information access evaluation*, CLEF'11, pages 14–25, Berlin, Heidelberg. Springer-Verlag.

Tombros, A. and Sanderson, M. (1998). Advantages of query biased summaries in information retrieval. In *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '98, pages 2–10, New York, NY, USA. ACM.

Trappett, M., Geva, S., Trotman, A., Scholer, F., and Sanderson, M. (2012). Overview of the inex 2011 snippet retrieval track. In *Proceedings of the 10th international conference on Initiative for the evaluation of XML retrieval*, INEX'11, Berlin, Heidelberg. Springer-Verlag.

Turpin, A., Scholer, F., Jarvelin, K., Wu, M., and Culpepper, J. S. (2009). Including summaries in system evaluation. In *SIGIR '09: Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*, pages 508–515, New York, NY, USA. ACM.

Varadarajan, R. and Hristidis, V. (2005). Structure-based query-specific document summarization. In *Proceedings of the 14th ACM international conference on Information and knowledge management*, CIKM '05, pages 231–232, New York, NY, USA. ACM.

Wang, C., Jing, F., Zhang, L., and Zhang, H.-J. (2007). Learning query-biased web page summarization. In *Proceedings of the sixteenth ACM conference on Conference on information and knowledge management*, CIKM '07, pages 555–562, New York, NY, USA. ACM.

Wasson, M. (2002). Using summaries in document retrieval. In *Proceedings of the ACL-02 Workshop on Automatic Summarization - Volume 4*, AS '02, pages 27–44, Stroudsburg, PA, USA. Association for Computational Linguistics.

White, R. W., Jose, J. M., and Ruthven, I. (2003). A task-oriented study on the influencing effects of query-biased summarisation in web searching. *Inf. Process. Manage.*, 39(5):707–733.

White, R. W., Ruthven, I., and Jose, J. M. (2002). Finding relevant documents using top ranking sentences: an evaluation of two alternative schemes. In *Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '02, pages 57–64, New York, NY, USA. ACM.

Zheng, Z., Chen, K., Sun, G., and Zha, H. (2007). A regression framework for learning ranking functions using relative relevance judgments. In *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '07, pages 287–294, New York, NY, USA. ACM.

A Comparison and Improvement of Online Learning Algorithms for Sequence Labeling

Zhengyan He Houfeng Wang *
Key Laboratory of Computational Linguistics (Peking University) Ministry of Education,China
hezhengyan.hit@gmail.com, wanghf@pku.edu.cn

ABSTRACT
Sequence labeling models like conditional random fields have been successfully applied in a variety of NLP tasks. However, as the size of label set and dataset grows, the learning speed of batch algorithms like L-BFGS quickly becomes computationally unacceptable. Several online learning methods have been proposed in large scale setting, yet little effort has been made to compare the performance of these algorithms. Comparison is often carried out on a few datasets with fine tuned parameters for specific algorithm. In this paper, we investigate and compare several online learning algorithms for sequence labeling with datasets varying in scale, feature design and label set. We find that Dual Coordinate Ascent (DCA) is robust across datasets even without careful tuning of parameter. Furthermore, a recently proposed variant of Stochastic Gradient Descent (SGD), Adaptive online gradient Descent based on feature Frequency information (ADF), has very fast training speed compared with plain SGD, but fails to converge under certain conditions. Finally, We propose a simple modification of ADF, which bears comparable convergence speed with ADF, and is consistently better than plain SGD.

TITLE AND ABSTRACT IN CHINESE

序列标注在线学习算法的比较和改进

序列标注模型，如条件随机场，已广泛用于自然语言处理的很多任务中。但随着数据规模和标记集的增大，批量学习算法（如L-BFGS）的训练时间复杂性变得越来越不可接受；于是，出现了多个大规模环境下的在线学习算法。这些算法有各自的特点，但对这些算法的比较研究很少有报道。通常情况下都是针对几个数据集，通过对特定算法细致调整参数来比较最后的测试结果。本文针对几个典型的序列标注在线学习算法，在不同规模、特征设计和标记集合的数据集上进行了比较研究。结果发现，即使没有特别对参数调优，对偶坐标上升算法（DCA）在不同数据集上也能有很好的表现；而随机梯度下降算法（SGD）的一个变种算法——基于特征频度的适应性在线梯度下降法（ADF）与普通的SGD相比，训练速度更快，但不能保证总收敛。最后，本文还对ADF提出了一种简单的改进，改进后的算法好于普通的SGD，与ADF 收敛速度接近。

KEYWORDS: Sequence labeling, online learning, stochastic gradient descent, named entity recognition.

KEYWORDS IN CHINESE: 序列标注模型，在线学习，随机梯度下降，命名实体识别

*Corresponding author

Proceedings of COLING 2012: Technical Papers, pages 1147–1162,
COLING 2012, Mumbai, December 2012.

1 Introduction

Sequence labeling models have been widely used in a variety of NLP tasks, such as word segmentation, part-of-speech tagging, chunking, and named entity recognition. Various sequence labeling models have been proposed, like hidden Markov models (HMM) (Rabiner, 1989), structured perceptron (Collins, 2002), conditional random fields (CRFs) (Lafferty et al., 2001) and SVM-HMM (Tsochantaridis et al., 2006). In recent years, discriminative models gain significant popularity over generative models on these tasks, and achieve state-of-the-art performance on most above tasks. Their strength and flexibility come from their ability to incorporate arbitrary declarative features.

CRFs is one of such discriminative models for sequence labeling built upon maximum entropy principle. The learning algorithms of CRFs can be divided into batch methods and online methods. Batch methods update parameters by estimating gradient over the entire training data, while online methods estimate noisy gradient with a small portion of the training data, and update parameters frequently. Among all the batch methods, L-BFGS is the most widely used and outperforms others by a substantial margin (Malouf et al., 2002); Conjugate-gradient (CG) method with proper preconditioner can converge as fast as L-BFGS (Sha and Pereira, 2003). However, discriminative models for typical sequence labeling tasks are very large and may involve hundreds of thousands of features, rendering even fastest batch learning methods very slow and impractical for large scale datasets.

Several online learning algorithms have been proposed to speed the training process of structured prediction problems, such as Passive-Aggressive (PA) algorithm (Crammer et al., 2006), Dual Coordinate Ascent (DCA) (Martins et al., 2010) and Stochastic Gradient Descent (SGD). SGD is known for its performance in the back propagation training of neural network. It also shows extremely good performance on machine learning tasks such as SVM (Bordes et al., 2009), CRFs (Vishwanathan et al., 2006), and Markov Logic Networks (Poon and Vanderwende, 2010). It may reach optimal performance even before it sees the whole training data on large datasets (Bottou and Bousquet, 2008). SGD takes typically 5-10 iterations to converge when training a multiclass Maximum Entropy (ME) model, while it takes over 50 iterations when training CRFs, much slower than training its unstructured counterpart, multiclass ME. We show how simple feature frequency adaptive strategy may help accelerate training of CRFs within 10 iterations.

Despite recent progresses in online learning algorithms, little effort has been made to compare these algorithms thoroughly. In this paper, we focus on several online learning algorithms for sequence labeling. More specifically, we investigate PA, DCA, SGD and SGD's variant ADF (Sun et al., 2012). We perform comparison on several standard datasets with diverse settings of feature design and label set. We make it as close as possible to real application scenarios whenever resources are available. Experiment reveals distinct behavior of these algorithms under different settings.

Our contributions are threefold. First, we make a fair and extensive comparison of state-of-the-art online learners for sequence labeling and characterize the strength of each algorithm. Second, we confirm the effectiveness of ADF on most datasets despite its lack of theoretical convergence guarantee for now; inspired by ADF, we propose Modified ADF (MADF), which guarantees convergence and converges as fast as ADF. Finally, we explore the use of Tongyici Cilin (Extended) as a semantic lexicon in Chinese named entity recognition, which remains unexplored for this task in previous work.

The remaining of this paper is organized as follows: in Section 2, we describe in detail the online learning algorithms, and propose MADF; in Section 3 we evaluate performance of these algorithms, and present novel usage of Tongyici Cilin (Extended) in Experiment 2; we review related work in Section 4; finally we conclude.

2 Online Learning Algorithms for Sequence Labeling

A sequence labeling task is defined as follows: given an observation sequence $\mathbf{x} = \{x_1, x_2, \ldots, x_n\}$, output its corresponding label sequence $\mathbf{y} = \{y_1, y_2, \ldots, y_n\}$, one label for each x_i. The output space is $|Y|^n$ where Y is the label set that each individual y_i takes values from, and n is the length of $\mathbf{y}$.

In discriminative sequence labeling models, feature functions are used to describe interdependency between observed and hidden variables. Under first order Markov assumption, the feature function can be divided into transition features $\phi(\mathbf{x}, y_i, y_{i+1})$ and emission features $\phi(\mathbf{x}, y_i)$. These can be combined to $\phi(\mathbf{x}, \mathbf{y})$.

Dynamic programming technique like viterbi decoding is often employed during inference:

$$\hat{\mathbf{y}} = \arg\max_{\mathbf{y}} \theta^T \phi(\mathbf{x}, \mathbf{y}) \tag{1}$$

where θ are model parameters. Next we will describe how to estimate parameters θ with different online algorithms. Table 1 shows a list of denotations for convenience.

$\mathbf{x}$	input sequence
$\mathbf{y}$	output label sequence
θ	parameter vector
$\phi(\mathbf{x}, \mathbf{y})$	feature function that maps sequence to feature vector
η_t	learning rate for the t-th sample
λ	regularization weight of $\frac{\lambda}{2}\|\theta\|_2^2$
M	number of training samples (sequences)
$l(\hat{\mathbf{y}}, \mathbf{y})$	cost function given the predicted and gold sequences

Table 1: Denotations

2.1 Passive-aggressive algorithm (PA)

Passive-Aggressive (PA) algorithm (Crammer et al., 2006) is a family of margin based online learning algorithms. This algorithm updates the parameter to satisfy the constraint imposed by the current example (aggressively), and does nothing if the current example is correctly classified (passively). Equation 2 defines the objective function for PA algorithm (referred by the author as PA-I):

$$\theta^{(t+1)} = \arg\min_{\theta} \frac{1}{2}\|\theta - \theta^{(t)}\|^2 + C\xi \quad \text{s.t.} \quad l(\hat{y}, y) < \xi \text{ and } \xi \geq 0 \tag{2}$$

where ξ is a non-negative slack variable that copes with wrongly labeled data; C controls the aggressiveness of parameter updating, and is a trade-off between convergence speed and model

quality. The update rule of parameter is as follows:

$$\theta^{(t+1)} = \theta^{(t)} - \eta(\phi(\mathbf{x}^{(t)}, \hat{\mathbf{y}}^{(t)}) - \phi(\mathbf{x}^{(t)}, \mathbf{y}^{(t)})) \tag{3}$$

$$\text{where} \quad \eta = \min\{C, \frac{\theta^T(\phi(\mathbf{x}^{(t)}, \hat{\mathbf{y}}^{(t)}) - \phi(\mathbf{x}^{(t)}, \mathbf{y}^{(t)})) + l(\mathbf{y}^{(t)}, \hat{\mathbf{y}}^{(t)})}{\|\phi(\mathbf{x}^{(t)}, \hat{\mathbf{y}}^{(t)}) - \phi(\mathbf{x}^{(t)}, \mathbf{y}^{(t)})\|^2}\} \tag{4}$$

$$\text{and} \quad \hat{\mathbf{y}}^{(t)} = \arg\max_{\mathbf{y}} \theta^T(\phi(\mathbf{x}^{(t)}, \mathbf{y}) - \phi(\mathbf{x}^{(t)}, \mathbf{y}^{(t)})) + l(\mathbf{y}^{(t)}, \mathbf{y}) \tag{5}$$

where $l(\mathbf{y}^{(t)}, \hat{\mathbf{y}}^{(t)})$ is the penalty we incur if our prediction is $\hat{\mathbf{y}}^{(t)}$ and the true label is $\mathbf{y}^{(t)}$. $\hat{\mathbf{y}}^{(t)}$ can be solved with cost augmented decoding, which can be efficiently accomplished if $l(\mathbf{y}^{(t)}, \hat{\mathbf{y}}^{(t)})$ decomposes the same way as the feature vector function (Smith, 2011). This is referred to as a max-loss update in (Crammer et al., 2006).

When applied to sequence labeling PA is a special case of the general algorithm where output $\mathbf{y}$ is a label sequence. Hamming loss (Eq. 6) is often used. However, other loss can also fit when one faces with task specific needs (Song et al., 2012) (Mohit et al., 2012).

$$l(\hat{\mathbf{y}}, \mathbf{y}) = Hamming(\hat{\mathbf{y}}, \mathbf{y}) = \sum_{i=1}^{n} \delta_{y_i \neq \hat{y}_i}(\hat{y}_i, y_i) \tag{6}$$

2.2 Dual coordinate ascent (DCA)

(Martins et al., 2010) present a general framework for online learning of structured classifiers. It bears some resemblance to the PA algorithm in that it shares the passive-aggressive property of PA. This algorithm applies to a wide class of loss functions; CRFs, SVM, structured perceptron can all be deemed as its special cases. Furthermore, learning rate is automatically determined for each instance, hence pesky learning rate tuning is no longer needed. The learning objective is the sum of loss on datasets plus a regularization term:

$$\min_{\theta \in \mathbb{R}^d} \lambda R(\theta) + \frac{1}{m} \sum_{t=1}^{m} L(\theta; \mathbf{x}^{(t)}, \mathbf{y}^{(t)}) \tag{7}$$

The update rule 8 is very similar to the PA algorithm in large margin setting (Eq. 3), if we divide Equation 7 by λ, replace $\frac{1}{\lambda m}$ with C, and set $\nabla L = \phi(\mathbf{x}^{(t)}, \hat{\mathbf{y}}^{(t)}) - \phi(\mathbf{x}^{(t)}, \mathbf{y}^{(t)})$. The update rule for DCA is:

$$\theta^{(t+1)} = \theta^{(t)} - \eta_t \nabla L(\theta; \mathbf{x}^{(t)}, \mathbf{y}^{(t)}) \tag{8}$$

$$\text{where} \quad \eta^{(t)} = \min\{\frac{1}{\lambda m}, \frac{L(\theta^{(t)}; \mathbf{x}^{(t)}, \mathbf{y}^{(t)})}{\|\nabla L(\theta^{(t)}; \mathbf{x}^{(t)}, \mathbf{y}^{(t)})\|^2}\} \tag{9}$$

As for sequence labeling problem, CRFs is derived by setting the loss function to the negative log-loss with no cost function, i.e. $L_{CRF} = -\log P(\mathbf{y}_t|\mathbf{x}_t)$. The loss function of CRFs is just a special case of the general loss function L.

2.3 CRFs with SGD learning and its variants

Linear chain conditional random fields (CRFs) (Lafferty et al., 2001) is widely used model for sequence labeling. It predicts the output sequence $\mathbf{y}$ by its conditional probability $P(\mathbf{y}|\mathbf{x})$:

$$P(\mathbf{y}|\mathbf{x}) = \frac{1}{Z(\mathbf{x})} \exp \theta^T \phi(\mathbf{x}, \mathbf{y}) \tag{10}$$

where $Z(\mathbf{x}) = \sum_{\mathbf{y}} \exp \theta^T \phi(\mathbf{x}, \mathbf{y})$ is normalization term which ensures that the probability sums to 1. The training objective is to maximize the likelihood on the training data (often accompanies a prior over parameters to get max-a-posterior estimation), which is equivalent to minimizing a negative log loss over all data points plus a regularizer:

$$L = \sum_{i=1}^{m} -\log P(\mathbf{y}|\mathbf{x}) + \lambda m \|\theta\|^2 \tag{11}$$

Here we multiply the regularizer by m to keep it consistent with Equation 7. This way λ has the same meaning as in Equation 7. Minimizing this objective gives the gradient for one instance in online setting:

$$\frac{\partial}{\partial \theta} L(\theta; \mathbf{x}^{(t)}, \mathbf{y}^{(t)}) = E_{P(\mathbf{y}|\mathbf{x};\theta^{(t)})} \phi(\mathbf{x}^{(t)}, \mathbf{y}^{(t)}) - \phi(\mathbf{x}^{(t)}, \mathbf{y}^{(t)}) + \lambda\theta \tag{12}$$

where the expectation of feature vector given current example $\mathbf{x}^{(t)}$ is taken over all possible sequences of $\mathbf{y}$, and can be efficiently computed using forward-backward algorithm.

Generally regularizer is added to avoid overfitting. Various regularizers have been proposed. L_2 regularizer, $R(\theta) = \frac{\lambda}{2}\|\theta\|_2^2$, which is used by various CRF based tools, often leads to superior performance and can be numerically optimized. L_1 regularizer, $R(\theta) = \sigma\|\theta\|_1$, known as Lasso, encourages sparse parameter. Elastic net, a linear interpolation of L_1 and L_2 regularizer, is used by (Lavergne et al., 2010) to regularize CRFs, and performs as good as L_2 regularizer, while still retaining compact model.

2.3.1 SGD and its variants

Stochastic gradient descent (SGD) is known for its fast convergence on machine learning tasks (Bottou and Bousquet, 2008) (Vishwanathan et al., 2006) (Shalev-Shwartz et al., 2007). Every time the algorithm randomly draws one sample (or small batch of samples in mini-batch setting), and performs update according to the gradient of this sample. In general, SGD has the following simple update rule:

$$\theta^{(t+1)} = \theta^{(t)} - \eta^{(t)} \mathbf{B} \nabla L \tag{13}$$

where $\eta^{(t)}$ is a scalar learning rate, $\mathbf{B}$ is a matrix; $\mathbf{B} = \mathbf{I}$ for a plain SGD and $\mathbf{B} \approx \mathbf{H}^{-1}$ for second order SGD. Despite its ease of implementation, it's generally hard to tune and schedule the learning rate properly.

(Murata, 1998) shows that with a $1/t$-annealing learning rate it can be asymptotically as effective as batch learning in terms of generalization error. (Bottou and Bousquet, 2008) further shows that utilizing second order information by setting $\eta\mathbf{B} = \frac{1}{t}\mathbf{H}^{-1}$, optimal asymptotic convergence rate is achieved. With $\eta = 1/t$ and fixed $\mathbf{B}$, convergence is guaranteed based on the theory of stochastic approximation (Murata, 1998).

Various ways of approximating the inverse of Hessian have been proposed (Hsu et al., 2009) (Vishwanathan et al., 2006). But these methods are much slower than plain SGD in terms of execution time of one pass over the entire dataset, so the speed up is very limited. Full approximation of $\mathbf{H}^{-1}$ is quite expensive , so low rank approximation and diagonal approximation (Bordes et al., 2009) is very appealing. The next two algorithms is closely related to diagonal rescaling.

2.3.2 ADF

In ADF, instead of a single global learning rate in plain SGD, each dimension of parameters has its own learning rate. The learning rate decays periodically according to its associated feature frequency, with high frequency decaying faster. This adaptive strategy is based on the intuition that frequent observed features are more adequately trained so smaller learning rates are needed. The author shows its high convergence speed compared with plain SGD in Chinese word segmentation task.

The method works well in most of our experiments, except for a few datasets. As we observed in our experiments in several other sequence labeling tasks, despite its speed of convergence and of reduction in training set error rate, it might fail to generalize well to testset and its parameters fluctuate a lot with different random shuffling of data. Moreover, it is not clear how to tune the upper bound and lower bound of the decay factor. In fact, ADF can be seen as diagonal approximation of the inverse of Hessian with exponential decrease learning rate based on frequency adaptive information. Unfortunately, this decrease in learning rate does not have theoretically convergence guarantee for now, although it works well in practice. (Murata, 1998) shows that SGD with $1/t$-annealing learning rate guarantees convergence.

2.4 Modified ADF (MADF)

Second order SGD (2SGD) uses an approximation of inverse of Hessian by setting $\eta\mathbf{B} = \frac{1}{t}\mathbf{H}^{-1}$ in order to achieve the optimal learning rate. Inspired by ADF and current theory foundation of 2SGD, we propose to use feature frequency information to approximate $\mathbf{H}^{-1}$, while still keeping a $\eta = 1/t$ annealing factor, so convergence is guaranteed (Eq. 14).

Our method works as follows: at the beginning of the algorithm, we compute the diagonal scaling matrices $\mathbf{B}$ using Equation 15, which is of the same size as the parameter vector. For each dimension of θ, we use a separate learning rate $\frac{1}{t}B_{ii}$.

$$\theta^{(t+1)} = \theta^{(t)} - \frac{1}{t}\mathbf{B}\nabla L(\theta) \tag{14}$$

$$\mathbf{B}_{ii} = \frac{1}{\beta + (\alpha - \beta) \times \frac{\#\phi(x,y)}{\#tokens}} \quad \text{where} \quad a = \frac{1}{\alpha}\ b = \frac{1}{\beta} \tag{15}$$

where a and b serve as lower and upper bounds of diagonal scaling element $\mathbf{B}_{ii}$. We keep $\mathbf{B}$ constant during the training process. $\#\phi(x,y)/\#tokens$ is the relative feature frequency associated with i-th dimension.

There are two main differences of our method compared with ADF. The first is how the frequency is counted. In ADF, frequency is counted as $\sum_y \phi(x,y)$ per sentence and is the same for each predicate x with different y. Our method counts $\phi(x,y)$ per token in the training set and use separate learning rate for predicate x with different y. Another difference is that we keep an annealing learning rate $\eta^{(t)} = 1/t$ with fixed diagonal scaling matrix $\mathbf{B}$, while ADF can be seen as exponential decaying $\mathbf{B}$ with constant η.

It is difficult to set the upper and lower bounds of scaling factor $\mathbf{B}_{ii}$. One solution is to grid search a and b with held-out dataset. In this paper, we find it works surprisingly well by setting $a = 0.001$ and $b = 1$ on almost all datasets.

It is easy to interpret our method in the view of input rescaling. In the back propagation learning of neutral network, mapping a too large input to a relatively small output would result in a small learning rate in order to ensure stable convergence, leading to slow convergence speed [1]. (LeCun et al., 1998) suggest simply to normalize the input to combat this problem. Furthermore, input scaling is closely related to diagonal approximation of the inverse of Hessian (Bordes et al., 2009). However, scaling input feature value in sparse dataset is not realistic. Our idea is that in batch setting, the update of parameters associated with frequent features tend to be larger than those associated with rare features, so we scale the learning rate by the inverse of its associated relative feature frequency.

We show in Section 3 how this simple adaptive learning rate can significantly speedup the learning process while still is equipped with convergence guarantee. It runs as fast as plain SGD in terms of per iteration execution time, without the penalty of approximating the inverse of Hessian. The speedup is attributed to a proper estimation of initial learning rate, especially on datasets with more skewed feature frequency distribution.

3 Evaluation and Analysis

3.1 Implementation

There are several implementation issues on how to obtain good performance with different algorithms.

Weight averaging: averaged perceptron, PA and DCA can get much better performance by weight averaging. We only have to maintain two weight vectors $\theta^{(t)}$ and $\hat{\theta}^{(t)}$; each time we perform update $\theta^{(t+1)} = \theta^{(t)} - \nabla L$ and $\hat{\theta}^{(t+1)} = \hat{\theta}^{(t)} - t\nabla L$. Finally the average weight is obtained by $\bar{\theta} = \sum_{t=0}^{T} \theta^{(t)} = \theta^{(T)} - \hat{\theta}^{(T)}/T$.

Randomize data: if we have to pass the dataset multiple times, it's better to randomize the dataset. This can get better performance on all online algorithms in our experiments, not only SGD variants.

Regularization with L_2: In NLP tasks, the feature vector is typically sparse. The gradient ∇L (see Eq. 12) consists of two parts, one corresponds with the loss and the other with the regularizer. The first part is often sparse and can be efficiently carried out. The update of the regularizer part is dense and quite expensive if done for every sample.

There are two methods that can combat this problem. (Shalev-Shwartz et al., 2007) propose to represent the weight vector θ by the product of one scaling factor and one vector. We can see the reason by a simple rearrangement of the formula $\theta^{(t+1)} = \theta^{(t)} - \eta(\nabla l + \lambda\theta^{(t)}) = (1 - \eta\lambda)\theta^{(t)} - \eta\nabla l$. The first term can be done efficiently with a scalar product. (Bordes et al., 2009) propose in SVMSGD2 another method in which the regularizer term is treated as a special example and updated periodically. The method works on both first order and second order SGD. We will use this method in our experiment if not particularly mentioned.

Initial learning rate for SGD: The initial learning rate of SGD plays a critical role in the whole process of learning. It is chosen by heuristic method. We can sample a subset of the training data, run SGD algorithm for one pass over the subset and pick the learning rate with smallest training objective value as the initial guess of learning rate. This method is generally helpful for all SGD variants. We use this setting for all variants of SGD.

[1] http://www.willamette.edu/~gorr/classes/cs449/precond.html

3.2 Datasets settings

We compare the performance on several sequence labeling tasks, namely Chinese word segmentation, Chinese named entity recognition, CoNLL 2000 chunking, CoNLL 2003 NER, and Chinese part-of-speech tagging. The datasets vary across tasks in label set size and feature design. We inject as much knowledge as possible to mimic real application scenarios. For under-resourced tasks, we simply use token based n-gram features. Table 2 gives a brief view of features used. Table 4 shows the statistics after feature generation.

Tasks	Type	Features
Chinese word segmentation	basic	character unigram $w_{-2}, w_{-1}, w_0, w_1, w_2$, character bigram $w_{-2}w_{-1}, w_{-1}w_0, w_0w_1, w_1w_2$, whether w_j and w_{j+1} are identical and whether w_j and w_{j+2} are identical in windows of 2 characters on the left and 2 characters on the right; unigram/bigram dictionary features as described in (Sun et al., 2012)
	extended	accessor variety and mutual information (Sun and Xu, 2011)
Chinese named entity recognition	basic	character unigram and bigram in the context window of size 2; bigram of previous character and next character; whether character is word,letter,digit or punctuation in windows of size 1
	extended	basic features plus Tongyici Cilin (extended) derived word boundary and semantic type, entity list derived from Baidu Baike, in windows of size 2; whether the current character is a single character word or multiple character word through forward maximum matching and backward maximum matching;
Chunking	basic	word unigram in windows of size 2, word bigram in windows of size 1; part-of-speech unigram,bigram,trigram in windows of size 2;
English named entity recognition	basic	word unigram in windows of size 2, word bigram in windows of size 1; part-of-speech unigram,bigram,trigram in windows of size 2; character shape unigram and bigram in windows of size 2
	extended	basic features plus word cluster code prefix with length 4,10,16,20, both unigram and bigram in windows of size 2; gazetteer list feature of the current token;
Chinese POS tagging	basic	word unigram and bigram in windows of size 2; current word prefix and suffix of size up to 3, which is the baseline of (Sun and Uszkoreit, 2012).

Table 2: Features used for different tasks. For basic features, I refer to most simple token based feature and word type (letter,digit,punctuation) within a window of certain size. Extended features vary with available resources.

1. **Chinese Word Segmentation (CWS)** SigHan 2005 dataset is used for CWS [2].

2. **Chinese Named Entity Recognition (NER)** Named entity recognition requires large amount of world knowledge. List lookup features from gazetteer, lexicon and dictionaries can greatly enhance an NER system (Nadeau and Sekine, 2007). So for extended features, we explore the use of Baidu Baike and Tongyici Cilin (Extended) [3] as two knowledge

[2] http://www.sighan.org/bakeoff2005/

[3] http://ir.hit.edu.cn/phpwebsite/index.php?module=pagemaster&PAGE_user_op=view_page&PAGE_id=162

sources. Table 3 gives a brief view of Tongyici Cilin. The first column gives the category

Category	Word Cluster
Af10B05#	省长(governor of province) 市长(mayor) 县长(county head) 区长 乡长 村长 ...
Ae13A10#	**教授**(professor) 副教授(associate professor) 讲师(instructor) 助教(teaching assistant) ...
Hg05A01=	讲课 授课 讲授 讲解 **教授**(teach) 教书 ...
Dm01A46#	安全部(ministry of security) 财政部(ministry of finance) 参谋部 电力部 ...

Table 3: A Snippet from Tongyici Cinlin (Extended)

one word belongs to. The category codes with prefix of different lengths give different levels of abstraction of its semantic meaning. Some of the clusters are good indicators of named entities. For instance, the first row is a good indicator of previous word being a location, and the first and second row is a good indicator of next word being a person. This categorized lexicon is quite similar to word cluster features in (Ratinov and Roth, 2009), but is more precise. However, to our certain knowledge, this resource remains unexplored in previous Chinese NER tasks.

Words in Chinese do not have space like in English. In Chinese NER task, texts are given without word boundaries, so segmentation is an essential preprocessing step. But this will bring segmentation error to the system, especially most named entities are out-of-vocabulary words. On the other hand, if we perform inference at character level directly, we quickly loss the meaning of words . We propose two simple strategies to alleviate these problems: first, while still performing inference at character level, we do forward maximum matching and backward maximum matching to provide basic word boundary features; second, word meaning is injected with the category of the matching word in Tongyici Cilin. The class of a word also serves as a mechanism of word cluster, which holds similar words together. Hence Tongyici Cilin serves as both word clusters and a lexicon when performing maximum matching.

Finally, we use entity list extracted from Baidu Baike as additional entity list features. The datasets we use are SigHan 08 Chinese NER dataset and one month of People Daily.

As our main concern is to build a resource rich feature design for Chinese named entity recognition, we do not make further comparison with other algorithms. Other approaches typically use complex model combinations, which are not directly comparable to our single model based method.

3. **CoNLL English NER and Chunking** (Ratinov and Roth, 2009) perform an extensive study on NER and extract valuable resources, which can be readily incorporated into any existing NER system. We use the word class hierarchy and gazetteer lists from their package[4]. We do not use other features for simplicity. Word class features are derived from brown clustering algorithm and intended for bridging the gap of unseen text . The brown algorithm hierarchically clusters the words, and paths with different lengths to the root represent different levels of abstraction. Gazetteers are dictionaries of named entities and injected to the system to provide world knowledge. We use these two features as described in (Ratinov and Roth, 2009).

   For English chunking task, we use the template provided with CRF++ [5], this template is

[4]http://l2r.cs.uiuc.edu/cogcomp/software.php
[5]http://crfpp.googlecode.com/svn/trunk/doc/index.html

also used in the benchmark of CRFSGD [6].

4. **Chinese part-of-speech(POS) tagging** The setting follows the baseline of (Sun and Uszkoreit, 2012). As we do not have access to the more complex features, we just use this baseline, which performs reasonably good on a different datasets.

Dataset	#tokens	#unigram	#bigram	#labels	cutoff	max %freq
CWS MSR	4,050,469	1,852,255	1,852,255	3	1	0.0428
CWS CITYU	2,403,024	1,594,695	1,594,695	3	1	0.0456
CWS PKU	1,822,380	1,202,381	1,202,381	3	1	0.0407
CWS PKU(e)	1,826,448	365,254	1	4	5	0.9954
NER MSRA	1,089,050	332,989	1	10	3	0.8763
NER CITYU	1,772,202	505,185	1	10	3	0.9143
NER PD(e)	1,811,931	744,396	1	10	3	0.8782
CoNLL2000	211,727	76,328	1	23	3	0.1424
CoNLL2003(e)	203,621	860,462	1	17	1	0.8526
CoNLL2003ned	202,644	366,086	1	9	1	0.6847
POS PKU	1,116,754	2,252,374	1	103	1	0.0667
POS CITYU	1,092,687	2,257,166	1	44	1	0.0614

Table 4: Statistics on different datasets. The meanings of each field are as follows: number of tokens, unigram features, bigram features and class labels; we use features with frequency no less than `cutoff`; `max%freq` refers to maximum relative frequency (ref. Eq. 15), a larger value means a more skewed feature frequency distribution.

3.3 Experiments

As for plain SGD, we replicate CRFSGD implementation for fair comparison. For all variants of SGD, we use the regularizer proposed by (Bordes et al., 2009). All learning rates are searched by subsampling, except for PA and DCA, which do not need to specify learning rates. For ADF, we use the same setting as (Sun et al., 2012). We run SGD for 50 iterations and other online methods for 30 iterations. This setting is suffice for most algorithms to reach a stable state.

Also note PA and DCA need to specify an aggressive parameter C, which controls how aggressively the parameter perform updates. PA seems to be very sensitive to this parameter, and the algorithm leads to bad results if C is not properly set. The algorithm converges fast with a larger C, but may not provide good generalization performance. In our experiment, we set $C = 0.01$ for PA, which is a trade-off between convergence speed and an accurate model. DCA is not very sensitive to this parameter so we set C to 1.

For every task we also plot the final performance of CRF++ and Wapiti [7] at the beginning of iterations. We do not plot the learning curves because they use different learning methods (Wapiti does contain online learners but needs to switch to L-BFGS to fine tune the model parameter, so we only report results of L-BFGS learner). For POS tagging we do not plot the results of CRF++, because CRF++ will run for weeks. CRF++ uses L-BFGS for parameter estimation, and we use default stop condition. Wapiti (Lavergne et al., 2010) uses elastic net regularizer, and does feature selection automatically while training. The resulting model is

[6] http://leon.bottou.org/projects/sgd
[7] http://wapiti.limsi.fr/

compact and small, and still retains performance comparable to L_2-regularizer. We use the default setting of regularizer weights and a stop condition that error rate of development set does not further decrease in window of size 10.

For other variants of SGD, PSA (Hsu et al., 2009) and SMD (Vishwanathan et al., 2006) are out of our consideration because they are typically more than 10 times slower than plain SGD in execution time of one pass over the dataset, despite their theoretical appealing one pass over the data. We tried averaged SGD (ASGD) (Xu, 2011) on two of our datasets, but it did not perform so well even if we tried several switch time between SGD and ASGD.

For Chinese word segmentation, we evaluate F-score using the script for SigHan 2005 bakeoff. For all NER and chunking tasks, we report phrase based F-score with the script provided by CoNLL. For POS tagging, we report token based accuracy. Results are listed in Table 5.

Datasets	CWS MSR	CWS CITYU	CWS PKU	CWS PKU(e)	NER MSRA	NER CITYU	NER PD(e)	Chunking	CoNLL 2003 Eng(e)	CoNLL 2003 Ned	POS PKU	POS CITYU
PA	96.60	94.10	94.80	95.60	87.84	**80.91**	91.60	93.26	85.15	**75.57**	94.09	89.09
DCA	**97.00**	94.10	**95.20**	95.50	**89.45**	80.39	91.78	93.76	86.66	75.33	**94.24**	**89.21**
SGD	96.90	94.10	95.10	95.60	88.91	80.05	92.05	93.71	86.60	74.77	94.04	88.98
ADF	96.90	94.10	**95.20**	95.60	88.97	79.91	91.82	93.70	86.66	75.18	94.04	89.10
MADF	96.90	94.10	95.10	**95.80**	88.94	79.85	**92.13**	**93.78**	**86.77**	74.94	94.01	89.08
Wapiti	96.50	94.30	94.60	**95.80**	88.66	80.07	91.84	93.71	**87.25**	75.22	93.90	88.70
CRF++	96.70	94.40	94.70	95.70	88.91	79.80	92.09	93.74	86.50	74.64	na	na

Table 5: Results on all datasets. Note that PA and Wapiti have different model/regularizer from other CRFs models, hence not directly comparable to other CRFs-based models. (e) means extended feature set.

3.4 Discussion

Differences across datasets: For well resourced tasks, i.e. CWS pku(e), NER People Daily(e), CoNLL 2003 English NER(e) (4,7,9 in figure 1), SGD and its variants give better performance than PA and DCA, with faster convergence speed; while PA and DCA give very robust performance on under-resourced tasks despite the presence of only simple token based features, partly attribute to their weight averaging mechanism. Another observation is that small or under-resourced datasets often make the learning curve of SGD and its variants fluctuate a lot.

For simple datasets with only token based features, e.g. 6,10,11,12, PA and DCA performs relatively good or even better than other methods. PA is a margin based method and may generalize well to unseen data under such circumstances. Moreover, PA and DCA all use weight averaging for better generalization, which proves useful under simple feature set. In other cases, DCA performs as good as other training methods and converges as fast as SGD, but are always more stable. This may ease the selection of stopping criterion, as a non-stable learning curve may stop accidentally at a bad point. (Lavergne et al., 2010) suggest it is good practice to use a separate development set to determine a stop criterion, but this set is not always available. Note also that the aggressiveness parameter C for PA is critical for a reasonably good performance, as in chunking (8) and English NER (9) tasks, PA gets very poor results.

Effect of η_0: As we observe in our experiments, the high convergence speed is largely determined by a good choice of initial learning rate. For plain SGD, high frequency features cause the initial estimate of learning rate to decrease, resulting in a low convergence speed. In datasets

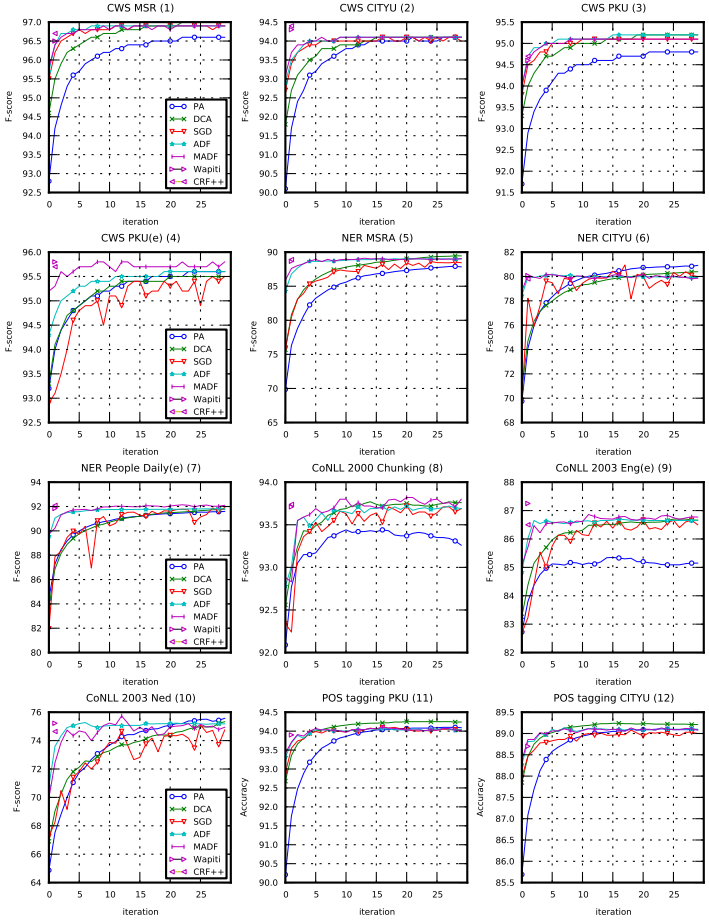


Figure 1: F-score (for CWS, NER, chunking) or accuracy (for POS tagging) on various datasets.

4,5,6,7,9,10, the initial estimations of η_0 in MADF are several times larger than that in SGD, which explains the convergence speed differences. (note the feature frequency in table 4 and convergence speed difference of SGD and MADF in figure 1.) In datasets with feature frequency not so skewed, SGD converges as fast as ADF and MADF.

ADF and MADF always converge faster than plain SGD and often achieve top performance within 10 iterations. ADF is more stable than MADF because of its fast decay of learning rate. But this effect comes at the price that ADF cannot reach top performance on some datasets. For instance, in the CWS PKU (4), Chinese NER People Daily (7) and English NER tasks (9), ADF reaches an F-score 0.1% to 0.3% lower than MADF; SGD can also performs better than ADF on these datasets after 100 iterations which I do not plot. But this is not without exception, in CoNLL 2003 Dutch NER task, where I use simple context token based features, ADF performs better than other methods. This dataset is small compared with others. The result implies ADF is more stable and suitable in most situations. MADF suffers from fluctuation on small datasets, but is still more stable, better, and faster than SGD on most datasets.

Running time: All online algorithms are 5-30 times faster than CRF++ to achieve comparable performance. ADF and MADF typically need a mere 10 iterations, and other online methods need several dozens iterations to get competitive result. In terms of one pass time over the dataset, PA clearly outperforms others because it does not have to compute normalization factor and expensive `log/exp` operations; DCA is only a little slower than SGD; ADF and MADF run as fast as SGD, while give more stable learning curve and faster convergence.

Batch vs. online: We plot the results of CRF++ and Wapiti, which can be seen as the near optimal solution to the optimization problem, we can see online methods provide as good as or even better performance than batch method. The elastic net regularizer of Wapiti is very competitive compared with L_2 regularizer. Except on CoNLL 2003 English NER data, in which Wapiti exceeds L_2 regularizer by a large margin, they gives similar results on all datasets.

Batch learning method (i.e. CRF++) rarely gives the best generalization performance. This implies that expensive batch optimization methods are not necessary for large learning tasks. Online methods will suffice. One exception is CITYU CWS task, in which CRF++ performs better than all online methods. We find the problem is how a feature fires when it is false. Both CRF++ and Wapiti treat it as a feature "F" and we omit it when we fire this feature. After adding this the F-score on testset goes up from 94.1% to 94.4% for SGD and MADF, 95.5% for ADF, comparable to CRF++. However, on a different dataset CWS PKU, adding this "F" feature decreases F-score by 0.1%, still higher than CRF++ by 0.3%.

Summary: PA is competitive with properly chosen aggressiveness parameter on under-resourced tasks. DCA converges as fast as SGD and is more stable, and is often as good as or even better than SGD variants. ADF and MADF are consistently faster than plain SGD and often reach reasonably good performance after a mere 10 passes over the datasets. ADF sometimes gets suboptimal results and losses the opportunity to further refine the parameters because of a too fast decay of learning rate; while MADF has convergence guarantee but may have a little more fluctuation in small datasets. Finally, online methods are generally several dozens times faster and get better performance than batch method.

4 Related Work

In many NLP related tasks, the data distribution is skewed, and generally only a small number of features are fired in each example, resulting in a sparse distribution. Skewed feature frequency

can affect speed of SGD algorithms, resulting in a conservative small initial guess of learning rate, which slows down the convergence speed. (Sun et al., 2012) utilize feature frequency information to speed up training of CRF model. It is simple and fast compared with other Hessian approximation methods. The sparse distribution can greatly accelerate training speed of models through clever regularization as described in (Vishwanathan et al., 2006) (Bordes et al., 2009), or through sparse forward-backward decoding (Lavergne et al., 2010).

Stochastic gradient descent (SGD) is well known for its performance on machine learning tasks (Bottou and Bousquet, 2008). Its recent successes in learning CRFs (Vishwanathan et al., 2006) and SVM (Shalev-Shwartz et al., 2007) show its advantage over batch learning algorithms in both convergence speed and generalization performance. It is particularly suitable in a large scale setting, and may achieve top performance even before seeing the whole dataset.

Various methods based on SGD have been proposed to accelerate training of CRFs. Several variants of SGD aim at theoretically one pass over the training data to get optimal performance (Hsu et al., 2009). The core idea of these methods is approximating the inverse of Hessian in order to accelerate training (and is why they are called second order SGD). However, the approximation is expensive and much slower than a plain SGD in terms of per iteration running time. (Xu, 2011) proposes averaged SGD (ASGD) that is as fast as SGD and converges within several iterations. However, in several datasets we tested, the testset performance is below standard SGD even after we tried several switch time of SGD and ASGD.

Besides the regularizer mentioned above, group Lasso has recently been proposed to regularize a structured classifier (Martins et al., 2011). The author encodes prior structural knowledge of the feature space by grouping different features into M groups and using separate regularizer weight for each group. The resulting model is compact and avoids the problem of overfitting with large number of free parameters.

Conclusion

We investigate several online learning algorithms for sequence labeling and empirically show how each algorithm performs on datasets with distinct feature design and label set. This will ease the selection of algorithms in similar tasks in future. Our experiments show that most online algorithms outperform batch method (CRF++) at both speed and generalization performance. We can gain further speedup by adopting simple strategy as ADF and MADF do.

We propose our own algorithm inspired by ADF, which is a variant of SGD. We confirm the effectiveness of ADF on several datasets. While ADF works in most situations, sometimes it leads to suboptimal solutions. Our algorithm performs consistently better than SGD, and converges as fast as ADF. These simple frequency adaptive methods can greatly accelerate training speed under skewed feature frequency distribution. As many NLP tasks involve the cycle of training the model and refining features then retraining the model, fast training methods are particularly useful, especially on large dataset with a large label set. It is also interesting to see how these two simple frequency adaptive approaches help in other structured learning problems in future.

Acknowledgments

This work was partially supported by National High Technology Research and Development Program of China (863 Program) (No. 2012AA011101), National Natural Science Foundation of China (No.91024009, No.60973053), and the Specialized Research Fund for the Doctoral Program of Higher Education of China (Grant No. 20090001110047)

References

Bordes, A., Bottou, L., and Gallinari, P. (2009). Sgd-qn: Careful quasi-newton stochastic gradient descent. *The Journal of Machine Learning Research*, 10:1737–1754.

Bottou, L. and Bousquet, O. (2008). The tradeoffs of large scale learning. *Advances in neural information processing systems*, 20:161–168.

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

Crammer, K., Dekel, O., Keshet, J., Shalev-Shwartz, S., and Singer, Y. (2006). Online passive-aggressive algorithms. *The Journal of Machine Learning Research*, 7:551–585.

Hsu, C., Huang, H., Chang, Y., and Lee, Y. (2009). Periodic step-size adaptation in second-order gradient descent for single-pass on-line structured learning. *Machine learning*, 77(2):195–224.

Lafferty, J., McCallum, A., and Pereira, F. (2001). Conditional random fields: Probabilistic models for segmenting and labeling sequence data.

Lavergne, T., Cappé, O., and Yvon, F. (2010). Practical very large scale crfs. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 504–513, Uppsala, Sweden. Association for Computational Linguistics.

LeCun, Y., Bottou, L., Orr, G., and Müller, K. (1998). Efficient backprop. *Neural networks: Tricks of the trade*, pages 546–546.

Malouf, R. et al. (2002). A comparison of algorithms for maximum entropy parameter estimation. In *Proceedings of the sixth conference on natural language learning (CoNLL-2002)*, pages 49–55.

Martins, A., Gimpel, K., Smith, N., Xing, E., Figueiredo, M., and Aguiar, P. (2010). Learning structured classifiers with dual coordinate ascent. Technical report, DTIC Document.

Martins, A., Smith, N., Figueiredo, M., and Aguiar, P. (2011). Structured sparsity in structured prediction. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 1500–1511, Edinburgh, Scotland, UK. Association for Computational Linguistics.

Mohit, B., Schneider, N., Bhowmick, R., Oflazer, K., and Smith, N. A. (2012). Recall-oriented learning of named entities in arabic wikipedia. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 162–173, Avignon, France. Association for Computational Linguistics.

Murata, N. (1998). A statistical study of on-line learning. *Online Learning and Neural Networks. Cambridge University Press, Cambridge, UK*.

Nadeau, D. and Sekine, S. (2007). A survey of named entity recognition and classification. *Lingvisticae Investigationes*, 30(1):3–26.

Poon, H. and Vanderwende, L. (2010). Joint inference for knowledge extraction from biomedical literature. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 813–821, Los Angeles, California. Association for Computational Linguistics.

Rabiner, L. (1989). A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286.

Ratinov, L. and Roth, D. (2009). Design challenges and misconceptions in named entity recognition. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL-2009)*, pages 147–155, Boulder, Colorado. Association for Computational Linguistics.

Sha, F. and Pereira, F. (2003). Shallow parsing with conditional random fields. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1*, pages 134–141.

Shalev-Shwartz, S., Singer, Y., and Srebro, N. (2007). Pegasos: Primal estimated sub-gradient solver for svm. In *Proceedings of the 24th international conference on Machine learning*, pages 807–814. ACM.

Smith, N. (2011). Linguistic structure prediction. *Synthesis Lectures on Human Language Technologies*, 4(2):1–274.

Song, H.-J., Son, J.-W., Noh, T.-G., Park, S.-B., and Lee, S.-J. (2012). A cost sensitive part-of-speech tagging: Differentiating serious errors from minor errors. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1025–1034, Jeju Island, Korea. Association for Computational Linguistics.

Sun, W. and Uszkoreit, H. (2012). Capturing paradigmatic and syntagmatic lexical relations: Towards accurate chinese part-of-speech tagging. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Jeju Island, Korea.

Sun, W. and Xu, J. (2011). Enhancing chinese word segmentation using unlabeled data. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 970–979, Edinburgh, Scotland, UK. Association for Computational Linguistics.

Sun, X., Wang, H., and Li, W. (2012). Fast online training with frequency-adaptive learning rates for chinese word segmentation and new word detection. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 253–262, Jeju Island, Korea. Association for Computational Linguistics.

Tsochantaridis, I., Joachims, T., Hofmann, T., and Altun, Y. (2006). Large margin methods for structured and interdependent output variables. *Journal of Machine Learning Research*, 6(2):1453.

Vishwanathan, S., Schraudolph, N., Schmidt, M., and Murphy, K. (2006). Accelerated training of conditional random fields with stochastic gradient methods. In *Proceedings of the 23rd international conference on Machine learning*, pages 969–976. ACM.

Xu, W. (2011). Towards optimal one pass large scale learning with averaged stochastic gradient descent. *Arxiv preprint arXiv:1107.2490*.

Creating an Extended Named Entity Dictionary from Wikipedia

Ryuichiro Higashinaka Kugatsu Sadamitsu
Kuniko Saito Toshiro Makino Yoshihiro Matsuo
NTT Media Intelligence Laboratories, NTT Corporation
1-1 Hikarinooka Yokosuka-Shi Kanagawa 239-0847 Japan
{higashinaka.ryuichiro, sadamistu.kugatsu, saito.kuniko, makino.toshiro, matsuo.yoshihiro}@lab.ntt.co.jp

ABSTRACT
Automatic methods to create entity dictionaries or gazetteers have used only a small number of entity types (18 at maximum), which could pose a limitation for fine-grained information extraction. This paper aims to create a dictionary of 200 extended named entity (ENE) types. Using Wikipedia as a basic resource, we classify Wikipedia titles into ENE types to create an ENE dictionary. In our method, we derive a large number of features for Wikipedia titles and train a multiclass classifier by supervised learning. We devise an extensive list of features for the accurate classification into the ENE types, such as those related to the surface string of a title, the content of the article, and the meta data provided with Wikipedia. By experiments, we successfully show that it is possible to classify Wikipedia titles into ENE types with 79.63% accuracy. We applied our classifier to all Wikipedia titles and, by discarding low-confidence classification results, created an ENE dictionary of over one million entities covering 182 ENE types with an estimated accuracy of 89.48%. This is the first large scale ENE dictionary.

TITLE AND ABSTRACT IN ANOTHER LANGUAGE (JAPANESE)

Wikipedia を用いた拡張固有表現辞書の構築

従来の固有表現辞書では，少ない数（最大で 18）の固有表現タイプが用いられてきたため，ピンポイントな情報抽出に適用することが難しいという問題があった．そこで，本稿では，200 の拡張固有表現タイプを用いた固有表現辞書の構築を目指す．具体的には，教師あり学習による多クラス分類器を用い，Wikipedia の見出し語を拡張固有表現タイプに分類することで辞書を構築する．特徴量として，見出し語そのもの，本文，そして，カテゴリ等のメタデータに関するものを数多く列挙し用いた．結果として，見出し語を，79.63%の精度で，拡張固有表現タイプに分類できることが分かった．学習された多クラス分類器を，Wikipedia のすべての見出し語に適用し，また，信頼度の低い分類結果については除外するようにしたところ，推定分類精度が 89.48%で，また，182 の拡張固有表現タイプをカバーする，百万以上のエントリを持つ拡張固有表現辞書を構築することができた．この辞書は，初の大規模な拡張固有表現辞書である．

KEYWORDS: Extended Named Entity, Dictionary, Wikipedia.

KEYWORDS IN JAPANESE: 拡張固有表現，辞書，Wikipedia.

Proceedings of COLING 2012: Technical Papers, pages 1163–1178,
COLING 2012, Mumbai, December 2012.

1 Introduction

For information extraction, it is important to recognize named entities (NEs) in texts. NEs are typically recognized by such techniques as support vector machines (SVMs) (Isozaki and Kazawa, 2002) and conditional random fields (Suzuki et al., 2006), using words surrounding a target entity as cues to determine if that entity belongs to a certain NE type. The limitation is that it is difficult to recognize NEs when there are few contextual cues, such as in search queries and snippets of web search results. In such cases, an NE dictionary, or a gazetteer, is particularly useful. Here, an NE dictionary means a list of entities associated with their NE types (e.g., Tokyo → LOCATION, Barack Obama → PERSON). Such a dictionary is also useful for deriving gazetteer features for training an NE recognizer (Kazama and Torisawa, 2008).

A number of studies have focused on automatically creating NE dictionaries; e.g., (Toral and Muoz, 2006; Saleh et al., 2010). Such studies generally use a small number of entity types, mostly adopting those defined in the Message Understanding Conference (MUC) (Grishman and Sundheim, 1996) or Information Retrieval and Extraction Exercise (IREX) (Sekine and Isahara, 2000). To enable more fine-grained information extraction, some attempted to cover more NE types: (Chang et al., 2009), (Watanabe et al., 2007), and (Tkatchenko et al., 2011), using up to 18 NE types. Still, we consider the current granularity of the NE types used to be too coarse, especially for tasks such as question answering (Voorhees and Dang, 2005), where systems need to pinpoint exact entities requested by users.

This paper proposes to create a dictionary of **extended NEs** (**ENEs**). An ENE hierarchy was proposed by Sekine et al. (2002); Sekine and Nobata (2004), and it defines three levels of NE types. At the leaf level, it has 200 ENE types. We aim to create an ENE dictionary that covers all these 200 types. In our approach, using Wikipedia as a basic resource, we classify Wikipedia titles into ENE types to create an ENE dictionary. We perform supervised learning; we derive a large number of features for Wikipedia titles and train a multiclass classifier. The features encode various aspects of Wikipedia titles, including those related to the surface string of a title, the content of an article, and the meta data, such as categories and infoboxes. We devise an extensive list of features for accurate classification into a large number of ENE types. The idea of using Wikipedia for creating an NE dictionary is not new; however, no work has sought to use such a large number of NE types. We want to verify whether it is possible to create an NE dictionary covering such a large number of NE types. We also want to know what types of features are useful in classifying entities into fine-grained ENE types. Note here that this work uses Japanese Wikipedia. Although we want to make our features as language-independent as possible, we introduce some possibly language-specific features for better accuracy.

In the next section, we describe related work. Section 3 describes our proposed method and shows the complete list of features used for training a classifier. Section 4 describes our experiments to verify our proposed method, the effect of the size of training data, and how to refine the acquired dictionary by using probability estimates of the learned classifier. Section 5 summarizes and mentions future work.

2 Related Work

The work to create NE dictionaries has centered around Wikipedia, with the focus on classifying Wikipedia titles (articles) into NE types. As far as we know, the earliest work is by Toral and Muoz (2006). They focused on the first sentence of a Wikipedia article (generally a definition statement) and counted the number of nouns related to three NE types, namely, Location, Organization, and Person, and applied heuristic rules that take into account the numbers in order to determine the article's NE type. Bhole et al. (2007) followed and proposed a super-

vised machine learning approach to the same task, involving the training of an SVM classifier. The features used were the bag-of-words of Wikipedia articles.

In addition to the texts of articles, there are rich meta data in Wikipedia, which can be helpful in distinguishing the NE types of articles. In addition to the nouns of first sentences, Nothman et al. (2008) used head nouns of categories assigned to articles and used heuristic rules to map them to one of four NE types; namely, LOC, PER, ORG, and MISC. Dakka and Cucerzan (2008) trained an SVM classifier by using features related to the structure of Wikipedia articles. Specifically, they introduced bag-of-words features of abstracts (Wikipedia provides short abstracts for articles), tables (infoboxes and contents boxes), and links to other articles. As NE types, they used five: LOC, PER, ORG, MISC, and COMM (common object). Saleh et al. (2010) also used features derived from abstracts, infoboxes, and categories to train their SVM classifier. A similar feature set was also used by (Tardif et al., 2009) who worked on six NE types. Richman and Schone (2008) exploited the multilingual nature of Wikipedia. By using links to articles in other languages, they classified non-English articles into NE types by using their English counterparts, whose NE types can be estimated by using their category information. They used four NE types: DATE, GPE (geographical and political entity), ORG, and PERSON.

The above studies used a relatively small number of NE types, but there are also studies that aimed to cover a larger number of NE types. Chang et al. (2009) used nine NE types (person, act, communication, location, animal, artifact, time, object, and group), which are a subset of supersenses defined in WordNet. To cope with the larger number of NE types, they introduced new features, such as the supersenses of the head nouns of first sentences and the synsets of category names, for training their maximum entropy classifier. Watanabe et al. (2007) worked on 13 NE types taken selectively from the ENE hierarchy. They had to avoid using the full 200 ENE types because of a sparseness problem for their graph-based algorithm. Tkatchenko et al. (2011) used 18 NE types taken from the BBN's question answering taxonomy (Brunstein, 2002). The features used for their SVM and naive Bayes classifiers were conceptually identical to those in (Tardif et al., 2009).

Our work is similar to the previous literature in that we use Wikipedia to create an NE dictionary, but different in that we aim to deal with a much larger number of NE types: 200 ENE types. We want to verify the feasibility of creating such a fine-grained NE dictionary and want to explore useful features for the classification into ENE types. To this end, we make an extensive list of features, adopting those previously proposed and also proposing new ones, for training our classifier. We describe our features later in Section 3.2.

Although not directly related to creating an NE dictionary, there is a good body of work that aims at constructing an ontology (a hierarchy of words or concepts connected with relations such as *is-a* and *has-a*) from Wikipedia (Ponzetto and Strube, 2007; Suchanek et al., 2008; Nagata et al., 2010). Since ontologies are useful resources for the deep processing of texts, such as the inference, once we have created our ENE dictionary, our next step would include constructing an ontology by relating the ENEs.

3 Proposed Method

Following the previous studies, we also propose to use supervised learning and learn a classifier that classifies Wikipedia titles (articles) into ENEs. We have three steps: the creation of training data, extraction of features, and training of a classifier. Since we deal with many NE types (200 ENE types), we place a special emphasis on the extraction of features for accurately distinguishing the ENE types. In what follows, we describe each step in detail.

3.1 Creating training data

As our training data, we need Wikipedia titles tagged with ENE types. However, since there are many ENE types and we need a reasonable amount of training data for each ENE type to avoid data sparseness, the manual creation of training data will be very costly.

Therefore, to facilitate the creation of training data, our basic idea is to turn to an existing corpus annotated with ENE types. In Japanese, there is one such corpus publicly available (Hashimoto et al., 2008). The corpus contains newswire articles in which entities are annotated with ENE tags. From such a corpus, we can extract entities together with their ENE tags to create what we call a **seed dictionary**, whose entries can be matched against Wikipedia titles to take an intersection so that Wikipedia titles can be automatically annotated with their ENE types. Of course, when there are other corpora or gazetteers available, they can also be exploited to augment the seed dictionary. The approach we employ here is similar to (Bhole et al., 2007) and (Zhang and Iria, 2009) in that entries of external dictionaries/gazetteers are intersected with Wikipedia titles to create training data. The difference is that we use an annotated corpus to create such entries. Below, we enumerate the steps we performed to create our seed dictionary and training data.

1. From the Hashimoto corpus (Hashimoto et al., 2008), we extracted all tagged sections. There are 8828 newswire articles (Mainichi Shimbun newspaper '95) in the corpus with 255407 tagged sections. Since some tags are not related to ENE types, we first discarded such tagged sections. We also discarded entities that were annotated with multiple ENE tags because such entities can be ambiguous and would introduce noise. For example, the entity "Rakuten" is a company but also a sports organization owned by the same company; hence it is given Company in one context and Pro_Sports_Organization in the other. By retaining only the unambiguous entities, we obtained 59318 unique entities with their ENE tags.
2. We performed the same procedure as above on our in-house corpus, which we have been maintaining and consists of newswire articles and blogs annotated with ENE types. There are 7184 documents with 118051 ENE tagged sections, from which we extracted 40231 unique entities with their ENE tags.
3. We maintain gazetteers of ENEs for evaluating our ENE recognizer. From the gazetteers, we first removed ambiguous entities (some entities are listed in multiple gazetteers) and then extracted 35858 unique entities with their ENE tags.
4. We merged the results of steps 1–3 to create a seed dictionary. After removing overlaps and ambiguous entities (NB. different tags can be given to the same entity depending on the source of the entity), we created a seed dictionary that contains 128213 unique entities with their ENE tags.
5. The entries of the seed dictionary were matched against Wikipedia titles to take an intersection. We used the Japanese Wikipedia dump of 2012-08-06, which contains 1411994 titles (we use this dump of Wikipedia throughout the paper). The titles here contain all entries including redirects, categories, and disambiguation pages. We found 51576 Wikipedia titles whose surface forms exactly matched the entries in the seed dictionary. We then removed, matched titles that consisted only of a single character, bare numbers, or two or less Hiragana/Katakana (Japanese phonologic characters) because they are potentially vague; their ENE tags could have been incidentally unique due to the lack of instances in the corpus or gazetteer; e.g., bare numbers can be of any ENE type concern-

Rank	ENE type	Count	Ratio	Accum
1	Person	9456	18.93%	18.93%
2	City	1897	3.80%	22.73%
3	Position_Vocation	1827	3.66%	26.38%
4	Product_Other	1805	3.61%	30.00%
5	Company	1725	3.45%	33.45%
6	Doctrine_Method_Other	1380	2.76%	36.21%
7	Date	1070	2.14%	38.35%
8	Dish	786	1.57%	39.93%
9	Book	736	1.47%	41.40%
10	Character	723	1.45%	42.85%
11	Broadcast_Program	618	1.24%	44.08%
12	School	601	1.20%	45.29%
13	Food_Other	600	1.20%	46.49%
14	Movie	594	1.19%	47.68%
15	GOE_Other	556	1.11%	48.79%
16	Show_Organization	554	1.11%	49.90%
17	Music	542	1.08%	50.98%
18	Corporation_Other	537	1.07%	52.06%
19	Station	458	0.92%	52.98%
20	Game	454	0.91%	53.89%

Table 1: Top 20 ENE types in our training data.

ing a number, such as Date, Age, or N_Person. This process left us with 49956 Wikipedia titles (3.54% of all titles), and this becomes our training data.

Our training data (49956 titles) cover 191 ENE types; unfortunately, we could not cover all 200 ENE types, because some ENE types, especially those related to numeric expressions, were scarcely seen in Wikipedia titles. The ones that could not be covered were Address_Other, Weight, Email, URL, Calorie, Intensity, Postal_Address, Seismic_Magnitude, and Volume. Table 1 shows the top-20 ENE types in the training data. We can see that, although Person is by far the most frequent, the decreasing pace of the frequencies of the subsequent types is slow. See Fig. 1 for the relationship between the ranks of ENE types and their counts. We can observe that, for a large proportion of ENE types, we have a reasonable number of training data, with a median around 100.

3.2 Feature Extraction

For each Wikipedia title in the training data, we extract features. We created an extensive list of features to cope with the many ENE types, covering many of the previously proposed features and also introducing new ones that we thought would be useful for distinguishing fine-grained types. Some features can be specific to Japanese. We propose to use 22 kinds of features in all. They are divided into three categories: surface string of a title, content of an article, and meta data of Wikipedia. It should be noted here that when a title has a redirect to another title, we extract features for both titles and merge them to create its features.

3.2.1 Features related to the surface string of the title of an article

The surface string of a title could be greatly indicative of its ENE type. For example, a title that ends with “shi (city)”, is likely to be the name of a city and therefore should be given

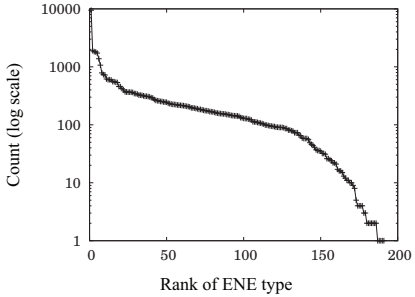


Figure 1: Relationship between the ranks of ENE types and their counts in the training data.

City as its ENE type. Rivers and Mountains, especially in Japanese, have names that end with "kawa (river)" or "yama (mountain)"; therefore, such names can be easily classified into River and Mountain ENE types. We have 16 features (T1–T16: 'T' stands for title) regarding various aspects of the surface string of a title. As far as we know, conventional studies have never put an emphasis on the surface string of a title, probably because most previous work worked with the English language. One exception is (Tardif et al., 2009) that used bag-of-unigrams in the title, corresponding to our T1. T2 through T16 are our newly introduced features. We enumerate the features below.

(T1-T2) Word unigram/bigram We first run the title in question through a morphological analyzer and separate it into words. Note that there are no marked word boundaries in Japanese. Then, we extract word unigrams and bigrams as features. Here, the features are bag-of-words features, indicating the existence of particular unigrams or bigrams with a binary value (i.e., 1 or 0). As a morphological analyzer, we use JTAG (Fuchi and Takagi, 1998).

(T3-T4) Character unigram/bigram We split a title into character tokens and then create bag-of-words features of character unigrams and bigrams. We have this feature since characters, especially Kanji (Chinese-origin) characters in Japanese, have individual meanings even when they form part of a word.

(T5-T6) POS unigram/bigram Using the results of the morphological analysis, we create bag-of-words features for the unigrams and bigrams of part-of-speech (POS) tags. Japanese POS taggers, including JTAG, generally output POS tags that correspond to subcategories for proper nouns, which can be useful in distinguishing ENE types. Other POS information, such as the existence of numbers, could also be useful.

(T7) Last common noun or noun/counter suffix We create a feature from the last word of a title whose POS tag is either a common noun or noun/counter suffix. This is a binary feature, indicating the existence of such a word. The rationale for using the last word of a title is that it is usually the head in Japanese. We chose to use common nouns and noun/counter suffixes because they denote general conceptions or entity categories in Japanese, and are hence more suitable than other POS tags. Here, the idea of using common nouns is similar to using the plural form of nouns in English (Suchanek et al., 2008).

(T8) Semantic category of the last common noun or noun/counter suffix JTAG outputs semantic categories for words. Here the semantic categories are those defined in the Japanese Goi-Taikei ontology (Ikehara et al., 1997). There are 2715 semantic categories in all. We extract features that indicate the existence of semantic categories for the last common noun or noun/counter suffix. It is natural to use these categories because they can be directly indicative of certain ENE types. The semantic categories can also provide abstract meanings for some words, which can be useful when there is sparseness in training data.

(T9-T11) Last one/two/three character(s) We have three features that use the last characters of a title because some characters alone could indicate certain ENE types in Japanese, especially when they are found at the end of a word. For example, words "ninja" and "geisha" both end with the Kanji character "sha", which by itself indicates a person. We use the existence of the last characters (one to three characters) of a title as features. We decided to use up to three characters to increase the coverage of subwords.

(T12) Semantic categories We extract semantic categories for all words in the title by using JTAG and create bag-of-words features. We use this feature to compensate for any possible lack of information that arises from using only the semantic categories of the last common noun or noun/counter suffix.

(T13) Proper noun semantic categories JTAG outputs special semantic categories for proper nouns. They are defined in the Goi-Taikei ontology and there are 130 such categories. Since proper nouns are conceptually similar to NEs, these categories will be useful for the classification of ENE types. We extract proper noun semantic categories for all words in a title and create their bag-of-words features.

(T14) IREX-based NEs We run an off-the-shelf NE recognizer, NameLister (Saito and Nagata, 2003; Suzuki et al., 2006), and extract IREX-based NEs in the title, from which we create bag-of-words features, indicating the existence of each of eight NE types in IREX. Here, the granularity of NEs is coarse; we regard this feature as a complement to other semantic category related features.

(T15) Last character type Japanese has several characters types, and certain types can be indicative of certain NE types. For example, Katakana characters are likely to be used for entities of a foreign origin such as cars and products, whereas Hiragana characters are likely to be used for Japanese entities. Here, we distinguish five types of characters: Hiragana, Katakana, Kanji, Alphabet, and Other, and use the type of the last character of a title as a feature.

(T16) Character type construction In addition to the last character type, this feature focuses on how character types constitute a title. We first split the title into character tokens and converted them into character types. Then, we concatenated the character types as a single string that represents the title's character type construction. For example, "London" is written with four consecutive Katakana characters in Japanese. Therefore, we have "K-K-K-K" (K stands for Katakana) as a binary feature.

3.2.2 Features related to the content of an article

The content of the article of a title obviously has important information about the entity. In this paper, we use two features (C17–C18, where 'C' stands for content) about the content of an article. We do not use all the words of an article because it would make the feature space too sparse and could cause over-fitting to the training data. We focus on the representative

parts of an article; namely, the first sentence and headings (section titles). These features have been used in previous studies (Toral and Muoz, 2006; Dakka and Cucerzan, 2008; Tkatchenko et al., 2011).

(C17) Last common noun or noun/counter suffix of the first sentence It is widely known that the first sentence of an article in Wikipedia is a definition statement. To obtain the first sentence of an article, we first obtain the abstract text of the article from the abstract data provided with a Wikipedia dump and then select its first sentence by selecting a text span from the beginning to the first punctuation mark. Here, we use the abstract data to facilitate our extraction of the initial part of an article. Then, we analyze the sentence with JTAG to extract the last common noun or noun/counter suffix in the first sentence. Finally, we create a binary feature indicating the existence of the last common noun or noun/counter suffix.

(C18) Headings Headings or section titles summarize what is written in an article. For example, the article of "Nobunaga Oda", a famous warlord in Japan, has section titles such as life, personality, portrait, and policies, which clearly indicate that this article is about a person. Therefore, we create bag-of-words features of section titles, each of which indicates the existence of a particular section title. Here, we only use top-level section titles. We ignore such section titles as "links to other articles" and "references" because they can be found in arbitrary articles and therefore would not be useful for distinguishing NE types. We extract section titles from the XML dump of Wikipedia by locating texts enclosed by "==" (section title markers in the Wiki format).

3.2.3 Features related to the meta data of an article

Ever since the work of Nothman et al. (2008), meta data in Wikipedia have been vigorously used for the classification of articles into NE types and have proved their usefulness. In this paper, we extract four features (M19–M22: 'M' stands for meta data) regarding the meta data of an article. Here, M20 and M22 are our newly introduced features.

(M19) Direct categories Categories are one of the most widely used features for distinguishing NE types. We analyze each category assigned to an article with JTAG and extract the last common noun or noun/counter suffix to create a bag-of-words feature. This is similar to using head nouns of categories (Nothman et al., 2008).

(M20) Upper categories In Wikipedia, categories have a network (mostly hierarchical) structure where articles are the sub-nodes of the categories. Starting from the article in question, we find the shortest path to the root category ("Main Category") and use the categories on the path. Such categories can be regarded as the upper categories for the article, and can be useful for distinguishing NE types especially when the direct categories are too specific. For each category on the path (except for the root), we extract the last common noun or noun/counter suffix to create a bag-of-words feature. We use Wik-IE (Mori et al., 2009) to create the network structure, which calculates the distance between nodes in Wikipedia using the Dijkstra's algorithm.

(M21) Infobox attributes Infoboxes provide tabular data for articles. Since the attributes (attribute names) of a table generally indicate the attributes of the entity in question, in a similar manner to (Saleh et al., 2010), we use the infobox attributes to create our features. For each attribute in the infobox, we create a bag-of-words feature indicating the existence of that attribute. We used the Japanese version of DBPedia (http://ja.dbpedia.org/), which offers the infobox data for Japanese articles as triples.

(M22) Instance types Inspired by Richman and Schone (2008), who used the English version of Wikipedia for non-English articles, we also turn to the English version for useful information. We noticed that the English version is heavily linked with the English version of DBPedia (http://en.wikipedia.org/wiki/DBpedia), which offers instance types for English articles. The instance types are given in the vocabulary of various ontologies, such as the DBPedia ontology, schema.org, and Friend of a Friend (FOAF), and are likely to be helpful in distinguishing NE types. For example, "Paris" has Place, PopulatedPlace, Settlement, and Thing as instance types. To create this feature, we first search for the English version of an article by using the external language links and find its instance types by looking up DBPedia. Then, for each instance type, we create a bag-of-words feature. Note that Japanese DBPedia currently does not provide instance type data.

3.3 Training a classifier

For each title in the training data, we extract the features and train a multiclass classifier that classifies the title into one of the ENE types. Here, we employ logistic regression as a learning algorithm. We first create binary logistic regression classifiers for all ENE types. Then, in classifying a title, we find the classifier that outputs the best probability estimate for that title, and use the ENE type for that classifier as an output ENE type.

Although previous studies have extensively used SVMs for NE type classification, we chose logistic regression because of its capability to output probability estimates. Such estimates can be used as confidence scores and can be used to refine the classification results by discarding low-confidence entries with a threshold (See Section 4.4). Note that the scores output by SVMs cannot be directly used as confidence scores because they denote distances from hyperplanes whose absolute values cannot be compared with a fixed threshold. For training the classifiers and calculating probability estimates, we use the LIBLINEAR toolkit (http://www.csie.ntu.edu.tw/~cjlin/liblinear/).

4 Experiments

We performed a series of experiments to verify our method. First of all, we created two sets of data by dividing our training data into two: one for training a classifier (TRAIN SET) and the other for testing (TEST SET). From the 49956 titles in our training data, we randomly sampled 10000 titles as TEST SET and made the rest TRAIN SET (39956 titles). TRAIN SET and TEST SET cover 190 and 179 ENE types, respectively.

We performed four experiments. The first experiment examined the classification accuracy when only one of the features is employed. The purpose of this experiment is to verify the effectiveness of each feature. Then, in the second experiment, we combined the features to maximize the classification accuracy. In the third experiment, we examined how the classification accuracy is affected by the size of training data. In the last experiment, we examined how the obtained ENE dictionary can be refined by discarding low-confidence entries. As an evaluation metric, we mainly use the classification accuracy, which is the rate of accurately classified ENE types in TEST SET. At the end of this section, we also describe our final ENE dictionary that we created by applying the trained classier to all titles in Wikipedia.

4.1 Results by individual features

Table 2 shows the classification accuracy when the classifiers were trained by using one of the features. Since some features can only be extracted for certain titles and since it is impossible

Feature		Acc	Rise in Acc
(T1)	Word unigram (BASE)	50.63%	0.00%
(M19)	Direct categories	72.54%	21.91%
(C17)	First sentence common noun	65.95%	15.32%
(T12)	Semantic categories	63.51%	12.88%
(M20)	Upper categories	63.40%	12.77%
(M21)	Infobox attributes	62.28%	11.65%
(C18)	Headings	62.08%	11.45%
(T4)	Character bigram	61.38%	10.75%
(M22)	Instance types	59.12%	8.49%
(T8)	Semantic category of last common noun	58.78%	8.15%
(T3)	Character unigram	58.39%	7.76%
(T9)	Last character	57.75%	7.12%
(T6)	POS bigram	56.99%	6.36%
(T5)	POS unigram	56.03%	5.40%
(T10)	Last two characters	55.37%	4.74%
(T13)	Proper noun semantic categories	54.06%	3.43%
(T14)	IREX NEs	53.37%	2.74%
(T11)	Last three characters	52.39%	1.76%
(T16)	Character type construction	51.94%	1.31%
(T15)	Last character type	51.59%	0.96%
(T2)	Word bigram	51.21%	0.58%
(T7)	Last common noun	50.94%	0.31%

Table 2: Classification accuracy for the individual features. All features are sorted by their performance except for the word unigram feature, which is used as a base feature.

to learn classifiers without a feature, we trained classifiers by using each of the features with an obligatory use of the word unigram as a base feature. As can be seen in the table, all features contribute to the improvement in the classification accuracy. The features related to the content and meta data, which proved their usefulness for a smaller number of NE types, also proved their effectiveness for ENE types. The direct category feature was by far the most effective. We have two title-related features, T12 and T4, that contributed to the rise of more than 10% over the base feature, suggesting the usefulness of using the title information for the classification into ENEs. We find it interesting that character bigrams contribute greatly to the classification accuracy, suggesting the importance of characters/subwords in Japanese.

4.2 Results by the combination of features

We use a backward selection method to find the best combination of features; that is, we first use all the features and then remove one feature at a time whose removal improves the performance the most. As a result, we removed seven features: (T3) Character unigram, (T4) Character bigram, (T5) POS unigram, (T10) Last two characters, (T13) Proper noun semantic categories, (T16) Character type construction, and (T7) Infobox attributes.

The feature selection improved the classification accuracy from 79.10% to 79.63%, the best accuracy we attained in this work. See Table 3 for the results. We believe the classification accuracy of 79.63% is reasonably high when considering the large number of ENE types. The table also shows the drop in performance when other features are individually eliminated from the best combination, which shows that even when we remove the most influential feature, i.e.,

Feature		Acc	Drop in Acc
	Best combination (w/o T3–T5, T10, T13, T16, M21)	**79.63%**	0.00%
(T11)	w/o Last three characters	79.50%	-0.13%
(T7)	w/o Last common noun	79.49%	-0.14%
(T14)	w/o IREX NEs	79.44%	-0.19%
(T8)	w/o Semantic category of last common noun	79.44%	-0.19%
(T2)	w/o Word bigram	79.38%	-0.25%
(T12)	w/o Semantic categories	79.35%	-0.28%
(T6)	w/o POS bigram	79.35%	-0.28%
(C18)	w/o Headings	79.33%	-0.30%
(T15)	w/o Last character type	79.31%	-0.32%
(M22)	w/o Instance types	79.29%	-0.34%
(T1)	w/o Word unigram	79.29%	-0.34%
(M20)	w/o Upper categories	79.23%	-0.40%
(T9)	w/o Last character	79.02%	-0.61%
(C17)	w/o First sentence common noun	78.93%	-0.70%
(M19)	w/o Direct categories	77.83%	-1.80%

Table 3: Classification accuracy for the best combination and when one of the features is removed from the best combination. The features are sorted by their performance.

Feature set	Acc
T (T1–T16)	69.37%
C (C17, C18)	68.57%
M (M19–M22)	75.40%
T+C	76.14%
T+M	78.64%
C+M	75.90%
T+C+M	**79.10%**
Previous work only (T1, C17, C18, M19, M21)	75.11%
Newly introduced only (T2–T16, M20, M22)	75.09%

Table 4: Classification accuracy for the combinations of feature sets. When the word unigram (T1) is not included in a feature set, we had it included as a base feature.

(M19) Direct categories, from the best combination, the drop is small (1.80%). This indicates that most of the features possess complementary information. The magnitude of a drop can be considered as the specific information carried by a certain feature. We consider it interesting that (T9) Last character has the third largest drop, indicating again the effectiveness of using characters in Japanese.

Table 4 shows the classification accuracy for some combinations of feature sets. We can see that, when we use the features of all three categories (T+C+M), the best accuracy is achieved. The contribution of the newly introduced features is clear, raising the accuracy from 75.11% to 79.10%.

Table 5 shows the precision, recall, and F-measure for the most frequent 20 ENE types in TEST SET (the feature set used is the best combination in Table 3). We can see that the performance varies greatly depending on the ENE types. For some ENE types, such as Product_Other, Doctrine_Method_Other, Character, and GOE_Other, our method performs very poorly. We suspect

Rank	ENE type	Precision		Recall		F
1	Person	0.851	(1840/2162)	0.952	(1840/1933)	0.899
2	City	0.859	(371/432)	0.907	(371/409)	0.882
3	Product_Other	0.525	(255/486)	0.646	(255/395)	0.579
4	Company	0.756	(301/398)	0.820	(301/367)	0.787
5	Position_Vocation	0.766	(301/393)	0.875	(301/344)	0.817
6	Doctrine_Method_Other	0.501	(177/353)	0.697	(177/254)	0.583
7	Date	0.887	(197/222)	0.956	(197/206)	0.921
8	Dish	0.809	(127/157)	0.789	(127/161)	0.799
9	Book	0.723	(102/141)	0.646	(102/158)	0.682
10	School	0.948	(127/134)	0.907	(127/140)	0.927
11	Character	0.639	(62/97)	0.449	(62/138)	0.528
12	Broadcast_Program	0.735	(83/113)	0.654	(83/127)	0.692
13	Food_Other	0.644	(76/118)	0.650	(76/117)	0.647
14	Show_Organization	0.876	(85/97)	0.746	(85/114)	0.806
15	GOE_Other	0.653	(49/75)	0.450	(49/109)	0.533
16	Movie	0.745	(70/94)	0.648	(70/108)	0.693
17	Music	0.772	(71/92)	0.676	(71/105)	0.721
18	Station	0.942	(81/86)	0.880	(81/92)	0.910
19	Corporation_Other	0.673	(72/107)	0.791	(72/91)	0.727
20	Game	0.892	(74/83)	0.831	(74/89)	0.860

Table 5: Precision, recall, and F-measure for the most frequent 20 ENE types in TEST SET.

this low performance is partly attributable to the variation of their names. We leave it to our future work to improve the accuracy for these hard-to-guess ENE types.

4.3 Learning curve

We split TRAIN SET into blocks of 1000 (1K) titles, and examined how the performance improves when each block is added to the training data. TEST SET was used for testing. Figure 2 shows the learning curve. We can see a steady improvement until we reach around 10K. The classification accuracy at 10K is 75.07%. However, after that, the pace of improvement is very gradual. From 30K on, the gradient is just 0.14% per K. Even were this gradient to continue, to reach 100%, we would need an additional 145K of training data, which would be too hard to create manually. For further improvement, it would definitely be necessary to devise new useful features. In addition, it would be helpful to bring in external gazetteers and link them with ENE types so that they can be used to augment our seed dictionary. The idea of linking with other gazetteers has been successful in the linked data community (http://linkeddata.org/), which makes this approach promising. We would like to pursue this approach in the future.

4.4 Refinement by using probability estimates

Our choice of logistic regression was motivated by its capability to output probability estimates so that low-confidence results can be discarded. We examined how this discarding process improves the accuracy. We set up a variable t for a cut-off threshold and moved it from 0 to 1 by increasing it by 0.05. When t becomes larger, erroneous (low-confidence) results would be discarded, and the classification accuracy would improve. However, the improvement in accuracy would come at the cost of decreased coverage (the number of retained samples over the total number of samples). Figure 3 shows how the accuracy and coverage change as

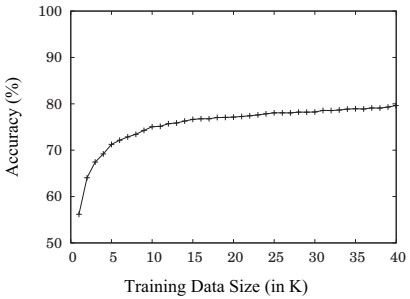


Figure 2: Learning Curve.

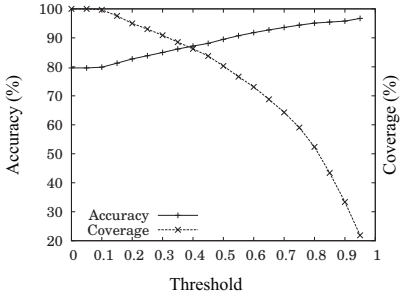


Figure 3: Accuracy and coverage by a varying threshold.

t increases. One can see that the improvement of accuracy is gradual between 80% to 95%, whereas that of the coverage is rather sharp. We also see a clear trade-off between the accuracy and coverage. We use this finding in the creation of our ENE dictionary in the next section.

4.5 Creating the final ENE dictionary

We apply our ENE classifier to all titles in Wikipedia in order to create our ENE dictionary. For this purpose, we newly trained an ENE type classifier using all of our training data (49956 titles). Then, we extracted the features (best combination in Table 3) for all Wikipedia titles (1411994 titles). Using the extracted features, we classified the titles into ENE types with their probability estimates. The result becomes our initial ENE dictionary.

As a dictionary resource, a weight should be given to accuracy over coverage. According to Fig. 3, when t is 0.5, about 90% accuracy (89.48%) can be achieved with over 80% coverage (80.29%). Since we viewed this as a fair setting, satisfying the requirements of a good dictionary, we decided to use 0.5 for t for refinement. After the refinement, 1015563 (71.92%) entries remained. Althouth the number of retrained entries were slightly smaller than the expected 80.29%, we could still retain over one million entries. Here, from Fig. 3, the estimated accuracy of the entries is 89.48%. The final dictionary covers 182 ENE types.

Table 6 shows the top 20 ENE types in the created ENE dictionary. Although Person occupies a good proportion of the entries, we also see large numbers of entities of other ENE types. Compared to the statistics of the training data (Table 1), the orders of the ENE types had some changes, reflecting the differences between the ENE corpus and Wikipedia. Although we cannot show the relationship between the ranks of ENE types and their counts by a figure similar to Fig. 1 for lack of space, the shape of the curve is nearly identical. Up to the 148th rank, we have over 100 entries, suggesting that we have a reasonable number of entities even when the ENE type is rather rare. A fine-grained ENE dictionary such as this one would be greatly helpful for various information extraction tasks.

5 Summary and Future Work

Using Wikipedia as a basic resource, we classified Wikipedia titles into ENE types to create an ENE dictionary. We derived a large number of features for the titles and trained a multiclass classifier. The features used encode various aspects of Wikipedia titles, such as those related

Rank	ENE type	Count	Ratio	Accum
1	Person	301625	29.700%	29.700%
2	Product_Other	53751	5.293%	34.993%
3	School	42179	4.153%	39.146%
4	Company	36462	3.590%	42.737%
5	City	34139	3.362%	46.098%
6	Road	32252	3.176%	49.274%
7	Music	29262	2.881%	52.155%
8	Position_Vocation	27712	2.729%	54.884%
9	Broadcast_Program	26759	2.635%	57.519%
10	Doctrine_Method_Other	24616	2.424%	59.943%
11	Station	21408	2.108%	62.051%
12	Book	15638	1.540%	63.591%
13	Game	14033	1.382%	64.972%
14	Movie	12956	1.276%	66.248%
15	Show_Organization	11778	1.160%	67.408%
16	Worship_Place	11076	1.091%	68.499%
17	Train	11014	1.085%	69.583%
18	Date	10624	1.046%	70.629%
19	Province	10213	1.006%	71.635%
20	GOE_Other	10190	1.003%	72.638%

Table 6: Top 20 ENE types in the created ENE dictionary.

to the surface string of a title, the content of the article, and the meta data provided with Wikipedia. By experiments, we successfully showed that it is possible to classify Wikipedia titles into ENE types with a reasonable accuracy of 79.63%. We also showed that, by applying the classifier to all Wikipedia titles and by discarding low-confidence entries, it is possible to create an ENE dictionary of over one million entities covering 182 ENE types with an estimated accuracy of 89.48%. Our main contributions are as follows: (1) We created the first large-scale ENE dictionary; no work has attempted to classify entities into such fine-grained types. (2) We made clear the features that are useful for the classification into ENE types; we ascertained the effectiveness of the previously proposed features regarding the content and meta data and newly found useful title-related features, such as the semantic categories found in the title and character bigrams.

Our method has a number of limitations. First, the classification accuracy is still low. As shown in Table 5, the accuracy for some ENE types are very poor. We want to devise new features and also find ways to augment our training data. Second, some features, e.g., character-based features, could be dependent on the Japanese language. We need to examine whether our method is applicable to other languages, especially English. Third, we could not cover all ENE types in our dictionary; many ENE types especially those related to numerical expressions were not included in the dictionary. This is mainly because Wikipedia titles do not cover such expressions. We want to investigate other resources to enrich our dictionary with currently scarce ENE types. Fourth, we want to evaluate our dictionary extrinsically, for example by using it in such information extraction tasks as question answering. We also want to use the dictionary to derive gazetteer features for our ENE recognizer that is under development. Finally, we want to extend our dictionary to ontologies so that it can be used for more intelligent tasks. Since ENE types themselves form a hierarchy, we will be able to use this structure to relate the ENEs in our created dictionary.

References

Bhole, A., Fortuna, B., Grobelnik, M., and Mladenic, D. (2007). Extracting named entities and relating them over time based on Wikipedia. *Informatica (Slovenia)*, 31(4):463–468.

Brunstein, A. (2002). Annotation guidelines for answer types. LDC2005T33, Linguistic Data Consortium.

Chang, J., Tsai, R. T.-H., and Chang, J. S. (2009). Wikisense: Supersense tagging of Wikipedia named entities based WordNet. In *Proc. PACLIC*, pages 72–81.

Dakka, W. and Cucerzan, S. (2008). Augmenting Wikipedia with named entity tags. In *Proc. IJCNLP*, pages 545–552.

Fuchi, T. and Takagi, S. (1998). Japanese morphological analyzer using word co-occurrence: JTAG. In *Proc. COLING-ACL*, pages 409–413.

Grishman, R. and Sundheim, B. (1996). Message understanding conference - 6: A brief history. In *Proc. COLING*, pages 466–471.

Hashimoto, T., Inui, T., and Murakami, K. (2008). Constructing extended named entity annotated corpora. In *SIG-NL-188, Information Processing Society of Japan*, pages 113–120. (in Japanese).

Ikehara, S., Miyazaki, M., Shirai, S., Yokoo, A., Nakaiwa, H., Ogura, K., Ooyama, Y., and Hayashi, Y. (1997). *Goi-Taikei – A Japanese Lexicon*. Iwanami Shoten.

Isozaki, H. and Kazawa, H. (2002). Efficient support vector classifiers for named entity recognition. In *Proc. COLING*, pages 1–7.

Kazama, J. and Torisawa, K. (2008). Inducing gazetteers for named entity recognition by large-scale clustering of dependency relations. In *Proc. ACL-HLT*, pages 407–415.

Mori, T., Masuda, H., Kiyota, Y., and Nakagawa, H. (2009). Wik-IE: A tool to extract structure of Wikipedia entries. In *SIG-SWO-A803-02, The Japanese Society for Artificial Intelligence*. (in Japanese).

Nagata, M., Shibaki, Y., and Yamamoto, K. (2010). Using Goi-Taikei as an upper ontology to build a large-scale japanese ontology from Wikipedia. In *Proc. the 6th Workshop on Ontologies and Lexical Resources*, pages 11–18.

Nothman, J., Curran, J. R., and Murphy, T. (2008). Transforming Wikipedia into named entity training data. In *Proc. the Australasian Language Technology Association Workshop 2008*, pages 124–132.

Ponzetto, S. P. and Strube, M. (2007). Deriving a large scale taxonomy from Wikipedia. In *Proc. AAAI*, pages 1440–1445.

Richman, A. E. and Schone, P. (2008). Mining Wiki resources for multilingual named entity recognition. In *Proc. ACL-HLT*, pages 1–9.

Saito, K. and Nagata, M. (2003). Multi-language named-entity recognition system based on HMM. In *Proc. the ACL 2003 Workshop on Multilingual and Mixed-language Named Entity Recognition*, pages 41–48.

Saleh, I., Darwish, K., and Fahmy, A. (2010). Classifying Wikipedia articles into NE's using SVM's with threshold adjustment. In *Proc. the 2010 Named Entities Workshop*, pages 85–92.

Sekine, S. and Isahara, H. (2000). IREX: IR & IE evaluation project in Japanese. In *Proc. LREC*.

Sekine, S. and Nobata, C. (2004). Definition, dictionaries and tagger for extended named entity hierarchy. In *Proc. LREC*.

Sekine, S., Sudo, K., and Nobata, C. (2002). Extended named entity hierarchy. In *Proc. LREC*.

Suchanek, F. M., Kasneci, G., and Weikum, G. (2008). YAGO: A large ontology from Wikipedia and WordNet. *Web Semantics: Science, Services and Agents on the World Wide Web*, 6(3):203–217.

Suzuki, J., McDermott, E., and Isozaki, H. (2006). Training conditional random fields with multivariate evaluation measures. In *Proc. COLING-ACL*, pages 217–224.

Tardif, S., Curran, J. R., and Murphy, T. (2009). Improved text categorisation for Wikipedia named entities. In *Proc. the Australasian Language Technology Association Workshop 2009*, pages 104–108.

Tkatchenko, M., Ulanov, A., and Simanovsky, A. (2011). Classifying Wikipedia entities into fine-grained classes. In *Proc. the 2011 IEEE 27th International Conference on Data Engineering Workshops*, pages 212–217.

Toral, A. and Muoz, R. (2006). Proposal to automatically build and maintain gazetteers for named entity recognition by using Wikipedia. In *Proc. the EACL 2006 workshop on NEW TEXT Wikis and blogs and other dynamic text sources*, pages 56–61.

Voorhees, E. M. and Dang, H. T. (2005). Overview of the TREC 2005 question answering track. In *Proc. TREC*.

Watanabe, Y., Asahara, M., and Matsumoto, Y. (2007). A graph-based approach to named entity categorization in Wikipedia using conditional random fields. In *Proc. EMNLP-CoNLL*, pages 649–657.

Zhang, Z. and Iria, J. (2009). A novel approach to automatic gazetteer generation using Wikipedia. In *Proc. the 2009 Workshop on The People's Web Meets NLP: Collaboratively Constructed Semantic Resources*, pages 1–9.

Statistical Method of Building Dialect Language Models for ASR Systems

Naoki HIRAYAMA[1] *Shinsuke MORI*[1,2] *Hiroshi G. OKUNO*[1]
(1) Graduate School of Informatics, Kyoto University
(2) Academic Center for Computing and Media Studies, Kyoto University
Yoshida-honmachi, Sakyo-ku, Kyoto, Japan
hirayama@kuis.kyoto-u.ac.jp, forest@i.kyoto-u.ac.jp, okuno@i.kyoto-u.ac.jp

ABSTRACT
This paper develops a new statistical method of building language models (LMs) of Japanese dialects for automatic speech recognition (ASR). One possible application is to recognize a variety of utterances in our daily lives. The most crucial problem in training language models for dialects is the shortage of linguistic corpora in dialects. Our solution is to transform linguistic corpora into dialects at a level of pronunciations of words. We develop phoneme-sequence transducers based on weighted finite-state transducers (WFSTs). Each word in common language (CL) corpora is automatically labelled as dialect word pronunciations. For example, *anta* (Kansai dialect) is labelled *anata* (the most common representation of 'you' in Japanese). Phoneme-sequence transducers are trained from parallel corpora of a dialect and CL. We evaluate the word recognition accuracy of our ASR system. Our method outperforms the ASR system with LMs trained from untransformed corpora in written language by 9.9 points.

KEYWORDS: spoken language, dialect, language model, weighted finite-state transducer (WFST).

Proceedings of COLING 2012: Technical Papers, pages 1179–1194,
COLING 2012, Mumbai, December 2012.

1 Introduction

Automatic speech recognition (ASR) systems for spoken language are yet far from practical use. ASR systems for written sentences have been widely studied, and recognition accuracy has rapidly improved. In contrast, recognition accuracy is drastically lower for spontaneous speech (Anusuya and Katti, 2009, p. 194). People in their daily lives do not actually speak in a stable way like written sentences. Their speeches include casual expressions, fillers, and even vocabulary specific to dialects.

This paper especially handles improving Japanese dialect ASR. Most previous application systems with speech interface have assumed well-formed sentences in the common language (CL), although they have assumed non-expert speakers. Non-expert speakers will obviously utter informal expressions other than those in written language, and even words specific to their own dialect; dialect ASR systems have difficulty in recognition accuracy or scalability.

Dialects in the world have various kinds of differences (Benincà, 1989). The major differences between dialects and the CL are categorized into the following types: 1) pronunciation, 2) vocabulary, and 3) word order. The first type belongs to the difference in acoustic features, while the second and third belong to that in linguistic ones. Canadian English contains all of three types; 1) /tu/ is pronounced as /tju/, 2) 'high school' is called 'collegiate institute', and 3) 'next Tuesday' is changed into 'Tuesday next' (Woods, 1979). Many North American varieties of French have a tendency to take SVO (subject-verb-object) word order (Gadet and Jones, 2008). In Japanese dialects, the difference of vocabulary is characteristic, e.g., *tabe n* is used instead of *tabe nai* (do not eat) (Gottlieb, 2005).

Our method in this paper focuses on differences in pronunciation of vocabulary between dialects, which correspond to the first and second types. Vocabulary is a set of word entries used in a language or a dialect. We process pronunciation as the corresponding phoneme sequence to reduce the problem to text processing.

The main difficulty with dialect ASR lies in the shortage of linguistic corpora on dialects because they are spoken rather than written. This prevents us from building statistically reliable language models (LMs) including characteristic vocabulary for dialect ASR.

In this paper, we overcome the shortage of dialect corpora by training a vocabulary transformation system that gives labels of dialect expressions to each word in large CL linguistic corpora. (The LMs for ASR is trained based on the output of the above vocabulary transformation system.) The vocabulary transformation system is implemented as a weighted finite-state transducer (WFST) (Allauzen et al., 2007; Neubig et al., 2009). WFSTs model probabilistic transformation rules extracted from dialect-CL parallel corpora.

The three main advantages of our strategy are as follows. First, our system improves the recognition of dialect utterances even with a limited amount of dialect corpora. Second, our method dispenses with the manual enumeration of dialect transformation rules. Therefore, it enables us to build ASR systems for various dialects in the principled manner. Third, statistical corpus transformation gives a solution to how to choose one of multiple candidates for output by taking the contexts of parallel corpora into account.

This paper is organized as follows. Section 2 reviews related work on dialect ASR. Section 3 states major elements of our system, and describes our method of recognition of dialect utterances. Section 4 discusses our evaluation of the system in terms of word recognition accuracy, and finally conclusions summarize this paper and describe future work.

2 Related work

Most studies have focused on acoustic aspects in developing ASR systems for dialects, Ching et al. (1994) described the phonological and acoustic properties of Cantonese, one of the major dialects of Chinese language, based on energy profiles, pitch, and duration. Miller and Trischitta (1996) studied the phonetic features of Northern and Southern US dialects in linearly classifying each dialect. Their experiment achieved error rates of 8% in distinguishing Northern US dialect from those in the the South. Lyu et al. (2006) developed an ASR system for two Chinese dialects, Mandarin and Taiwanese. Dialect-mixed utterances could be recognized with combined character-to-pronunciation mapping in their system.

These systems had two main problems:

1. difficulty of collecting acoustic corpora of dialects
2. incapability of incorporating vocabulary difference

The first problem means that many dialect speakers are necessary for reliable analyses. These systems would work well for major dialects whose corpora were abundant, whereas it was not realistic to collect large corpora even for relatively minor dialects. The second problem prevented these systems from being in general use. Phonological methods are effective for the situation that variation of dialects mainly stems from differences of their phonemes, while these systems do not cover difference of vocabulary. If target dialects have large difference of vocabulary, these systems are less effective. The strategy of classifying dialects and selecting LMs is effective only if the vocabulary of target dialects are almost the same, but actually, dialect vocabulary is rather likely to differ between dialects (Wolfram, 2009, p. 144). The strategy of classifying dialects and next selecting LMs is possible, of course, but effective case is limited; classification would not work well if vocabulary dominates difference of target dialects and these dialects have similar phonological characteristics.

Instead of studying acoustic aspects of dialects containing the problems above, some studies focused linguistic aspects. Zhang (1998) described dialect machine translation (MT) between dialects in the Chinese language. Since dialect sentences were only represented in sound and had not been written down, his translations were between *pinyin* representations of two dialects, which is similar to those in our study. Munteanu et al. (2009), related to the correction of ASR results, tried to correct ASR results in the lecture domain by using a transformation model trained from correct sentences and the corresponding outputs. The scoring for each rule was based on how much the word error rate (WER) could be reduced by applying the rules. These studies still had problems. Zhang (1998) created translation dictionaries manually, and dealing with various dialects required the same process for each dialect. Munteanu et al. (2009) assumed that ASR results were correct; if much vocabulary specific to a target domain were not covered, e.g., for dialects, these methods would not work well. These problems indicates that the key to successful ASR systems is automatic building of LMs in dialects.

This paper deals with dialect ASR as follows. We develop a dialect ASR system by building LMs instead of analyzing acoustic features, because vocabulary is more characteristic in Japanese dialects, as mentioned in Section 1. This enables an ASR system to recognize vocabulary specific to each dialect. Translation dictionaries, transformation rules in other words, are automatically extracted from dialect-CL parallel corpora. The extracted rules are probabilistic based on the statistical analysis of parallel corpora; using large CL corpora, we can simulate dialect linguistic corpora including variations in word choices. This strategy is applied to transformations between spoken and written language (Akita and Kawahara, 2010; Neubig

et al., 2012). Since our transformation targeted dialects, it is more advanced than that for mere spoken language. Our transformation model is simpler than those in these studies, due to our assumption that the word order does not change.

3 ASR for Japanese Dialects

This section describes our method in detail. First, we enumerate elements of our system. Next, we explain how to develop the vocabulary transformation system based on WFST. Finally, we introduce examples of corpora to transform.

The inputs are utterances in dialects and the outputs are recognized word sequences in the CL. We make the following three assumptions behind the problem setting. First, dialects would have no effect on the word order; in other words, it would be only necessary to merely transform pronunciation. Second, dialects of input utterances are known and parallel corpora corresponding to the dialects are available. Third, one-to-many sentence correspondence for CL and dialect sentences, i.e., one CL sentence may be transformed into various dialect sentences by dialect speakers, while these dialect sentences have only one corresponding CL sentence. This problem setting has advantages that 1) we prefer that ASR systems output a CL sentence as its meaning instead of simple dialect transcription given a dialect utterance, and because 2) CL sentences are easy to handle as a canonical representation for applications such as speech dialogue systems.

3.1 Main idea underlying ASR for Japanese dialects

We simulate large dialect corpora to build a statistically reliable LM by transforming large CL corpora. The main problem in building a dialect ASR system is the shortage of large linguistic corpora in dialects due to rare transcriptions of sentences. The transformation produces large dialect corpora even if few actual dialect corpora are available. Each word in the new corpora contains the corresponding pronunciation in the dialects and the original word itself so that the dialect utterances can be recognized as CL sentences. This eliminates the cost of having to transform the ASR results again; we only need CL-to-dialect transformation for linguistic corpora and do not need reverse transformation. We can assume that only phoneme sequences are different because 1) as mentioned above, we handle dialects in structure of text processing, and because 2) Additionally, as we describe at the end of Section 2, we assume that the word order does not change. The vocabulary transformation system focuses on transformation at a level of phoneme sequences; it is called 'phoneme-sequence transducer' afterward.

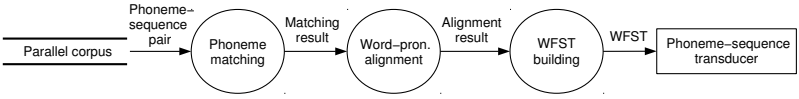
Our solution is composed of two steps:

1. training phoneme-sequence transducers
2. training LMs from corpora transformed with phoneme-sequence transducers

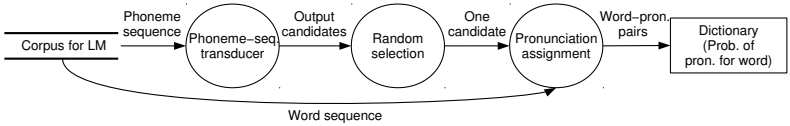
Figure 1 outlines data flow for our method.

Phoneme-sequence transducers are trained from CL-dialect parallel corpora in the first stage (Figure 1(a)). Units of *phonemes* are matched to corresponding sentences in parallel corpora. Next, pronunciations are aligned in units of *words*. Finally, *n*-gram models for phoneme-sequence transducers are trained from the results of alignment as sequences of phoneme-sequence pairs.

The second stage (Figure 1(b)) takes inputs from pronunciations of sentences in the CL corpora to phoneme-sequence transducers to obtain the corresponding pronunciations in the dialects.



(a) Training phoneme-sequence transducers.



(b) Training language models.

Figure 1: Data flow of our method.

After all sentences have been processed, the pronunciations that have been obtained are counted for each CL word entry, and the probabilities for each pronunciation are calculated.

The process involves four main steps:

1. phoneme-sequence transducer
2. random selection
3. pronunciation assignment
4. pronunciation dictionary

In the first step, the pronunciation of all sentences is transformed into dialects. This transformation is probabilistic; phoneme-sequence transducers output multiple candidates together with their probability. These probabilities are determined by the frequencies of transformation patterns in parallel corpora. If a transformation pattern frequently appear in parallel corpora, the phoneme-sequence transducers assign high probability to the pattern.

In the second step, to avoid only candidates with the maximum probability from being output, output is decided randomly from one of the candidates based on their probability. This is a kind of simulation of randomness of word choice. If only candidates with the maximum probability are output, only one pronunciation can be recognized for each CL word.

In the third step, phoneme-sequence transducers process pronunciations, not sentences themselves. This process deals with matching the output phoneme sequence to the original CL sentences (word sequences). After this process, each word in the original CL corpora will have its dialect pronunciation.

In the fourth step, pronunciation dictionaries in an ASR system contain each word entry and corresponding pronunciation as a phoneme sequence. Pronunciation dictionaries can contain multiple pronunciations together with their probabilities (LMs are treated as class n-gram models, in which each CL word entry corresponds to a class). Pronunciations and the corresponding probability is decided by the frequency of each pronunciation in the output of the previous process. These pronunciations make it possible to recognize dialect pronunciation as a word in the CL.

Our solution requires parallel corpora and linguistic corpora to train LMs. Parallel corpora are composed of pairs of phoneme sequences in a dialect and the CL. Our system uses the parallel

i	1	2	3	4	5	6	7	...																			
x	a	n	a	t	a	w	a	d	o	k	o	n	i	s	u	N	d	e	i	r	u	n	o				
y	a	N		t	a			d	o	k	o			s	u	N	d	e		r	u	N					
z	C	S	D	C	C	D	D	C	C	C	C	D	D	C	C	C	C	C	D	C	C	S	D				

(a) Example of alignment of phoneme-sequence pairs in parallel corpora. The first line is in the CL and the second line is in a dialect in the Kansai area (including Osaka). The third line is the matching result; `C` is a correct phoneme, `S` is a substitution error, and `D` is a deletion error.

```
a+a n_a+N t+t a+a w_a+NULL d+d o+o k+k o+o n_i+NULL
s+s u+u N+N d+d e+e i+NULL r+r u+u n_o+N
```

(b) Representation of transformation rules by using sequences of phoneme-sequence pairs. Symbol `+` separates two phoneme sequences in the CL and a dialect. `NULL` represents empty sequences.

Figure 2: Main idea in building rules for phoneme-sequence transducers.

corpus (National Institute for Japanese Language and Linguistics, 2008) composed of dialect sentences and the corresponding CL translations. They contain spoken sentences in various areas (prefectures) of Japan. The corpora for training LMs are sets of dialect sentences. They are created as explained in Section 3.1 using phoneme-sequence transducers.

3.2 Developing phoneme-sequence transducers

The rules for developing phoneme-sequence transducers are created from the parallel corpora previously mentioned. Briefly, the rules are created in two steps:

1. match of each pair of pronunciations,
2. obtain pronunciations in a dialect for each word in the CL.

First, each pair of pronunciations in parallel corpora is processed by matching based on the method of dynamic programming (DP-matching) using the minimum Levenshtein distance to create phoneme-pair sequences (Figure 2(a)), which describe what part of each pair of sequences corresponds to each other. Figure 3 outlines how to create sequences of pairs of the phoneme-sequences described in Figure 2(b) from the two phoneme sequences in Figure 2(a). Let `x[i]` be a phoneme sequence in the CL, and `y[i]` be that in a dialect. We have assumed that they have already been obtained by DP-matching together with the matching result, `z[i]`. Each element of `x` and `y` is a phoneme or empty. Each element of `z` is one of the following: `C` (correct phoneme), `S` (substitution error), `D` (deletion error) or `I` (insertion error).

We adopt WFSTs to build phoneme-sequence transducers. Phoneme-sequence transducers are represented as WFST $T = T_1 \circ L \circ T_2$ (The operation $\circ$ denotes the composition of (W)FSTs. See Allauzen et al. (2007) for more details), T takes phoneme sequences in the CL as input and the corresponding phoneme sequences in dialects together with their likelihoods as output. Figure 4 lists the roles of each (W)FST T_1, T_2, and L. T_1 is the FST for transforming a phoneme sequence in the CL into a sequence of phoneme-sequence pairs, in other words, enumerating sequences of phoneme-sequence pairs whose concatenation at the left is equal to the original phoneme sequence (see Figure 5(a)). T_2 is the FST for transforming a sequence of phoneme-sequence pairs into a phoneme sequence in a dialect, in other words, cutting down the left of each

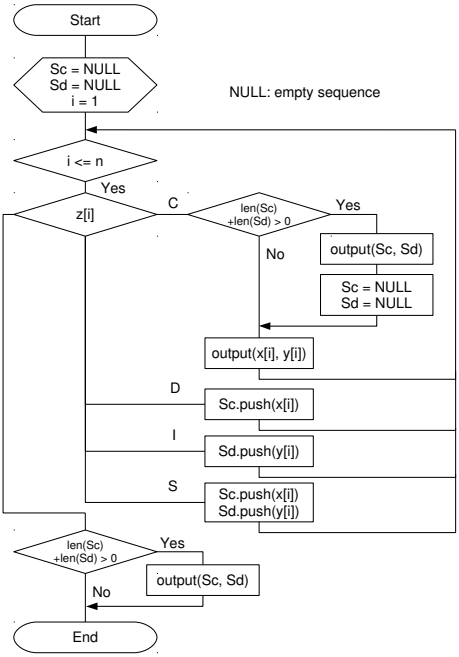


Figure 3: Creating sequences of phoneme-sequence pairs based on DP-matching results. Here 'Sc.push' append the symbol specified in the parameter to the end of sequence 'Sc'.

phoneme-sequence pair (see Figure 5(b)). L is the WFST of a 3-gram model of phoneme-sequence pairs (Chen, 2003) with the method of Kneser-Ney smoothing. L gives a likelihood value to each candidate and it can model phoneme transformation depending on the context. In this paper, OpenFst (Allauzen et al., 2007) is used for creating these (W)FSTs, and additionally Kylm is used for creating L.

We input phoneme sequence $\boldsymbol{x}$ in the CL to WFST T to obtain phoneme sequences in dialect $\boldsymbol{y}_1, \boldsymbol{y}_2, ...$ together with their likelihoods $L(\boldsymbol{y}_1|\boldsymbol{x}), L(\boldsymbol{y}_2|\boldsymbol{x}), ...$ (If $i < j$, $L(\boldsymbol{y}_i|\boldsymbol{x}) \geq L(\boldsymbol{y}_j|\boldsymbol{x})$). It is not efficient to calculate $L(\boldsymbol{y}_i|\boldsymbol{x})$ for all possible $\boldsymbol{y}_i$ since some of $\boldsymbol{y}_i$ have very small likelihoods and the number of candidates is sometimes very large. We only consider the n-best results $\boldsymbol{y}_1, ..., \boldsymbol{y}_n$ for the possible candidates, and cut off candidates from $\boldsymbol{y}_{n+1}$. Likelihoods $L(\boldsymbol{y}_i|\boldsymbol{x})$ for the possible candidates determine the probability of choosing $\boldsymbol{y}_i$; these probabilities $P(\boldsymbol{y}_i|\boldsymbol{x})$ are regularized likelihoods whose sum is equal to one:

$$P(\boldsymbol{y}_i|\boldsymbol{x}) = \frac{L(\boldsymbol{y}_i|\boldsymbol{x})}{\sum_{j=1}^{n} L(\boldsymbol{y}_j|\boldsymbol{x})}. \quad (1)$$

Next, we obtain pronunciation in a dialect for each word. One problem occurs here. The way pronunciation is transformed depends on its context, e.g., whether a given phoneme sequence

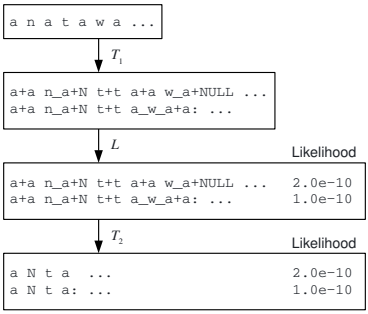
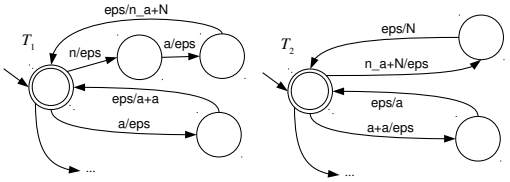


Figure 4: Roles of each (W)FST T_1, T_2, and L.



(a) Structure of T_1. (b) Structure of T_2.

Figure 5: Structure of FST T_1 and T_2. Each transition has a pair of input and output symbols delimited by symbol '/'. Symbol 'eps' represents a transition with no input or output symbols.

is itself a word or part of a word; viz., only given a pronunciation in the CL, some of the outputs of the phoneme-sequence transducers may be not suitable as pronunciation of the original word. We introduce word boundaries to the phoneme-sequence transducers. The modified phoneme-sequence transducers take phoneme sequence $\boldsymbol{x}$ containing some word boundaries in the CL and output at most n candidates of phoneme sequences containing word boundaries in a dialect. The modified phoneme-sequence is trained in three steps (see Figure 6).

1. extract what parts of given sequences correspond from phoneme-sequence pairs of pronunciations (Figure 6(a)) *without* word boundary information (Figure 6(b)).
2. align phoneme sequences to each word based on the extracted information (Figure 6(c)).
3. train word-based transformation rules from corresponding sequences *including* word boundary information (Figure 6(d)).

In the second step, we regard phoneme-sequence pairs crossing word boundaries as alignment of multiple ($m \geq 2$) CL words to a single dialect word, and insert $m-1$ symbol(s) representing boundary crossing before the next word boundary. The word-based transformation rules include identity transformations of a single phoneme such as a in the CL to a in dialects, so that the transducers can accept sequences containing a word that does not appear in parallel corpora. The transducers only output the same phoneme sequences for such words in input sequences.

Now, we are ready to transform the corpora. We segment words and estimate pronunciations for each sentence in large linguistic corpora in the CL to create input data for the modified

```
a n a t a | w a | d o k o | n i | s u | N | d e | i | r u | n o
a N t a d o k o s u N d e r u N
```

(a) Example pairs in parallel corpora. The first line is in the CL and the second line is in a dialect of the Kansai area. Symbol | represents word boundaries automatically decided by a morphological analysis tool.

```
a+a n_a+N t+t a+a w_a+NULL d+d o+o k+k o+o n_i+NULL
s+s u+u N+N d+d e+e i+NULL r+r u+u n_o+N
```

(b) Align two sentences at the phoneme level without word boundaries and express them with pairs of phoneme sequences. (Same as figure 2(b))

```
a n a t a | w a | d o k o | n i | s u | N | d e | i | r u | n o
a N   t a |     | d o k o |     | s u | N | d e |   | r u | N
```

(c) Align two sentences at the word level. This represents how each word is pronounced in a dialect.

```
a_n_a_t_a_|+a_N_t_a_|  w_a_|+|  d_o_k_o_|+d_o_k_o_|  n_i_|+|
s_u_|+s_u_|  N_|+N_|  d_e_|+d_e_|  i_|+|  r_u_|+r_u_|  n_o_|+N_|
```

(d) Transformation rules of phoneme sequences based on word-level alignment. Symbol | represents word boundaries.

Figure 6: Way in which word-level transformation rules were developed.

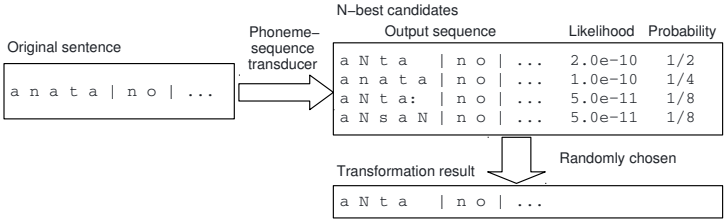
phoneme-sequence transducers. The modified phoneme-sequence transducers output phoneme sequences including word boundary information in a dialect. The transformed sequences are randomly chosen from the corresponding n-best results.

Figure 7 lists the process of building class n-gram LMs, in which each word entry is a class, from the transformed sentences. Class n-gram LMs allow many kinds of pronunciations to be manipulated without increasing the number of word entries of LMs. After all sentences have been transformed, the frequencies of pronunciations for each word entry are counted. We define the frequencies divided by the frequency of the word as the in-class probability of the pronunciation. Let $\#(\boldsymbol{x})$ be the number of CL word $\boldsymbol{x}$ that appears in the original sentences and $\#(\boldsymbol{y}|\boldsymbol{x})$ be the number of pronunciations $\boldsymbol{y}$ given to word $\boldsymbol{x}$; then the in-class probability, $P_c(\boldsymbol{y}|\boldsymbol{x})$, is written as

$$P_c(\boldsymbol{y}|\boldsymbol{x}) = \frac{\#(\boldsymbol{y}|\boldsymbol{x})}{\#(\boldsymbol{x})} = \frac{\#(\boldsymbol{y}|\boldsymbol{x})}{\sum_{\boldsymbol{y}} \#(\boldsymbol{y}|\boldsymbol{x})}. \quad (2)$$

3.3 Linguistic corpora to transform

The transformation method previously mentioned, of course, requires large linguistic corpora to transform. A former study (Lee et al., 2002) adopted 75 months of newspaper articles for a corpus, which is typical in studies on language models. Newspaper articles are relatively formal and in consistent style; therefore, they are suitable for recognizing speech in written articles, while not for spoken sentences including expressions that are characteristic of spoken language. One candidate for corpora in spoken language is the academic presentation speech corpora included in the *Corpus of Spontaneous Japanese* (CSJ) (Maekawa, 2003). This corpus consists of



(a) Transformation of sentences using phoneme-sequence transducers.

a n a t a | n o | ...	a N t a | n o | ...
a n a t a | w a | ...	a N t a: | ...
a n a t a | t o | ...	a N t a | t o | ...
a n a t a | k a r a | ...	a: t a | k a r a | ...
... | w a | a n a t a | ...	... | w a | a N t a | ...

In–class probabilities: P(a N t a | a n a t a) = 3/5, P(a N t a: | a n a t a) = 1/5, P(a: t a | a n a t a) = 1/5.

(b) Way in which in-class probabilities are determined. Phoneme-sequence transducers transform sentences including word **a n a t a** in the CL (at left) into sentences in a dialect (at right).

Figure 7: Way in which a corpus is transformed.

sentences including many technical terms, which are also not very suitable for spoken language.

This paper adopts corpora available on the Web, especially the Yahoo! *Chiebukuro* (Q&A) corpus. This corpus is presented by Yahoo! Japan Corporation and National Institute of Informatics (NII). Since corpora on the Web are created by various users, they contain various and some informal expressions like spoken language. The Yahoo! Q&A corpus contains sentences together with categories and subcategories to which each sentence belongs. It is possible to build LMs from sentences belonging to some specific categories near the target topics.

We adopt a corpus filtering method (Misu and Kawahara, 2006) in the Yahoo! Q&A corpus to build LMs. The major disadvantage of Web corpora is that some sentences are too inconsistent or not even in the form of sentences, e.g., Internet slang and ASCII arts. Speech recognition does not require these sentences and they are need to be excluded from corpora for training LMs. Corpus filtering is based on perplexity; we choose sentences with small perplexity on an LM from a set of sentence examples. These sentence examples were blog articles in the Balanced Corpus of Contemporary Written Japanese (BCCWJ) (Maekawa, 2008) core data. Words were segmented and pronunciations were estimated in BCCWJ core data these were manually checked by humans. Blog articles in the BCCWJ core data contained sentences that were close to those in spoken language including informal words and expressions.

4 Experiment

Our experiment evaluated the recognition accuracy of ASR. Dialect utterances were recognized as CL sentences and compared to referential CL sentences. We collected utterances in the CL and Kansai dialects. People from the Kansai area (Osaka, Hyogo, Nara and Shiga Prefectures in this experiment) read these sentences in their own dialects. The LMs for the CL were simply trained from the Yahoo! Q&A corpus. The LMs for the Kansai dialect were trained from the

	Data set	# of persons	# of sentences	# of words
Parallel corpora	Total		619	24,597*
	CL-Osaka		249	8,730*
	CL-Kyoto		226	6,980*
	CL-Hyogo		144	8,887*
Training LMs	BCCWJ Core		53,899	1,163,426
	Yahoo! Q&A		26,300**	1,164,317*
Evaluation	Kansai	4	100	1,682*
	CL	3		

*: Estimated by automatic word segmentation with KyTea.
**: Number of questions, because of difficulty of count sentences due to informal expressions.

Table 1: Size of corpora. The number of words in parallel corpora have been counted with reference to CL sentences.

pronunciation-transformed corpus based on the Yahoo! Q&A corpus mentioned in Section 3.2.

4.1 Conditions

Here we describe the training data for the phoneme-sequence transducers and LMs. Table 1 summarizes the size of corpora. Each LM had a common vocabulary size of 10,000.

This experiment adopted the parallel corpus (National Institute for Japanese Language and Linguistics, 2008) of the Kansai area (Osaka, Kyoto and Hyogo Prefectures). Each dialect sentence in this corpus was represented as pronunciation while each CL sentence was in plain text. We segmented CL sentences into words and estimated pronunciations of the words with KyTea (Neubig and Mori, 2010) so that the two kinds of sentences would have the same format in pronunciation.

We transformed the pronunciation of the Yahoo! Q&A corpus into that of the Kansai dialect to train the LMs. We chose 23,600 out of 335,685 questions in the category of daily life with the filtering method mentioned in Section 3.3, which has approximately the same number of words as the BCCWJ core data. One of at most the five-best dialect pronunciations was randomly chosen in the transformation, with the probability of their normalized likelihoods, and determined the probability of each pronunciation by using Equation 1.

Spoken sentences were translated into Kansai dialect by each speaker so that speakers would utter sentences clearly, since each speaker's dialect was slightly different. Each speaker read 100 sentences from blog articles in the BCCWJ. The spoken sentences and sentence examples for the filtering method did not overlap. We adopted Julius (Lee et al., 2001) as the ASR engine in this experiment, and the acoustic model was a phonetic tied-mixture (PTM) trigram model for Japanese language available at the Julius website.

4.2 Evaluation

This experiment evaluated ASR by recognition accuracy, Acc, calculated as

$$Acc = \frac{N - S - I - D}{N} \tag{3}$$

Language model		#1	#2	#3	#4	Average
Y	(Untransformed)	53.6	47.0	57.0	45.4	50.8
Y	(Dialect-transformed)	60.5	51.8	64.4	52.6	**57.3**
B	(Untransformed)	53.5	43.4	54.8	43.3	48.8
B	(Dialect-transformed)	60.1	49.4	63.9	49.4	55.7

Table 2: Word recognition accuracy of Kansai dialect [%]. Y and B stand for Yahoo! Q&A and BCCWJ, respectively.

Language model		#1	#2	#3	Average
Y	(Untransformed)	71.8	64.0	71.5	69.1
Y	(Dialect-transformed)	62.4	55.1	62.3	59.9
B	(Untransformed)	72.1	64.5	72.5	**69.7**
B	(Dialect-transformed)	62.0	56.3	58.7	59.0

Table 3: Word recognition accuracy of the CL [%]. Y and B stand for Yahoo! Q&A and BCCWJ, respectively.

where N, S, I, and D correspond to the sum of the lengths of referential word sequences, substitution errors, insertion errors, and deletion errors.

Tables 2 and 3 summarize word recognition accuracy for the Kansai dialect and CL, for LMs trained from transformed and untransformed corpora of Yahoo! Q&A and BCCWJ core data. The LMs from the Yahoo! Q&A corpus had better recognition accuracy than the LMs from the BCCWJ. As explained in Section 3.3, the Yahoo! Q&A corpus contained more sentences that had the characteristics of spoken sentences, which matched the blog articles of spoken sentences. The Yahoo! Q&A corpus made it easy to match specific kinds of topics by category filtering. These characteristics of the Yahoo! Q&A corpus made a difference despite the same size of the two corpora. Additionally, the LMs from transformed corpora resulted in better word recognition accuracy for the Kansai dialect than those from untransformed corpora (the opposite characteristics appeared for the CL). This means that the ASR system actually recognized some dialect-specific expressions like spoken language with LMs from transformed corpora. This demonstrated our method's effectiveness.

Dialect transformation was proved to reduce the effect of pronunciation-estimation errors, seeing the created dialect corpora. Automatic pronunciation estimation causes some errors, and these errors affect the recognition accuracy. Dialect transformation was proved to output correct pronunciations for inputs of mistakenly-estimated pronunciations, by training this "mistaken" transformation rules. Since dialect pronunciation in parallel corpora is correct, words with mistaken pronunciation in corpora for LMs are transformed into the correct dialect pronunciation if errors occur in a consistent way.

We interpolated the in-class probabilities of pronunciations of a dialect and the CL to recognize both pronunciations. The interpolation of probabilities is defined as follows; let P_c of Equation (2) for dialect d be rewritten as $P_{c,d}$, then

$$P_{c,mix}(\boldsymbol{y}|\boldsymbol{x}) = \sum_d \alpha_d P_{c,d}(\boldsymbol{y}|\boldsymbol{x}), \quad (4)$$

$$\text{s.t.} \quad \sum_d \alpha_d = 1, \ \alpha_d \geq 0$$

Transformation ratio			#1	#2	#3	#4	Average
Y	0%	(Untransformed)	53.6	47.0	57.0	45.4	50.8
Y	25%		60.8	52.6	66.1	52.2	57.9
Y	50%		61.3	51.9	65.9	51.1	57.6
Y	75%		62.0	53.8	66.0	52.9	**58.7**
Y	100%	(Completely transformed)	60.5	51.8	64.4	52.6	57.3

Table 4: Word recognition accuracy of Kansai dialect [%] with interpolated pronunciation dictionaries.

Language model		#1	#2	#3	#4	Average
Y	(75% transformed)	66.0	56.6	68.9	55.6	**61.8**

Table 5: Word recognition accuracy [%] after ignoring variation of expressions in spoken language.

gives the interpolated in-class probabilities, $P_{c,mix}$. Table 4.2 lists the recognition accuracy with interpolated pronunciation dictionaries.

Word recognition accuracy with a transformation ratio of 75% (i.e., $\alpha_{\text{dialect}} = 0.75$, and $\alpha_{\text{CL}} = 0.25$) scored the best (58.7%), which is 9.9 points higher than the result for LMs trained from the BCCWJ, and 7.9 points higher than that for LMs trained from the untransformed Yahoo! Q&A corpus. Dialect-transformed LMs have dialect pronunciations of words, but fewer kinds of CL pronunciations. Not all words in spoken language are characteristic of dialects; spoken language is composed of both dialect and CL pronunciations. The result showed that interpolated dictionaries was able to improve recognition accuracy more.

Word recognition accuracy depends on four components.

1. phoneme-sequence transducers and their parallel corpora
2. corpora with dialect-specific words
3. acoustic models
4. variation of expressions in spoken language

The first component, *phoneme-sequence transducers and their parallel corpora*, was the main idea presented in this paper. Parallel corpora determine what pronunciation ASR systems can recognize as dialect expressions. Phoneme-sequence transducers in this paper had word boundary features as well as phoneme sequences themselves. Pronunciation in a dialect can actually differ from words having the same pronunciation in the CL, depending on the part-of-speech (POS) tags of each word. One of possible improvement is including the POS tags of each word and its previous and next words to phoneme-sequence pairs in Figure 6(d).

The second component, *corpora with dialect-specific words*, is necessary to recognize actual dialogues in specific areas. Spoken sentences in this experiment did not include dialect-specific proper nouns. In other words, the problem was how to collect sentences containing such proper nouns. Corpus candidates are local pages in newspapers. Among major newspaper companies in Japan, Mainichi newspapers Co., Ltd. distributes data of local pages as well as national press.

The third component, *acoustic models*, is required to deal with acoustic features specific to dialects (e.g., changes in phonemes). Since the acoustic model in this experiment did not assume dialect speech recognition, acoustic features specific to dialects may affect word recognition accuracy.

The fourth component, *variation of expressions in spoken language*, makes non-essential errors affect recognition accuracy. Some words and phrases, especially those in spoken language, have the same meaning and role in sentence and are unnecessary to distinguish. It is important for ASR systems to rather recognize the meaning of sentences than strictly recognize sentences word by word. If they also have similar pronunciation, the variations in recognition results are likely to increase recognition errors. For example, *want to* and *wanna* in English language, even though not in all cases, can be regarded as the same phrases. In Japanese language, some particles (e.g., *na* and *ne* on the end of sentences, corresponding to tag questions in English) and verbs (...*teiru* and ...*teru*: progressive form) have similar variations.

We modified the results of the 75%-transformed Yahoo! Q&A corpus to correct errors related to the fourth component. Table 5 lists word recognition accuracy after modifications. We should regard these results to demonstrate the considerable accuracy of ASR systems. These were 3.1 points higher than that of the same LM in Table 2 (61.8%) on average.

Conclusions

This paper described how to develop an ASR system that could recognize utterances in Japanese dialects. The main idea behind our system was how to create dialect corpora, few of which are actually available. We developed phoneme-sequence transducers trained from dialect-CL parallel corpora to statistically model transformations of pronunciations between dialects and the CL. Each word in the linguistic corpora was labelled as dialect word pronunciations to simulate dialect corpora. The experiment with measuring of word recognition accuracy confirmed the effectiveness of our system in recognizing dialect utterances. We were able to obtain higher recognition accuracy by adopting sentences like spoken language as corpora for training LMs. Furthermore, interpolation of in-class (pronunciation) probabilities of the CL and dialects improved recognition accuracy a little more. Our method does not depend of language and dialects; an ASR system for another language could be developed as long as parallel corpora were available.

Even though this paper assumed dialects would have no effect on the word order, a little more work should be necessary to handle slight changes in the word order as mentioned in Section 1. One possible solution is to introduce a parameter of extraneous word generation probability, p_0, like IBM model 3 (Brown et al., 1993). If WFST L (see Section 3.2) is represented by larger n-gram models ($n = 3$ in this paper) than the length of phrases within which the word order changes, WFST would model a few changes in the word order.

Our next project will be developing an ASR system to recognize various dialects alone. One possible solution to recognize utterances in multiple dialects is interpolation of in-class probabilities of pronunciations of the dialects in the same way as Section 4.2. This treatment may be too simple to work well because it assumes independence of dialects of each word. Adjacent words are intuitively likely to belong to the same dialects, and its modeling will be the main problem of recognition of multiple dialects. After that, it will be a further step to train acoustic models from dialect utterances and develop a method of switching dialects.

Acknowledgments

This study was partially supported by a Grant-in-Aid for Scientific Research (S) (No. 24220006) and the Global COE Program.

References

Akita, Y. and Kawahara, T. (2010). Statistical transformation of language and pronunciation models for spontaneous speech recognition. *Audio, Speech, and Language Processing*, 18(6):1539–1549.

Allauzen, C., Riley, M., Schalkwyk, J., Skut, W., and Mohri, M. (2007). OpenFst: A general and efficient weighted finite-state transducer library. In *Proc. of CIAA 2007, Lecture Notes in Computer Science*, volume 4783, pages 11–23. Springer.

Anusuya, M. and Katti, S. (2009). Speech recognition by machine: A review. *International Journal of Computer Science and Information Security*, 6(3):181–205.

Benincà, P., editor (1989). *Dialect variation and the theory of grammar*. Foris Publications.

Brown, P., Pietra, V., Pietra, S., and Mercer, R. (1993). The mathematics of statistical machine translation: Parameter estimation. *Computational linguistics*, 19(2):263–311.

Chen, S. (2003). Conditional and joint models for grapheme-to-phoneme conversion. In *Proc. of EuroSpeech 2003*.

Ching, P., Lee, T., and Zee, E. (1994). From phonology and acoustic properties to automatic recognition of Cantonese. In *Proc. of Speech, Image Processing and Neural Networks, 1994*, pages 127–132.

Gadet, F. and Jones, M. (2008). Variation, contact and convergence in french spoken outside france. *Journal of language contact*, 2(1):238–248.

Gottlieb, N. (2005). *Language and society in Japan*. Cambridge University Press.

Lee, A., Kawahara, T., and Shikano, K. (2001). Julius—an open source real-time large vocabulary recognition engine. In *Proc. of EuroSpeech 2001*, pages 1691–1694.

Lee, A., Kawahara, T., Takeda, K., Mimura, M., Yamada, A., Ito, A., Itou, K., and Shikano, K. (2002). Continuous speech recognition consortium —an open repository for CSR tools and models—. In *Proc. of LREC 2002*, pages 1438–1441.

Lyu, D., Lyu, R., Chiang, Y., and Hsu, C. (2006). Speech recognition on code-switching among the Chinese dialects. In *Proc. of ICASSP 2006*, volume 1, pages 1105–1108.

Maekawa, K. (2003). Corpus of spontaneous japanese: Its design and evaluation. In *ISCA & IEEE Workshop on Spontaneous Speech Processing and Recognition*.

Maekawa, K. (2008). Balanced corpus of contemporary written Japanese. In *Proc. of ALR6 2008*, pages 101–102.

Miller, D. and Trischitta, J. (1996). Statistical dialect classification based on mean phonetic features. In *Proc. of ICSLP 1996*, volume 4, pages 2025–2027.

Misu, T. and Kawahara, T. (2006). A bootstrapping approach for developing language model of new spoken dialogue systems by selecting web texts. In *Proc. of ICSLP 2006*, pages 9–12.

Munteanu, C., Penn, G., and Zhu, X. (2009). Improving automatic speech recognition for lectures through transformation-based rules learned from minimal data. In *Proc. of ACL and AFNLP*, pages 764–772. Association for Computational Linguistics.

National Institute for Japanese Language and Linguistics, editor (2001–2008). *Database of Spoken Dialects all over Japan: Collection of Japanese Dialects (In Japanese)*, volume 1–20. Kokushokankokai.

Neubig, G., Akita, Y., Mori, S., and Kawahara, T. (2012). A monotonic statistical machine translation approach to speaking style transformation. *Computer Speech & Language*, 26(5):349–370.

Neubig, G. and Mori, S. (2010). Word-based partial annotation for efficient corpus construction. In *Proc. of LREC 2010*, pages 2723–2727.

Neubig, G., Mori, S., and Kawahara, T. (2009). A WFST-based log-linear framework for speaking-style transformation. In *Proc. of InterSpeech 2009*, pages 1495–1498.

Wolfram, W. (2009). Dialect awareness, cultural literacy, and the public interest. In *Ethnolinguistic Diversity and Literacy Education*, chapter 6, pages 129–149. Routledge.

Woods, H. (1979). *A socio-dialectology survey of the English spoken in Ottawa: A study of sociological and stylistic variation in Canadian English*. PhD thesis, The University of British Columbia.

Zhang, X. (1998). Dialect MT: a case study between Cantonese and Mandarin. In *Proc. of ACL and COLING 1998*, volume 2, pages 1460–1464.

Tailored Feature Extraction for Lexical Disambiguation of English Verbs Based on Corpus Pattern Analysis

Martin HOLUB[1] *Vincent KRÍŽ*[1]
Silvie CINKOVÁ[1] *Eckhard BICK*[2]
(1) Charles University in Prague, Faculty of Mathematics and Physics
Malostranské náměstí 25, Prague
(2) University of Southern Denmark, Campusvej 55, Odense M
{Holub|Kriz|Cinkova}@ufal.mff.cuni.cz, Eckhard.Bick@mail.dk

ABSTRACT
We give a report on a detailed study of automatic lexical disambiguation of 30 sample English verbs. We were drawing on a lexicon of English verb patterns based on the Corpus Pattern Analysis (CPA), which is a novel lexicographic method that seeks to cluster verb uses according to the morpho-syntactic, lexical and semantic/pragmatic similarity of their contexts rather than to associate them with abstract semantic definitions. We have trained several statistical classifiers to recognize these patterns, using morpho-syntactic as well as semantic features. In this paper we mainly concentrate on the procedures for feature extraction and feature selection and their evaluation. We show that tailoring the features to the verbs respectively, as they are implicitly contained in the pattern definitions (explicitly described in the lexicon), has the potential to significantly improve the accuracy of supervised statistical classifiers.

TITLE AND ABSTRACT IN CZECH

Rysy šité na míru anglickým slovesům pro automatickou lexikální disambiguaci pomocí Corpus Pattern Analysis

Předkládáme detailní studii automatické lexikální disambiguace na pilotním vzorku třiceti anglických sloves za použití lexikonu vzorů slovesných užití (patterns), který vychází z Corpus Pattern Analysis (CPA). Tato inovátorská lexikografická metoda namísto na abstraktních definicích jednotlivých významů staví na souhře morfosyntaktické, lexikální a sémantické/pragmatické podobnosti slovesných užití. Natrénovali jsme několik statistických klasifikátorů na rozpoznávání těchto vzorů. Klasifikátory využívají jak morfosyntaktických, tak sémantických rysů. V naší studii se soustředíme na procedury pro extrakci rysů, jejich výběr a jejich evaluaci. Ukazujeme, že rysy na míru uzpůsobené jednotlivým slovesům, jež jsou implicitně obsaženy v definici každého vzoru v lexikonu, mají potenciál významně zvýšit přesnost statistických klasifikátorů s učitelem.

KEYWORDS: English verbs, Corpus Pattern Analysis, supervised lexical disambiguation, tailored feature extraction, machine learning.

KEYWORDS IN CZECH: anglická slovesa, Corpus Pattern Analysis, automatická lexikální disambiguace, rysy šité na míru, strojové učení s učitelem.

Proceedings of COLING 2012: Technical Papers, pages 1195–1210,
COLING 2012, Mumbai, December 2012.

1 Introduction

This study focuses on the lexical semantics of English verbs and its automatic analysis based on contextual hints, morpho-syntactic as well as lexical. It is generally known that the manual annotation of verbs for Word Sense Disambiguation (WSD) tasks has to face the issue of inter-annotator confusion. The commonest verbs are often used to represent several events or aspects of an event at once. For instance, *throwing bread crumbs to the birds* comprises a number of different but interlinked events: propelling an object (typically humans with hands), targeting a propelled object, discarding an object, passing an object to someone else, passing it to an animal as food. Fine-grained lexicons would list all or many of these partial events, since there are good examples of contexts, where one of the aspects is outstanding: throwing missiles, throwing something away/in the sink, throw corn to chickens, etc. In contexts like the one mentioned above, where none of the partial events is dominating, the inter-annotator confusion is almost inevitable, since the instance matches several semantic definitions at once. Using a coarse-grained lexicon, on the other hand, would mean that *throwing darts* and *throwing sour milk in the sink* are similar events, with all the implications for inferencing or translations. Facing this issue, we became fascinated by the Corpus Pattern Analysis, a manual method of sorting corpus concordances according to their morpho-syntactic, lexical and semantic/pragmatic similarity, coined by Hanks (1994). Our current work has been inspired by its implementation, the Pattern Dictionary of English Verbs (PDEV) (Hanks and Pustejovsky, 2005). PDEV as a database has been practically developed at Masaryk University in Brno (Horák et al., 2008) and is publicly available at http://deb.fi.muni.cz/pdev/.

PDEV is a semantic concordance built on yet a different principle than FrameNet, WordNet, PropBank, or OntoNotes: the manually extracted patterns of frequent and normal verb uses are, roughly speaking, intuitively similar uses of a verb that express "in a syntactically similar form" a similar event in which similar participants (e.g. humans, artifacts, institutions, other events) are involved. Two patterns can be semantically so tightly related that they could appear together under one sense in a traditional dictionary. The patterns are *not* senses but syntactico-semantically characterized *prototypes*. Concordances that match these prototypes well are called *norms* while concordances that match them with a reservation (metaphorical uses, argument mismatch, etc.) are called *exploitations* (Hanks, forthcoming). The PDEV corpus annotation indicates the norm-exploitation status for each concordance. Compared to other semantic concordances, the granularity of PDEV is high and thus discouraging in terms of expected inter-annotator agreement. However, selecting among patterns does not really mean disambiguating concordance but rather determining to which pattern it is most similar — a task easier for humans than WSD is. This principle seems particularly promising for verbs as words expressing events, which resist the traditional word sense disambiguation the most.

2 Lexicon of Verb Semantic Patterns

Each lexical entry in the PDEV scheme consists of numbered categories (an example is given in Table 1). Each category consists of a *pattern* and an *implicature*. The pattern represents the morphological, syntactic and lexical characteristics of the verb used in a certain context. The meaning is represented by the implicature. The pattern takes the form of a predication. The pattern-defining verb complements are represented by *semantic types* or *lexical sets*. A lexical set is a list of characteristic collocates, whereas semantic types are items in Hanks' ontology.

Verb	No.	Pattern / Implicature
gleam	1	**[[Physical Object | Surface]] gleam [NO OBJ]** [[Surface]] of [[Physical Object]] reflects occasional flashes of light
gleam	2	**[[Light | Light Source]] gleam [NO OBJ]** [[Light Source]] emits an occasional flash of [[Light]]
gleam	3	**{eyes} gleam [NO OBJ] (with [[Emotion]])** {eyes} of [[Human]] shine, expressive of [[Emotion]]
wake	3	**[no object] [Human] wake ({up}) AdvTime({from} {nightmare | dream | sleep | reverie}) ({to} Eventuality)** the mind of [[Human]] returns at a particular [[Time]] to a state of full conscious awareness and alertness after sleep
wake	4	**pv [phrasal verb] [[Human 1] ^ [Sound] ^ [Event]] wake [[Human 2] ^ [Animal]] ({up})** [[Human 1 | Sound | Event]] causes the mind of [[Human 2 | Animal]] to return to a state of full conscious awareness and alertness after sleep
wake	7	**[Anything] wake [Emotion] ({in} Human)** [[Anything]] causes [[Human]] to feel or become aware of [[Emotion]]
wake	9	**waking * ({up})** [Human|Animal]'s returning to a state of full conscious awareness and alertness after sleep

Table 1: Example patterns defined for the verbs *gleam* and *wake*.

2.1 Pilot Sample English Verbs

We have performed our experiments using a newly developed lexical resource called *VPS-30-En*, recently published by Cinková et al. (2012). *VPS-30-En* (Verb Pattern Sample, 30 English verbs, henceforth VPS) is a pilot lexical resource of 30 English lexical verb entries enriched with semantically annotated corpus samples. VPS is publicly available on the web page http://ufal.mff.cuni.cz/spr/pdev30verbs.[1] The data describes regular contextual patterns of use of the selected verbs in the BNC (2007). VPS has arisen as a practical result of previous studies published by Hanks, drawing on his PDEV, see e.g. (Hanks and Pustejovsky, 2005). VPS contains the verbs showed in Table 2.

VPS is a collection of 30 revised PDEV verbs in which the adjustments of the entries and the original concordance samples were driven by inter-annotator agreement (IAA) findings. The collection was designed as a small sample of PDEV that was revised and cleaned up as a gold-standard data set for statistical pattern recognition.

During the annotation, the annotators got a random 50-concordance sample along with the lexicographer-annotated reference sample and the entry. They matched each random concordance to the categories according to the similarity of implicatures, the similarity of the patterns and, not least, according to the overall similarity of the concordance to the concordance clusters associated with the respective categories.

[1]This language resource has been developed and/or stored and/or distributed by the LINDAT-Clarin project of the Ministry of Education of the Czech Republic (project LM2010013). In the LINDAT-Clarin repository the VPS data is available under the handle https://ufal-point.mff.cuni.cz/xmlui/handle/11858/00-097C-0000-0005-BF95-B.

Verb	Verb characteristics							Human accuracy				MFC
	Orig. Tagset size	Tagset size	Training set size	Weight %	Frequency group	Perplexity	Fleiss kappa	Annotator AV %	Annotator JT %	Annotator EK %	Average %	Baseline %
access	10	4	300	0.29	C	3.1	0.73	86.3	84.3	86.3	85.6	47.0
ally	8	5	250	0.24	C	3.9	0.73	88.2	80.4	90.2	86.3	47.6
arrive	7	6	250	3.83	B	3.0	0.92	94.1	96.1	94.1	94.8	68.0
breathe	18	7	350	0.60	C	5.1	0.84	94.1	94.1	82.4	90.2	37.7
claim	11	6	500	7.85	A	3.0	0.85	94.1	86.3	92.2	90.9	67.8
cool	16	7	300	0.36	C	5.5	0.88	88.2	82.4	90.2	86.9	27.3
crush	14	9	350	0.27	C	6.9	0.65	82.4	62.8	90.2	78.4	28.9
cry	15	5	250	0.75	B	3.5	0.84	94.1	94.1	88.2	92.2	52.4
deny	12	7	300	3.02	B	5.2	0.69	84.3	68.6	90.2	81.0	44.7
enlarge	6	5	300	0.33	C	2.3	0.62	94.1	66.7	92.2	84.3	76.7
enlist	6	5	300	0.22	C	3.5	0.86	93.8	87.5	91.7	91.0	49.0
forge	14	9	350	0.32	C	7.4	0.64	82.4	72.6	76.5	77.1	26.3
furnish	9	5	300	0.25	C	4.3	0.80	94.1	92.2	74.5	86.9	43.7
hail	10	5	300	0.49	C	2.9	0.83	92.2	98.0	90.2	93.5	67.4
halt	4	4	250	0.54	C	1.8	0.70	94.1	88.2	90.2	90.9	83.6
part	13	9	300	0.24	C	6.2	0.86	94.1	90.2	86.3	90.2	43.0
plough	18	9	250	0.22	C	6.9	0.95	96.1	96.1	92.2	94.8	32.4
plug	14	11	300	0.22	C	8.7	0.72	76.6	80.9	83.0	80.1	31.3
pour	22	10	300	0.57	C	7.9	0.74	90.2	80.4	76.5	82.4	24.3
say	16	6	500	59.3	A	1.9	0.90	96.1	96.1	96.1	96.1	85.2
smash	12	6	300	0.33	C	4.0	0.81	92.2	86.3	90.2	89.5	53.4
smell	11	8	300	0.26	C	5.8	0.88	94.1	86.3	94.1	91.5	36.3
steer	24	14	300	0.27	C	11.1	0.73	80.4	82.4	86.3	83.0	20.3
submit	6	5	250	1.42	B	2.6	0.88	98.0	88.2	96.1	94.1	70.8
swell	25	11	300	0.25	C	9.0	0.82	78.4	80.4	88.2	82.4	21.7
tell	18	9	500	13.5	A	3.8	0.93	98.0	94.1	94.1	95.4	65.2
throw	74	26	1000	2.33	B	16.9	0.65	80.4	62.8	78.4	73.9	22.7
trouble	14	10	300	0.24	C	6.2	0.76	96.1	72.6	88.2	85.6	44.3
wake	11	7	300	0.57	C	4.8	0.78	88.2	82.4	88.2	86.3	45.0
yield	12	10	300	0.93	B	7.4	0.76	86.3	78.4	82.4	82.4	29.0

Table 2: Basic characteristics of the 30 sample English verbs under study. For detailed explanation see Section 3.

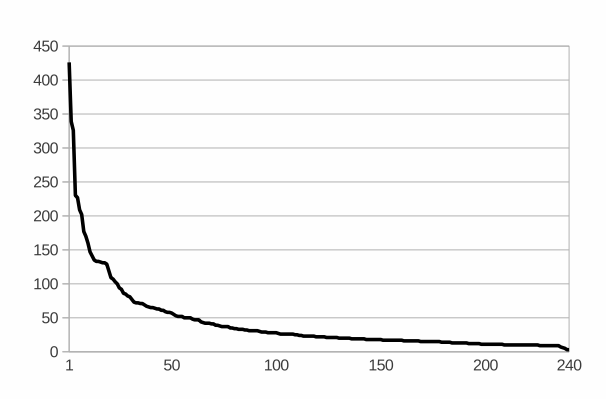


Figure 1: Number of examples of the classified tags in the training data set. The x-axis corresponds to the 240 pattern tags sorted by their frequency in the training data. The value on the y-axis is the number of the training example sentences. For example, only 55 pattern tags have more than 50 examples in our training data.

After each annotation round, IAA was measured and disagreements were manually analyzed. The disagreement analysis was supported by confusion matrices computed for each annotator pair. Provided the annotation of the random sample reached a satisfactory IAA, the disagreements were manually adjudicated by the lexicographer in a spreadsheet table: the lexicographer highlighted evident annotation errors, listed all acceptable values and "one best choice" in a separate column to each concordance. The "one best" annotation was typed back into the user interface as part of the gold standard data set.

The gold standard data set consists of the reference sample and of the adjudication table for the adjudicated sample. As a rule, it consists of 350 concordances (a 250-concordance original sample, one 50-concordance trial sample, and one 50-concordance adjudicated sample).

3 Experimental Data Set

To keep things simple we have neglected the norms and exploitations in this series of experiments. Also, we have, to a significant extent, preserved the classical WSD setup, i.e. both the annotators and the classifier are forced to pick one tag only.

Table 2 shows the most important characteristics of the verbs in the VPS data set. VPS contains about 450 different pattern tags. However, we reduced the number of patterns for classification task and accepted only those with more than 8 occurrences in the training data. The number of pattern tags that we use is 240. The verbs are divided into three frequency groups (A, B, C) according to their frequency in the corpus. Note that 6 most frequent verbs in the data set (*say, arrive, claim, deny, throw* and *tell*) cover 90% of our subcorpus.

The distribution of the number of examples per one tag in the training data set is shown in Figure 1. Unfortunately for most of the 240 tags that should be classified we have a small

Frequency Group	Weight	Perplexity	MFC Average Accuracy %
A	80.7	2.3	73.9 ±0.5
B	12.3	6.5	49.9 ±0.7
C	7.1	5.3	43.7 ±0.5
All	100.0	3.0	68.8 ±0.5

Table 3: Accuracy baselines (the accuracy of a MFC classifier) for all verbs and by frequency groups.

number of training examples (only 91 tags have more than 30 examples).

We also measure the perplexity of the verbs using a standard formula based on the entropy of a probabilistic distribution. The weighted average perplexity of verbs and the baseline accuracy by frequency groups are summarized in Table 3. MFC stands for a most-frequent-case classifier (i.e. the most frequent tag of the training set is used to classify all instances of the test set). The training set was randomly divided into 9 parts to perform a 9-fold cross-validation. In Tables 3 and 6 the AVG accuracy is displayed together with the confidence intervals based on the standard t-test at the significance level $\alpha = 5\,\%$.

The test sets contain 50 multi-annotated instances of each verb. The IAA values (measured as Fleiss' kappa) displayed in Table 2 were calculated using human annotations made independently by 4 annotators. In Table 2 we also indicate the values of "human accuracy", just to illustrate how difficult the classification task is for people.

4 Feature Extraction and Selection

We have identified the feature extraction for machine learning as a central issue, with great impact on the performance of automatic classifiers. Since we used only one-sentence contexts, we dealt only with *local* features. Each data instance to be classified consists of the target verb (TV) and some context words. Therefore the features that describe data instances are based on the observed characteristics of both the TV and the context words. Intuitively it seems to be a good idea to follow the structure of the defined patterns. Therefore we use two kinds of features for machine learning, the morpho-syntactic features and the semantic features. All features used in this study are binary, i.e. have only 0/1 values.

4.1 Types of features and feature sets

4.1.1 Morpho-syntactic features

We have considered three types of morpho-syntactic features: 1) morphological features of the target verb, 2) morphological features of words in a contextual window, 3) syntactic dependencies. By the morpho-syntactic feature set we were seeking to analyze collocates in relevant argument positions, negation and the modifiers of arguments. To alleviate the automatic parser errors, we formulated some features just based on part-of-speech tags. For instance, a (likely) direct object is also encoded as a noun following the target verb in a given window. We were also observing the tense, mood and voice of the target verb. Also, we were taking into account whether the verb is governed by or governs another verb. We were drawing on (Semecký, 2007), adapting the features to English.

First we tuned a model for pattern classification using only morpho-syntactic features, and only then we tried and improved it by using a set of semantic features.

4.1.2 Semantic features

Semantic features capture the semantics of nominal subjects and/or objects of the TV and/or prepositional phrases dependent on the TV, which exactly corresponds with semantic types and/or lexical sets indicated in the pattern definitions. For that purpose we use the system of *semantic prototypes* developed by Bick (first publicly mentioned in (Bick, 1996); the current documentation is available on the Web[2]). In fact, to create semantic features we take only the most coarse-grained level of the semantic prototypes, which is called "umbrella" categories. In Bick's system each of the 150–200 semantic prototypes is assigned to some of 40 umbrella categories, which are grouped into 22 "major umbrellas": *animal, botanical, human, location, vehicle, abstract concepts, actions/events/processes, anatomical, things (concrete/countable), clothes, materials, collectives/parts, domain concepts, features, food, perceptions and feelings, semantic/semiotic, state of affairs, time, tools/machines, units/quantities, weather*.

The semantic prototypes for English are drawing on similar systems for Portuguese and Danish. The original motivation for creating the semantic prototypes, such as <Hprof> (professional human), <tool> or <sem> (semantic/semiotic product) was the polysemy resolution for Portuguese-Danish machine translation in a Constraint Grammar context. Thus, context-driven rules are used to remove or select semantic categories for a given lemma, depending on its syntactic function, inflexion, definiteness and dependency relations. The granularity of the semantic prototypes was chosen to match linguistic usefulness, rather than for its descriptive value as such. Thus, tags should optimally be distinguishable with linguistic tests, such as the combinatorial potential, e.g. which prepositions are typically used with a noun in question, plural vs. mass determiners, or testing verbs: "you can eat it, drink it, write it . . . ". Too low a granularity (high level of abstraction) would reduce the distinctive power, too high a granularity (low level of abstraction) would make it impossible to express general contextual rules and not gain much compared to lexical rules targeting the individual lemma.

4.1.3 Universal and tailored feature sets

We developed and evaluated two kinds of feature sets. While *universal* feature sets are common to all verbs under study, *tailored* feature sets are verb-specific. The procedure for extracting the tailored features is fully automatic and is based on an automatic analysis of the pattern definitions. As we show in this paper, using tailored feature sets enables us to build automatic classifiers with a significantly better performance.

After many experiments we tuned and evaluated 5 models for feature extraction and selection. First, we started with a basic universal model that deals only with morpho-syntactic features (U1). Second, to evaluate the contribution of semantic features, we designed a more advanced universal model that also uses semantic features (U2). Then we focused on using the specific clues contained in the pattern definitions and developed 3 tailored models (T1, T2, and T3) that work with all kinds of features and differ in procedures for feature selection. An overview is given in Table 4.

[2] http://beta.visl.sdu.dk/semantic_prototypes_overview.pdf

Model	Number of features	Relation to verbs	Feature characteristics	Feature selection method
U1	58	universal	morpho-syntactic features only	selection using Decision Trees (Information Gain)
U2	65	universal	morpho-syntactic and semantic features	same as U1
T1	68-106	tailored	the U2 set + features based on pattern definitions	same as U1 and U2 + features tailored to verbs
T2	19-88	tailored	all morpho-syntactic, semantic, and pattern-based features	greedy forward selection to maximize SVM accuracy
T3	29-147	tailored	the union of the feature sets selected by T1 and T2	greedy backward elimination to maximize SVM accuracy

Table 4: Overview of the models used for feature extraction.

4.2 Morpho-syntactic Feature Extraction

Morpho-syntactic features are extracted from sentences using both a morphology analyzer based on the Penn TreeBank (PTB) morphological tagset (Santorini, 1990) and the Stanford parser with its Stanford dependencies representation (de Marneffe et al., 2006). In total we have established 79 fixed binary features and 4 lexicalized features. In fact the lexicalized ones generate a number of binary features, depending on the occurrence of certain auxiliary words (prepositions, particles, and conjunctions) in the training data. All morpho-syntactic features can be divided into 3 groups:

Characteristics of the TV
10 binary features: Passive voice, modality-1 (would, should), modality-2 (can, could, may, must, ought, might), negation, tense (PTB tags: VBN, VBD, VBG, VBP, VB), use in an infinite phrase (outside subject).

Characteristics of the context words that immediately precede or follow the TV
Context is limited to ± 3 words simply by the word order. 9 binary features have been established for each of the 6 closest context words (in total 54 binary features): nominal-like (NN, NNS, NNP, NNPS, DT, PDT, PRP, PRP$, POS, CD), adjective (JJ, JJR, JJS), verbs (VB, VBD, VBG, VBN, VBP, VBZ), modal (MD), adverbial (RB, RBR, RBS, RP, IN), *to* (TO), wh-pronoun (WDT, WP, WP$), wh-adverb (WRB), to_be (lemma = *be*).

Characteristics of the context words that syntactically directly depend on the TV
a) logical subjects (3 binary features: nominal subject, clausal subject, subject in the plural form); b) objects (8 binary features: direct object, indirect object, passive nominal subject, passive clausal subject, clausal complement, complementizer (typically the subordinating conjunction "that" or whether"), any object, any object in the plural form); c) particles (lexicalized); d) adverbials (4 binary features: adverbial modifier, adverbial clause modifier, purpose clause modifier, temporal modifier); e) preposition (lexicalized: prepositional modifier or prepositional clausal modifier); f) markers (lexicalized: subordinating conjunctions different from *that* or *whether*).

4.3 Morpho-syntactic Feature Selection

Feature selection was performed in two steps. First, we filtered out the features with useless value distribution. As a threshold we used the condition that the less frequent (binary) value should be detected in our training data at least 5 times. After that filtering we had 149 binary features. The second step was reducing the remaining feature set in order to "optimize" the classifier performance. After many experiments the following heuristic procedure won. For each of the 30 verbs separately we searched for a small subset of the features with the best performance using a decision tree classifier. We started with only one best feature and then greedily added further best features and tested the classifier performance. When it was not possible to improve the accuracy by adding any of the remaining features, the process stopped. Then all the "best" small sets for all 30 verbs were united and we got an overall feature set containing 58 morpho-syntactic features. We call this model "U1". Its accuracy for different verbs is shown in Table 5 and in Figure 2. We experimentally checked that U1 could be hardly beaten regarding the overall accuracy measured as a weighted average of all 30 verbs.

4.4 Semantic Feature Extraction and Selection

Our universal model U2 deals only with the umbrella prototypes (22 major + 40 subordinated) and observes only the semantic types of subjects and objects. As a starting point, the feature selection procedure takes the overall set of 124 (binary) semantic features (62 for the semantic type of subjects, and other 62 for objects). The final feature selection was done analogously as the previous selection of the morpho-syntactic features. We started with the union of the best 58 morpho-syntactic features and all 124 semantic features. First, we greedily searched for the "best" small feature subset for each of 30 verbs separately. Then we took the union of all small subsets. The result was a set of 65 features consisting of 21 semantic and 44 morpho-syntactic ones. We call this model "U2". Again, its accuracy for different verbs can be seen in Table 5 and in Figure 2. The overall average results for all 30 verbs are given in Table 6.

4.5 Tailored Feature Extraction and Selection

The extraction of the tailored features is based on contextual hints described in the patterns. This process is driven by 1) the presence of a member of a lexical set defined in patterns, 2) the verb forms indicated in patterns, 3) prepositions listed in prepositional phrases described in patterns, 4) particles dependend on the TV, 5) types of object clauses allowed in patterns, 6) the "no_object" attribute defined in patterns.

The tailored models differ in the feature selection method used. The T1 model simply uses the U2 feature set and all tailored featured corresponding to a given verb. The T2 model takes all possible features and greedily selects the best ones to maximize accuracy of an SVM classifier. The most advanced T3 model takes the union of the T1 and T2 feature sets and then reduces the whole set by greedy backward elimination, again to maximize accuracy of an SVM classifier.

5 Supervised Pattern Classification: Model Choice and Tuning

We experimented with several supervised machine learning methods, namely k-Nearest Neighbours (kNN), Decision Trees (DT), AdaBoost.M1 (ADA) based on DT, Support Vector Machines (SVM), and Naive Bayes classifier (NB). Our results are perfectly in line with the observation reported in (Màrquez et al., 2007) that the best results are obtained using SVM or ADA. We also observed that in case of small samples for different patterns, the SVM model tends to be

Verb	Average accuracy						Best tailored model				
	MFC	U1	U2	T1	T2	T3	M	#F	Acc	Imp-B	Imp-U
access	47.0	78.0	77.7	77.4	79.0	79.7	T3	55	79.7	69.5	2.6
ally	47.6	65.5	66.8	67.6	79.6	79.2	T2	54	79.6	67.3	19.2
arrive	68.0	70.8	72.4	76.8	82.0	82.6	T3	41	82.6	21.4	14.1
breathe	37.7	65.7	72.3	76.3	79.4	81.0	T3	41	81.0	114.7	12.0
claim	67.8	82.4	82.8	87.4	80.6	82.6	T1	75	87.4	28.9	5.5
cool	27.3	63.0	63.4	65.4	66.7	67.6	T3	36	67.6	147.1	6.6
crush	28.9	37.1	46.3	50.3	53.4	53.5	T3	56	53.5	85.3	15.5
cry	52.4	72.4	73.6	77.2	78.8	80.4	T3	44	80.4	53.4	9.2
deny	44.7	55.7	60.7	67.7	63.3	63.0	T1	74	67.7	51.6	11.5
enlarge	76.7	82.0	80.7	84.0	82.0	84.8	T3	43	84.8	10.6	5.1
enlist	49.0	74.4	84.4	84.7	89.4	89.9	T3	51	89.9	83.5	6.6
forge	26.3	48.3	52.6	59.7	56.9	58.6	T1	86	59.7	127.2	13.6
furnish	43.7	65.0	69.7	72.0	77.7	79.0	T3	49	79.0	80.8	13.4
hail	67.4	85.0	83.7	85.4	81.7	84.6	T1	73	85.4	26.7	2.0
halt	83.6	85.2	86.8	87.6	88.0	90.9	T3	59	90.9	8.8	4.8
part	43.0	73.0	73.0	72.7	69.0	67.0	T1	74	72.7	68.9	-0.4
plough	32.4	69.3	70.9	73.6	74.0	76.5	T3	44	76.5	135.9	7.9
plug	31.3	51.0	59.4	58.7	58.7	61.7	T3	41	61.7	96.7	3.9
pour	24.3	52.3	55.7	56.7	58.9	63.8	T3	77	63.8	162.1	14.5
say	85.2	90.6	90.6	90.8	86.0	86.9	T1	82	90.8	6.6	0.2
smash	53.4	65.1	69.7	74.3	76.7	77.7	T3	46	77.7	45.7	11.5
smell	36.3	57.0	61.0	63.0	58.3	63.7	T3	37	63.7	75.2	4.3
steer	20.3	40.4	44.0	45.6	49.0	50.6	T3	55	50.6	149.1	15.1
submit	70.8	85.2	85.2	85.6	84.0	86.8	T3	76	86.8	22.6	1.9
swell	21.7	46.6	51.0	57.3	62.0	62.8	T3	45	62.8	189.8	23.2
tell	65.2	75.8	79.2	79.4	79.2	81.2	T3	69	81.2	24.6	2.5
throw	22.7	43.0	42.6	53.7	56.6	56.6	T3	147	56.6	149.3	32.9
trouble	44.3	70.7	69.7	72.4	66.0	65.5	T1	75	72.4	63.2	3.8
wake	45.0	76.7	77.3	77.7	69.7	69.8	T1	75	77.7	72.6	0.5
yield	29.0	46.6	51.9	52.6	55.3	56.0	T3	46	56.0	93.1	7.8

Table 5: Comparison of two universal and three tailored models. The best model is always one of T1, T2, or T3. #F is the number of the features selected by the best model. Imp-B and Imp-U stand for improvement over the baseline and over the best universal model U2, respectively.

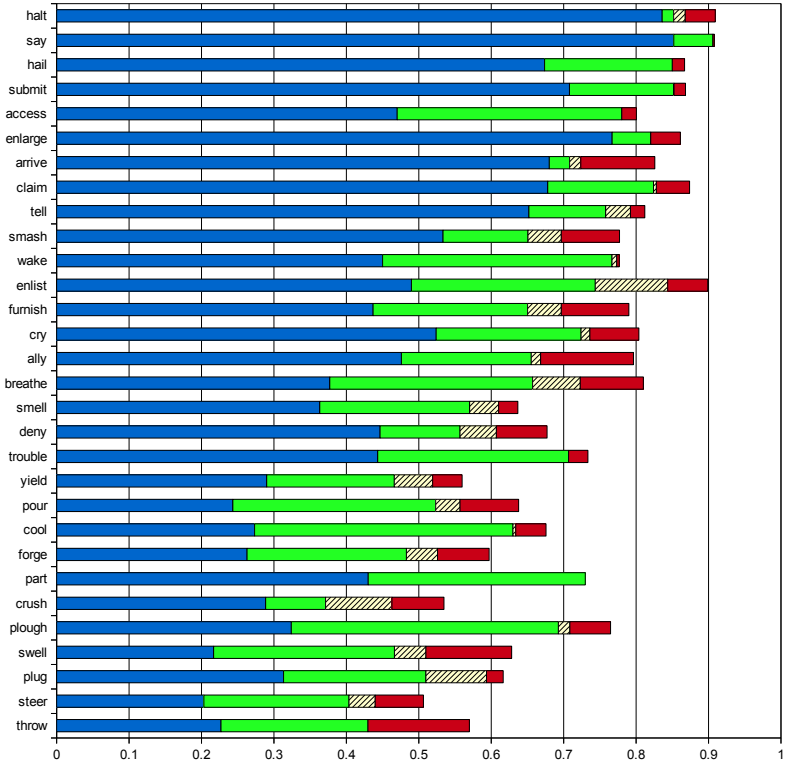


Figure 2: Accuracy improvement over the MFC baseline (in blue): the best models using universal feature sets U1 (green) and U2 (cream), and the best models using tailored feature sets (red).

Frequency Group	Best universal				Best tailored			
	AVG Accuracy %	Improvement %	ERR decrease %	Test Acc %	AVG Accuracy %	Improvement %	ERR decrease %	Test Acc %
A	87.9 ±1.9	10.4	38.2	86.4	88.9	11.8	41.5	83.1
B	63.9 ±2.1	38.3	27.1	58.8	72.3	60.6	45.6	66.6
C	68.7 ±0.9	72.0	42.8	66.9	74.1	87.8	53.4	69.6
All	83.6 ±1.6	18.2	37.1	81.6	85.8	23.1	42.8	80.1

Table 6: Accuracy of the best SVM models. *AVG Accuracy* is the result of the cross-validation test on the training data set. *Improvement* is the percentage difference between the AVG Accuracy and the Baseline Accuracy. *ERR decrease* stands for the percentage difference between the baseline error rate and the error rate of the respective model. *Test Acc* is the accuracy measured on the test sets.

better than ADA (cf. the overview article by Schapire (2003)). Finally we trained our best models (U1, U2, T1, T2, T3) using the SVM method and a grid search approach to parameter optimization. The results described in Table 5 and in Figure 2 show that tailored models almost always clearly outperform the universal ones. However, among the tailored models there is no absolute winner. Unfortunately, we have not yet developed one "best" method for feature selection that would universally lead to the best performance for any verb. Therefore we chose the best tailored model for each verb separately, according to the results of the cross-validation experiments performed on the training data.

6 Evaluation and error analysis

6.1 Universal models

Our overall result is that the "semantically enriched" model U2 slightly outperforms the "only morpho-syntactic" model U1, which can be observed in Table 5. However, only the difference for the low-frequent verb group is statistically significant. In Figure 2 the verbs are sorted according to the increasing perplexity; this figure also shows the decreasing accuracy tendency, which has naturally been expected.

The classifiers did not know the pattern definitions for the respective verbs. We made this decision in order to see to what extent the default feature set would do. This approach, along with the sparsity of our data, resulted in a few systematic errors. The most striking one is the misclassification of patterns that prescribe a participial form of the verb. When these patterns were not frequently assigned, the classifier did not learn the most important feature — that the verb should be a participle that is not used in an obvious passive voice, but is already a transition to an adjective or a noun (e.g. *cooling* in *cooling towers*). The classifier often assigned this tag to concordances in which the verbs did not have the form of a participle.

Another drawback of the classifiers is that they have not been fed a list of phrasemes. As the data is so sparse, idioms either remain unrecognized, or they are interpreted literally. For

Feature type	Frequency	%
All tailored	**283**	**16.1%**
Tailored – lexical sets	94	5.3%
Tailored – prepositions	68	3.9%
Tailored – particles	24	1.4%
Tailored – clauses	15	0.9%
Tailored – verb form	59	3.4%
Tailored – no object	23	1.3%
All semantic	**462**	**26.2%**
Semantic – Subj	118	6.7%
Semantic – Obj	304	17.3%
Semantic – PP	40	2.3%
All morpho-syntactic	**1016**	**57.7%**
All	**1761**	**100.0%**

Table 7: Overview of the structure of features used in best tailored feature sets.

instance the concordance *This organisation happily <ploughs> a furrow totally at odds with the notion of free trade* is interpreted as an agricultural context. This problem goes beyond just idioms, since many patterns with limited collocability are defined by lexical sets — lists of typical collocates. The nouns in these lists are often quite heterogeneous, encompassing several semantic types and the association with a semantic type is irrelevant. For instance *claim credit for something*. The data is too small for such lists to be learned directly.

Another interesting issue is semantic modulation in nouns. For instance, the verb *halt* distinguishes between abstract processes, such as financial crises, and vehicles or human groups in military contexts to be halted. In the following concordance, *advance* is a process, but what is really meant are the men and vehicles advancing. Semantic modulation is a typical cause of annotator confusion, often mimicked by the classifier: *And even Crown Prince Rupprecht, far removed from Verdun, had warned him days before the offensive began that the advance would be <halted> by flanking fire from the Left Bank.*

6.2 Tailored models

To some extent, tailored feature sets are able to provide a remedy for the errors described above. A summary is given in Table 6 to compare our best universal model U2 with the best tailored models (specific for each verb). Although tailored features cause the accuracy increase, Table 7 indicates the fact that the morpho-syntactic features dominate even in the best tailored models.

A glance at the results provided by the tailored models reveals a couple of observations. We have compared the results for the test data with the human confusion matrices and with the overall outcome of the universal models as we have described it in the previous section. We have identified four interesting points:

1) Participial patterns. The universal models did not learn that the confusion between a participial and a regular pattern is only acceptable when the target verb is in a participial form. The tailored models learned this successfully for most verbs where participial patterns occurred. There is only one major exception: cool 11 (participial) – confused for cool 1 (intransitive).

2) Patterns with a different number of objects. In several verbs, e.g. "deny something" (deny 9) confused for "deny somebody something" (deny 10), neither model learned to discriminate according to the number of non-prepositional objects, although the presence of the indirect object was among the features. We suspect this confusion to occur due to parsing errors.

3) Syntactically similar patterns with different implicatures or semantic types. The classifiers are in trouble whenever two pattern definitions are syntactically similar and the only difference lies in the semantic types of the collocates. Although the universal features contain the semantic types, this semantic information is not sufficiently granular. Unlike lexical sets, we do not have any detailed information on which words correspond to which semantic types.

4) Heterogenity of 'u' and 'x' tags. The most systematic error is in many verbs a pattern number assigned to a concordance classified as 'u' or 'x'. We speculate that learning these negative instances is extremely difficult. Their examples in the data are very heterogeneous, and each of them can be more similar to a positive instance, respectively, than they are among one another.

6.3 Future work

As a next step, we would like to exploit the potential of the semantic types determined in the patterns. We need to develop a robust method to "populate" the semantic types with lexical units. Also, we need to gain a better insight into the performance of parsers, since the most important features are inarguably the syntactic ones. A weak spot of the tailored features is that, in *some* cases, our best tailored models slightly overfit the training data (as can be observed in Table 6). So we need to make the proces of feature selection more robust. The main issue, however, that sets the limits on the performance of supervised classifiers seems to be the lack of sufficient amount of reliable training examples.

Conclusion

The two main goals of our research were to evaluate the usefulness of semantic prototypes if we use them directly as features for statistical learning, and to evaluate the power of features tailored to individual verbs and based on automatic analysis of pattern definitions. Our result is in line with previously published studies that usually agree on the fact that the morpho-syntactic features are the most important for statistically-driven semantic disambiguation. Nevertheless, for *some* verbs the use of semantic features plays an important role. The positive impact of tailored features is obvious.

Acknowledgments

This research work has been supported by the Czech Science Foundation (grant project no. P103/12/G084) and partly by the project META-NET (FP7-ICT-2009-4-249119 of the EU and 7E11040 of the Ministry of Education, Youth and Sports of the Czech Republic).

We thank our friends from Masaryk University in Brno for providing the annotation infrastructure and for their permanent technical support. We thank Patrick Hanks for his CPA method, for the original PDEV development, and for numerous discussions about the semantics of English verbs.

References

Bick, E. (1996). Automatic parsing of Portuguese. In *Garcia, Laura Sánchez (ed.), /Anais / II Encontro para o Processamento Computacional de Português Escrito e Falado/*. Curitiba: CEFET-PR.

Cinková, S., Holub, M., Rambousek, A., and Smejkalová, L. (2012). A database of semantic clusters of verb usages. In *Proceedings of the LREC 2012 International Conference on Language Resources and Evaluation*. Istanbul, Turkey.

de Marneffe, M.-C., MacCartney, B., and Manning, C. D. (2006). Generating typed dependency parses from phrase structure parses. In *Proceedings of the 5th International Conference on Language Resources and Evaluation (LREC 2006)*.

Hanks, P. (1994). Linguistic norms and pragmatic exploitations, or why lexicographers need prototype theory and vice versa. In F. Kiefer, G. K. and Pajzs, J., editors, *Papers in Computational Lexicography: Complex '94*. Research Institute for Linguistics, Hungarian Academy of Sciences.

Hanks, P. and Pustejovsky, J. (2005). A pattern dictionary for natural language processing. *Revue Francaise de linguistique appliquée*, 10(2).

Horák, A., Rambousek, A., and Vossen, P. (2008). A distributed database system for developing ontological and lexical resources in harmony. In *9th International Conference on Intelligent Text Processing and Computational Linguistics*, pages 1–15. Berlin: Springer.

Màrquez, L., Escudero, G., Martínez, D., and Rigau, G. (2007). Supervised Corpus-Based Methods for WSD. In Agirre, E. and Edmonds, P., editors, *Word Sense Disambiguation*. Springer, 2007.

Santorini, B. (1990). Part-of-Speech tagging guidelines for the penn treebank project. Technical Report MS-CIS-90-47, University of Pennsylvania.

Schapire, R. E. (2003). The boosting approach to machine learning: An overview. In Denison, Hansen, Holmes, Mallick, and Yu, editors, *Nonlinear Estimation and Classification*. Springer, 2003.

Semecký, J. (2007). Verb valency frames disambiguation. PhD. Thesis. Technical report, Institute of Formal and Applied Linguistics, Charles University in Prague.

Method mention extraction from scientific research papers

Hospice Houngbo *Robert E. Mercer*
Department of Computer Science
The University of Western Ontario, London, ON, Canada
hhoungbo@uwo.ca, mercer@csd.uwo.ca

ABSTRACT
Scientific publications contain many references to method terminologies used during scientific experiments. New terms are constantly created within the research community, especially in the biomedical domain where thousands of papers are published each week. In this study we report our attempt to automatically extract such method terminologies from scientific research papers, using rule-based and machine learning techniques. We first used some linguistic features to extract fine-grained method sentences from a large biomedical corpus and then applied well established methodologies to extract the method terminologies.

We focus the present study on the extraction of method phrases that contain an explicit mention of method keywords such as (**algorithm, technique, analysis, approach and method**) and other less explicit method terms such as **Multiplex Ligation dependent Probe Amplification**. Our initial results show an average F-score of 91.89 for the rule-based system and 78.26 for the Conditional Random Field-based machine learning system.

KEYWORDS: terminology extraction, rule-based, machine learning, corpus, linguistic features.

Proceedings of COLING 2012: Technical Papers, pages 1211–1222,
COLING 2012, Mumbai, December 2012.

1 Introduction

The methods and techniques used during scientific experiments and reported in scientific papers are often expressed in different forms. They can be in the form of a semantic sequence (see (Hunston, 2008) containing the word *method ("rank-based normalization method" in Sentence 1)*, the word *technique ("Non-negative matrix factorization technique" in Sentence 2)*, the word *analysis ("discriminant analysis" in Sentence 3)*, and so on.

Sentence 1: *In order to compare U133A and U133 Plus 2.0 data we further normalized the data with a* ***rank-based normalization method****.*

Sentence 2: *In the Bioinformatics field a great deal of interest has been given to* ***Non-negative matrix factorization technique NMF*** *due to its capability of providing new insights and relevant information about the complex latent relationships in experimental data sets.*

Sentence 3: *The relative performance of the four IBDQ dimensions in distinguishing best patients with minor symptoms from those with severe was studied by* ***discriminant analysis****.*

A method mention can be a terminology term such as the phrase **Multiplex Ligation dependent Probe Amplification** in Sentence 4.

Sentence 4: *Recently* ***< Multiplex Ligation dependent Probe Amplification > < MLPA >*** *has also been used to quantify copy number classes.*

A method expression can also be a verb phrase referring to an action performed during an experiment, such as the verb phrase *to search for* in sentence 5.

Sentence 5: *These sequences were used* ***to search for*** *the nearly invariant nucleotides of the inverse core AAC and core GTT sites separated by a distance typical of previously identified attC sites to bp .*

In some cases, the context in which the method terminologies are used in the text can contain valuable information about the method mention, such as synonyms, definition, and other relations with entities in the text.

The automatic extraction of terminologies has been the focus of many studies in the past (Maynard and Ananiadou, 2000), (Zhang and Wu, 2012), but not many works have focussed on automatic extraction of method mentions. We believe that the extraction of method expressions from research papers can help to build lexical resources that can be used in various NLP tasks on research papers.

For example, the automatic recognition of method expressions can help to easily detect method sentences and classify them into their rhetorical categories as in (Agarwal and Yu, 2009).

In addition, the automatic extraction of method mentions and the information surrounding them can be useful in dictionary building, ontology population and glossary creation. Also, it can help in the task of knowledge discovery from scientific papers, as any mention of methods and techniques used in that paper can easily be presented to readers without them having to read the whole text.

The extraction of contextual information around the method mentions can be useful in building Question-answering and focussed text summarization systems.

In this study, we present our attempt to extract method terminologies from sentences. Also we showed how we can use a terminology context to extract other relevant information about the term. We define "other relevant information" to be the syntactic and semantic relationship between the term and other words. It may be a definition, a variant terminology and so on.

This study doesn't take into account the extraction of verbs as method mention in scientific research papers.

The contribution of our work is twofold. First we used some linguistic filters to automatically select thousands of fine-gained method sentences that have a high probability to contain methodology terminologies; then we used the context of such sentences to extract terminologies and other information about them.

The rest of this paper is organized as follow. The next section reviews some related works. In Section 3, a detailed methodology of method mentions extraction techniques is presented. In Section 4, the results are described. We conclude the paper by a summary and directions for future work.

2 Related work

Automatic term extraction is an important component of many natural language processing systems. It is often used in applications such as knowledge discovery, knowledge management, automatic text indexing, and so on. Many studies have been conducted on the recognition of terminologies from research papers, especially in the biological domains where new terms are created constantly. These studies primarily focussed on the extraction of domain specific concepts such as nouns and collocations. But it is hasn't been possible to apply a general rules to extract all the terms.

Previous studies on automatic term extraction have been made possible with the availability of many language resources and large electronic corpora. Three main approaches are used in terminology extraction, namely, linguistics-based approaches, statistical approaches and hybrid approaches.

2.1 Linguistics-based approach

A term from a terminology is often a unit of meaning related to a specific field or domain of study. It can be a single word, such as *clustering*, or a compound expression made up of individual terms, such as *Hidden Markov Model*. Some terms in a terminology can follow comprehensive rules which can help with their generalization.

With the linguistics-based approach, candidate terminology is filtered by linguistic features using morphological analysis, such as part of speech (POS). Complex terms are extracted using shallow parsing and dependency analysis between words in the sentence (Bourigault, 1992). (Dagan and Church, 1994) limited the candidate terminology to a string that represents the pattern of noun sequences. Good results can be achieved in small corpora using linguistic methods, but due to the shortage of patterns, recall can be low and it is difficult to generalize these techniques across fields and languages.

2.2 Statistical approach and machine learning approach

Statistical approaches are based on statistical information, such as the frequency of terms appearing in the corpus. As already noted in (Zhang and Wu, 2012), statistical features can include term document frequency and inverse document frequency TF*IDF (Maedche and Staab, 2000), KF*IDF (Xu et al., 2002), C-value/NC-value (Frantzi et al., 2000) and so on. Term extraction is based on the computation of the Unithood – i.e. a degree of strength or stability of syntagmatic combinations or collocations, and Termhood – i.e. the degree that a linguistic unit is related to a domain-specific concept (Kageura and Umino, 1996) of the terminology.

Termhood calculation is often based on the frequency of the term in the paper or a given baseline corpus.

(Church and Hanks, 1990) used Mutual Information (MI) to compute Unithood, (Dunning, 1993) used LogL , and (Patry and Langlais, 2005) computed left/right entropy to extract unithood candidates. When computing termhood, methods such as TF*IDF (Maedche and Staab, 2000), DR-D (Velardi et al., 2001), C-value/NC value (Frantzi et al., 2000)) and Domain Component Feature Set (DCFS) (Zhang and Sui, 2007) are employed, as reported in (Zhang and Wu, 2012).

2.3 Hybrid approach

Linguistics-based and statistical approaches have their own advantages and disadvantages. They are often integrated to extract terminology. There are two ways to combine them. One way is to extract candidate terms with linguistics methods and then if no terms are found then the statistical methods are applied. (Daille, 1996) used the linguistic methods to get candidate terms and set them as the input of statistical models. Then statistical methods such as MI and LogL are used to get final terms.

The other way is to obtain candidate terms using statistical methods first and then use linguistic methods to discard those terms that are inconsistent with linguistic patterns. (Maynard and Ananiadou, 2000) extracted multi-word terms using a thesaurus and the semantic Web concept to get the semantic and category information, and then integrated it with the statistical and syntactic information in the corpora. Different terminology extraction toolkits can also be integrated to extract terminology besides the integration of linguistics and statistics methods. (Vivaldi and Rodriguez, 2001) integrated different term extraction tools using simple voting and the results were better than that with single term extraction tools. (Vivaldi et al., 2001) improved the previously mentioned voting approach, got the best integration strategy with Boosting algorithm and improved the performance of terminology extraction based on the hybrid approach.

3 Methodology

3.1 Corpus creation

The goal of this study is to extract the methods and techniques used in biomedical research papers in order to build a lexical resource comprising the name, the variants and definition of such methods and techniques as they are mentioned and used in research articles. The first task consists of the gathering of a comprehensive corpus that is large enough to contain an important number of method mentions and information about how they are used in the papers. One way is to select some research papers and scan through each of them to extract candidate

"method sentences" manually. Another way is to use filters to automatically select in a large repository of papers, "method sentences" that we believe are about the methods or techniques used in the paper as well as the context of such sentences. We define the context of a "method sentence" to be a window of sentences coming after or before them. The first option has proved in the past to be more precise, but very difficult to implement. In fact, a manual information extraction task can be tedious and time-consuming and requires many specialized skills and domain experts. It is therefore recommended to use automatic extraction techniques whenever it is possible. Since our purpose is to find as many relevant sentences as possible, it is obvious that we cannot rely on existing corpora to achieve our goal.

Some recent works have focussed on the classification of sentences from biomedical articles into the IMRAD (Introduction, Methods, Research, and, Discussion) categories. In their attempt to classify biomedical research papers into these categories (Agarwal and Yu, 2009) used a corpus of 1131 sentences. Of these sentences, 389 were labelled Introduction, 363 were labelled Methods, 273 were labelled Results and 106 were labelled Discussion. Even though the corpus contained "method" sentences, very few contain the mention of the techniques used in the paper. Also, the context of the sentence was not retrieved. (Liakata et al., 2012) used a corpus of 265 articles from biochemistry and chemistry annotated at the sentence level by experts in the domains. Even though the corpus contains 8404 method sentences, many sentences belong to the same papers and are about the same methods or techniques. For instance 10 consecutive sentences can be annotated to belong to the method category. Those sentences usually refer to the same method or technology and some of them may not even contain any mention of a method. Besides, few of the sentences contain a definition of the techniques used in the papers.

Based on the study of the corpora mentioned above, we deemed it necessary to build a different corpus that contains as many method sentences as possible and a definition or a usage context of the method mention.

We relied on some linguistic concepts such as anaphoric relations to produce our fine-grained method corpus.

In fact, most demonstrative noun phrases are anaphoric; therefore a sentence that begins with the demonstrative noun phase "This method" is anaphoric and its antecedents are likely to be found in previous sentences. (Torii and Vijay-Shanker, 2005) reported their work on anaphora resolution of demonstrative noun phrases in Medline abstracts, and found that nearly all antecedents of such demonstrative phrases can be found within two sentences. On the other hand, (Hunston, 2008) reported that interpreting recurring phrases in a large corpus enables us to capture the consistency in meaning as well as the role of specific words in such phrases. So, the recurring semantic sequence "this method" in the Pubmed corpus can help us to capture valuable information in the context of their usage.

To build our corpus we therefore search for sentences starting with "This method" in the PubMed article repository as well as the sentences immediately preceding them. Then we collected the pairs of sentences.

Sentence 6 in Example 1 contains the mention of the method and Sentence 7 contains its usage and definition context.

Example 1

Sentence 6 *: The Ortholuge method reported here appears to significantly improve the specificity*

(precision) of high-throughput ortholog prediction for both bacterial and eukaryotic species .

Sentence 7 *: This method , and its associated software , will aid those performing various comparative genomics-based analyses , such as the prediction of conserved regulatory elements upstream of orthologous genes .*

We can see that Sentence 6 is talking about the method and Sentence 7 is a reference to the method mention and how it is used. Combining both sentences we therefore have sufficient information to extract the "method" mention, its usage and its benefits.

We can then derive such information to fill lexical components such as:

- Method Mention: ***Ortholuge method***
- Usage/Role: ***comparative genomics-based analyses/ prediction of conserved regulatory elements upstream of orthologous genes.***

Example2

Sentence 8 *: An alternative method for predicting protein function is the Phylogenetic profile method, also known as the Co-Conservation method , which rests on the premise that functionally related proteins are gained or lost together over the course of evolution [4] .*

Sentence 9 *: This method predicts functional interactions between pairs of proteins in a target organism by determining whether both proteins are consistently present or absent across a set of reference genomes.*

In Example 2, it is possible to extract the following information:

- Method : ***Phylogenetic profile method***
- Variant/ also known as:***Co-Conservation method***
- Usage: ***for predicting protein function / predicts functional interactions between pairs of proteins***
- How: ***by determining whether both proteins are consistently present or absent across a set of reference genomes.***

The information that we can derive from this pair is sufficient enough to create lexical resources that can be used in many natural language processing tasks.

Using the retrieval technique mentioned above, we have been able to retrieve about 6500 such pairs of sentences from 189 different journals and 2000 papers.

We limit the scope of this study to the extraction and recognition of the method terminologies, due to time constraints.

3.2 Gold Standard datasets

We created 2 sets of gold standards with sentences taken only from BiomedCentral journals. The first gold standard comprises 918 pairs of sentences containing the first category of method mention – i.e. terminology units ending with a method keyword. The second gold standard comprises 122 pairs of sentences of method mentions that don't contain a method keyword. In

each gold standard, we assumed that the method mention is in the first sentence and the other information about its usage is in the second sentence.

We used grammatical rules to extract the first category (method mentions that contain keywords, such as algorithm, technique, analysis, approach and method) and machine learning techniques to extract the second category of method mention (those that don't contain the above keywords).

When a method keyword is not explicitly mentioned in a sentence containing a method mention, it is not obvious to apply general grammatical rule to recognize it as the words composing it can be of various forms. In the following sentences:

Sentence 10 : Enault and colleagues proposed an improved **< phylogenetic profile >** based on a **< normalized Blastp bit score >**.

Sentence 11 : Another way to obtain suboptimal solutions from a **< HMM >** is to do **< HMM sampling >**.

Sentence 12 : In this paper we introduce a **< Bootstrap procedure >** to test the null hypothesis that each gene has the same relevance between two conditions where the relevance is represented by the Shapley value of a particular coalitional game defined on a microarray data set .

we can notice that the different method mentions - *< phylogenetic profile >, < normalized Blastp bit score > < HMM >, < Bootstrap procedure >* are all of different word shapes. Also when most of them are nouns phrases, they can be confused with other noun phrases in the sentence. We thus believed that the extraction of this type of method mentions can be viewed as a named entity recognition task (Maynard et al., 2001), (Palmer and Day, 1997).

For this second task, we transformed each sentence into the BIO format (Table 1). There are 284 manual-tagged terms, 122 sentences, 2871 words and punctuations.

In Table 2 we shows the number of sentences in each dataset.

Enault and colleagues proposed an improved **< phylogenetic profile >** based on a **< normalized Blastp bit score >**	Enault	O
	and	O
	colleagues	O
	proposed	O
	phylogenetic	B-method
	profile	I-method
	based	O
	on	O
	a	O
	normalized	B-method
	Blastp	I-method
	bit	I-method
	score	I-method
	.	O

Table 1: Representation of a method sentence in the BIO format.

Category (keywords)	Number of sentences	Proportion
Method	439	42%
Analysis	200	19%
Model	63	6%
Algorithm	73	7%
Approach	145	14%
Other (Machine learning corpus)	122	12%
Total	1040	100%

Table 2: Corpus statistics (combining both datasets).

3.3 Rule-based extraction

Most of the method mentions in the first category can be represented by the following examples:

1. *rank-based normalization method*
2. *HRV power spectral analysis*
3. *Non-negative matrix factorization technique*
4. *linear regression analysis*
5. *Newton-type algorithm*
6. *tube group amplification approach*
7. *progressive alignment algorithm*
8. *metabolite profiling approach coupling mass spectrometry*
9. *profile-based HMM method*
10. *multifactor-dimensionality reduction MDR method*

As we can notice these are simple grammatical patterns that can be extracted with simple rules.

1. They can start and continue either with an adjective or a noun.
2. They can continue either with an adjective or a noun.
3. They end with a method keyword.

These rules can be represented by the following regular expression:

(Adjective | Noun)+(method | analysis | algorithm | approach| model)

To extract such patterns, we first used the Genia tagger (Tsuruoka et al., 2005), a Part-of-Speech tagger trained on a biomedical corpus, to tag every word in the sentence. Then, we used the rules to extract all phrases and terminologies that correspond to the above mentioned patterns. The results are presented in the Result section.

3.4 Machine learning and feature extraction

3.4.1 Feature extraction

The feature pool includes:

1. Word feature

   The word itself.

2. Part-of-speech tags

   We used a POS tagger (the Genia tagger) to tag each word in the sentence.

3. Word-shape features

   We check whether the word is lower case, upper case or has both lower case and upper case letters.

   These features include: isAllCaps, StartWithCap, isAllLowerCase, isMixedCase.

4. Position features

   We checked if the word is at the beginning of sentence (BOS), at the end of sentence (EOS), Not Beginning of Sentence (!BOS), Not Ending of Sentence (!EOS).

5. Token prefixes and suffixes features

   We extract the prefixes and suffixes for each word. These include the first four prefixes and the last four prefixes for each word.

6. Bigram features

   For each sentence we extracted bigrams containing only nouns and adjectives.

3.4.2 Conditional Random Field (CRF)

To train the model, we used Conditional Random Field machine learning on 90 % of the dataset and we tested on 10 % of the dataset.

CRF is a machine learning model proposed by (Lafferty et al., 2001). It is widely used in word segmentation, part-of-speech tagging, chunking recognition, named entity recognition and so on.

Conditional Random Fields (CRF) model is a state-of-the-art sequence labeling method, which can use the features of documents more sufficiently and effectively. As we said earlier, non-explicit method mention extraction can be considered as a task of entity recognition to which string labeling techniques can be successfully applied.

4 Results and Discussion

The experimental results used 924 for the rule-based task and 122 sentences for the machine learning task. Table 3 shows the performance of the two tasks. As we could expect recall is very high (100 %) for the rule-based task because every sentence in the dataset contains a method keyword. Our rules were able to recognize every noun phrase and adjective phase ending with a method keyword. Precision is 85.40, which is not bad. Some of the errors came from the tagger. We have spotted these errors and we will remove them in future study For the machine learning task, precision is 81.8 percent, recall is 75 and F-score is 78.26 percent. Features such as word shape, POS, and noun-bigrams performed better than the position features. Also some of the errors came from the all-lower-case terms as they tend to be confused with similar words in the sentence. Table 3 shows the performance of both systems. We believe that the scores can be improved with better linguistic filters in the case of the first task and a better feature selection for the second task.

System	Precision	Recall	F-Measure
Rule-based	85.40	100	91.89
Machine Learning	81.8	75.00	78.26

Table 3: Precision, Recall, F-measure of the Various Methods.

Our work can be compared to (Zhang et al., 2008) which uses CRF for automatic keyword extraction from documents; they reported promising results (F-score of 51.25 percent) on a Chinese corpus.

It can also be compared to (Zhang and Wu, 2012), which uses a multi-level termhood method to extract terminology candidates from a bilingual corpus. Their system achieves an F-score of 79.6% with CRF. Both works use similar techniques to ours, but most of the terminology extraction tasks are performed using a Chinese corpus.

5 Conclusions and future work

In this paper we have first presented a simple approach to extract fine-grained method sentences from large scientific corpora. We have also explored two established techniques to automatically extract method terminologies from method sentences. Our results showed that we can extract most of these terms using simple grammatical patterns. A few other terms can be extracted with machine learning techniques. A brief study of the corpus showed that the context of the method mentions can help in the extraction of important information about the method term. Our future work will then be to use the whole corpus to extract such information that is essential in the building of NLP resources such as glossaries, ontologies and specialist lexicons.

References

Agarwal, S. and Yu, H. (2009). Automatically classifying sentences in full-text biomedical articles into introduction, methods, results and discussion. *Bioinformatics*, 25(23):3174–3180.

Bourigault, D. (1992). Surface grammatical analysis for the extraction of terminological noun phrases. In *Proceedings of the 14th conference on Computational linguistics - Volume 3*, COLING '92, pages 977–981, Stroudsburg, PA, USA. Association for Computational Linguistics.

Church, K. W. and Hanks, P. (1990). Word association norms, mutual information, and lexicography. *Computational Linguistics*, 16(1):22–29.

Dagan, I. and Church, K. W. (1994). Termight: Identifying and translating technical terminology. In *ANLP*, pages 34–40.

Daille, B. (1996). Study and implementation of combined techniques for automatic extraction of terminology. In Klavans, J. and Resnik, P., editors, *The Balancing Act: Combining Symbolic and Statistical Approaches to Language*, pages 29–36. MIT Press, Cambridge, MA.

Dunning, T. (1993). Accurate methods for the statistics of surprise and coincidence. *Comput. Linguist.*, 19(1):61–74.

Frantzi, K., Ananiadou, S., and Mima, H. (2000). Automatic recognition of multi-word terms:. the C-value/NC-value method. *International Journal on Digital Libraries*, 3(2):115–130.

Hunston, S. (2008). Starting with the small words patterns, lexis and semantic sequences. *International Journal of Corpus Linguistics*, 13(3):271–295.

Kageura, K. and Umino, B. (1996). Methods of automatic term recognition: a review. *Terminology*, 3(2):259–289.

Lafferty, J. D., McCallum, A., and Pereira, F. C. N. (2001). Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth International Conference on Machine Learning*, ICML '01, pages 282–289, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Liakata, M., Saha, S., Dobnik, S., Batchelor, C. R., and Rebholz-Schuhmann, D. (2012). Automatic recognition of conceptualization zones in scientific articles and two life science applications. *Bioinformatics*, 28(7):991–1000.

Maedche, A. and Staab, S. (2000). Mining ontologies from text. In *Proc. of Knowledge Engineering and Knowledge Management (EKAW 2000)*, LNAI 1937, pages 169–189. Springer.

Maynard, D. and Ananiadou, S. (2000). TRUCKS: a model for automatic multi-word term recognition. volume 8, pages 101—125.

Maynard, D., Tablan, V., Ursu, C., Cunningham, H., and Wilks, Y. (2001). Named entity recognition from diverse text types. In *In Recent Advances in Natural Language Processing 2001 Conference, Tzigov Chark*.

Palmer, D. D. and Day, D. S. (1997). A statistical profile of the named entity task. In *Proceedings of the fifth conference on Applied natural language processing*, ANLC '97, pages 190–193, Stroudsburg, PA, USA. Association for Computational Linguistics.

Patry, A. and Langlais, P. (2005). Corpus-based terminology extraction. In *7th International Conference on Terminology and Knowledge Engineering*, pages 313–321, Copenhagen, Denmark.

Torii, M. and Vijay-Shanker, K. (2005). Anaphora resolution of demonstrative noun phrases in medline abstracts. In *Procedings of PACLING 2005*, pages 332–339.

Tsuruoka, Y., Tateishi, Y., Kim, J.-D., Ohta, T., McNaught, J., Ananiadou, S., and ichi Tsujii, J. (2005). Developing a robust part-of-speech tagger for biomedical text. In Bozanis, P. and Houstis, E. N., editors, *Panhellenic Conference on Informatics*, volume 3746 of *Lecture Notes in Computer Science*, pages 382–392. Springer.

Velardi, P., Missikoff, M., and Basili, R. (2001). Identification of relevant terms to support the construction of domain ontologies. In *Proceedings of the ACL 2001 Workshop on Human Language Technology and Knowledge Management*, pages 5:1–5:8, Toulouse.

Vivaldi, J., Màrquez, L., and Rodríguez, H. (2001). Improving term extraction by system combination using boosting. In *Proceedings of the 12th European Conference on Machine Learning*, EMCL '01, pages 515–526, London, UK. Springer-Verlag.

Vivaldi, J. and Rodriguez, H. (2001). Improving term extraction by combining different techniques. *Terminology*, 7(1):31–48.

Xu, F., Kurz, D., Piskorski, J., and Schmeier, S. (2002). A domain adaptive approach to automatic acquisition of domain relevant terms and their relations with bootstrapping. In *3rd International Conference on Language Resources and Evaluation*, page 7pp. European Language Resources Association.

Zhang, C., Wang, H., Liu, Y., Wu, D., Liao, Y., and Wang, B. (2008). Automatic keyword extraction from documents using conditional random fields. In *Journal of Computational Information Systems*, pages 1169–1180. Binary Information Press.

Zhang, C. and Wu, D. (2012). Bilingual terminology extraction using multi-level termhood. *The Electronic Library*, 30(2):295–309.

Zhang, Q. L., Q. L. and Sui, Z. F. (2007). Measuring termhood in automatic terminology extraction. In *Proceedings of the International Conference on Natural Language Processing and Knowledge Engineering*, pages 328–335. IEEE Press, Piscataway, NJ.

Context-Enhanced Personalized Social Summarization

Po Hu[1,2]*, Donghong Ji*[1]*, Chong Teng*[1] *and Yujing Guo*[1]
(1) Computer School, Wuhan University, China
(2) Computer School, Central China Normal University, China
phu@mail.ccnu.edu.cn, donghong_ji2000@yahoo.com.cn,
tchong616@126.com, yujingguo.ximo@gmail.com

ABSTRACT

This work investigates an interesting and challenging task in summarization, i.e., personalized social summarization, which aims to adapt summarization result of a specified document to an intended user based on his interests inferred from social context implicitly. Most existing summarization systems generate a uniform version of summary for different users no matter who is reading or generate personalized summaries employing only the local information in the document and the user profile. This paper proposes a novel unsupervised approach by making use of enhanced social context to aid personalized summary generation. In the proposed method, document expansion, user expansion, and implicit induction of the intended user's interest aspects are achieved simultaneously by adopting a fuzzy tripartite clustering algorithm. And both the informativeness of sentences and the user's interest aspects are incorporated in a unified ranking process. Preliminary experimental results on a social tagging dataset validate the effectiveness of the proposed approach.

KEYWORDS: Personalized social summarization, social context, fuzzy tripartite clustering

Proceedings of COLING 2012: Technical Papers, pages 1223–1238,
COLING 2012, Mumbai, December 2012.

1 Introduction

With the dramatic growth of the Internet, people are overwhelmed by a large number of accessible documents. In recent years, document summarization has become one of the most important research topics, which aims to address such dilemma by automatically capturing the essential content from document(s) and presenting it to a human reader in a succinct and friendly form. However, most existing summarization methods generate the same summary for different users, regardless of the interests of the readers for whom they are intended. These "one size fits all" methods may perform well in general but may not meet the needs of individuals.

Now with the rapid growth of social networking services like Delicious[1], CiteULike[2], and Flickr[3], users are no longer passive consumers of web contents. They can create contents and add metadata. Similarly, web documents no longer exist on their own and they are naturally associated with other documents and diverse users. All these information can be considered as the potential data source for document understanding and personalization.

For generating a personalized summary, traditional methods usually require that a user explicitly provides his interest aspects, such as specifying the categories he prefers (Díaz and Gervás, 2007) or clicking a subset of sentences in a document according to his interests (Yan et al., 2011). However, most users are reluctant to provide such information, thus it is more meaningful to infer a user's interests implicitly.

To address these concerns, we present an unsupervised approach for personalized summarization. The underlying assumption is that it is beneficial to understand both a single document and a single user better if appropriate social context can be leveraged under some constraints. In this work, the expanded social context used to infer users' interests and enrich document's content is highly selective, which comes from the most similar users and documents. We explored how the size of social context influences the summarization performance, and further demonstrated that appropriate contextual information can ensure better quality and personalization of summaries.

To the best of our knowledge, implicitly exploiting social contextual information to collaboratively summarize single document in a personalized way has been rarely investigated in the summarization community. In this work, we propose a novel personalized summarization approach which benefits from three important elements: the interests of like-minded users, the contents of topic-related documents, and semantically-related tags. In the approach, a fuzzy tripartite clustering algorithm is proposed and a multi-manifold ranking algorithm is adopted to generate personalized summary by considering both the informativeness of sentences and the intended user's interests.

The main contribution of this paper is summarized as follows:

1. we investigate an interesting and challenging summarization task, i.e., personalized social summarization.
2. we propose a novel approach making use of expanded social context to capture the intended user's interests, enrich the target document's content, and collaboratively summarize the document in a personalized way.

[1] http://delicious.com/
[2] http://www.citeulike.org/
[3] http://www.flickr.com/

3. we conduct preliminary experiments to validate the effectiveness of the proposed approach on a social tagging dataset and investigate how the expanded social context improves the performance of personalized summarization.

The remainder of the paper is organized as follows. The related work is introduced in Section 2. The proposed summarization approach is described in Section 3. Experimental results are shown in Section 4. Section 5 is our conclusion and future work.

2 Related work

Document summarization has been widely studied for many years. To date, various approaches have been proposed, and our work is under the framework of extractive summarization.

The vast majority of extractive methods identify which sentences are important by making use of unsupervised or supervised learning techniques. In unsupervised methods, feature-based ranking methods are usually based on a combination of linguistic and statistical features such as term frequency, sentence position, cue words, stigma words, lexical chains, rhetorical structure, topic signatures (Luhn, 1969; Lin and Hovy, 2000), etc. Clustering-based methods usually select one or more representative sentences from each subtopic to produce a summary with minimized redundancy and maximized coverage (Nomoto and Matsumoto, 2001). Graph-based methods have been shown to work well and are becoming more and more popular. LexRank (Erkan and Radev, 2004) and TextRank (Mihalcea and Tarau, 2004) are representative methods adopting models like PageRank and HITS to estimate the importance of sentences via the computation of the stationary distribution of a Markov chain or a mutual reinforcement process (Zha, 2002).

For supervised methods, summarization is often regarded as a classification task or a sequence labeling task at sentence level, and many supervised learning algorithms have been investigated including Hidden Markov Models (Conroy and O'leary, 2001), Support Vector Regression (You et al., 2011), Factor Graph Model (Yang et al., 2011), etc. However, such a supervised learning paradigm often requires a large amount of labeled data, which are not available in most cases.

With the rapid growth of online information, some work has began to employ context to aid summarization, such as contents from external documents (Wan and Yang, 2007) or cited papers (Mei and Zhai, 2008; Qazvinian and Radev, 2010), click-through data or search logs (Sun et al., 2005), and social tags (Qu and Chen, 2009; Hu et al., 2011), comments (Hu et al., 2008) or discussing tweets (Yang et al., 2011), etc.

However, such methods so far are usually designed for generic summarization and do not take into account the impact of users' interests on summary generation. Besides, in the existing studies, personalized summarization is often conducted with the help of a query (Sun, 2008; You et al., 2011) or a static user profile (Díaz and Gervás, 2007), and most studies only use the local content from target document(s) or the user profile, with little attention paid to the rich social contextual information affiliated with them.

Currently, an increasing number of social websites allow users to enrich the source content. Many documents are now presented together with various feedback information in the form of social tags, comments, or ratings, etc. These usage data can be exploited for personalized summarization since they provide a natural channel to reveal users' interests implicitly.

Based on the analysis above, we investigate a challenging task in summarization, i.e., personalized social summarization, and propose an unsupervised approach for this task. The characteristic of our proposed approach is that it can leverage topic-related documents, like-minded users, and semantically-related tags to infer the intended user's interests implicitly and collaboratively summarize the target document in a personalized context-aware way.

3 Personalized social summarization

3.1 Overview

Given a user u ($u \in U$), a document d ($d \in D$), and related social tagging data G (G = (D, U, T, R)), personalized social summarization aims to generate a tailored summary of d for u. Here D, U, and T are documents, users, and tags respectively. R is a ternary relation between them, which denotes the set of annotations of each tag in T to a document in D by a user in U.

In most social tagging sites, many documents have been annotated by few tags and most users have only annotated few documents. In this case, existing tag-based summarization methods will fail to produce a personalized summary (Boydell and Smyth, 2007; Zhu et al., 2009), since the user-related tags may be absent for that document. To address it, we propose to expand both the target document and the intended user with appropriate social context so that the important parts in the document that the intended user may care about can be identified from context.

The general framework of our proposed approach consists of three major steps.

Step1. Social context identification by document expansion and user expansion

In this step, the given document d is expanded to a small document set $D_d^{(c)}$ by adding a small number of topic-related documents, and the intended user u is expanded to a small user community $U_u^{(c)}$ by adding a small number of like-minded users. Here $D_d^{(c)}$ and $U_u^{(c)}$ are identified as the expanded social context, which is based on the intuition that we would better know a user if we know more like-minded users close to him and we would better understand a document if we read more topic-related documents close to it.

Step2. User interest discovery

In this step, the interest aspects of the intended user u are inferred from the social context $D_d^{(c)}$ and $U_u^{(c)}$ by making use of the social tagging information that the like-minded users gave to the topically related documents.

Step3. Personalized summary generation

In this step, given the expanded document context $D_d^{(c)}$ and the inferred interest aspects, the relationships of all sentences in $D_d^{(c)}$ against each interest aspect are incorporated in a unified ranking process to extract personalized informative sentences from document d.

3.2 Social context identification

In this study, the related social tagging data G, which the target document d and the intended user u belong to, is firstly collected. Then it is used to identify the social context, which can be demonstrated by the example in Figure 1.

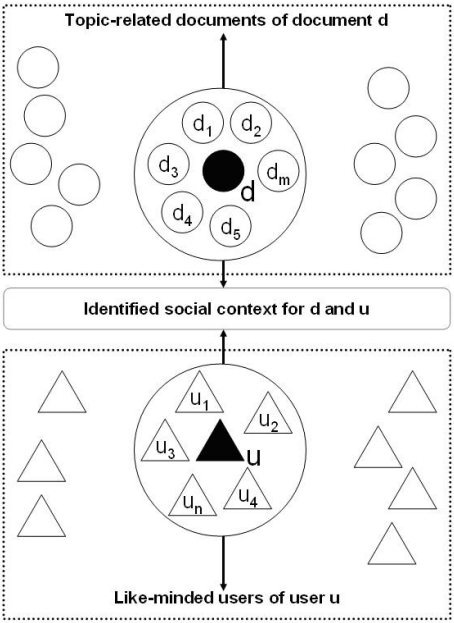


FIGURE 1 – Social context for document d and user u.

Since content-related documents are usually annotated with semantically-related tags by users with similar interests, it is feasible to find topic-related documents, like-minded users, and semantically-related tags simultaneously by clustering them collaboratively (Lu et al., 2009). Therefore, we propose a fuzzy tripartite clustering algorithm to solve the fuzzy partition issues peculiar in personalized social summarization: a document may cover different subtopics, a user may have diverse interest aspects, and a tag may be a polysemy. The potential benefit of our algorithm is that it can make use of the inherent cluster structure and interactions among the different types of objects to cluster them simultaneously and flexibly. By the algorithm, an object can have a fuzzy membership across clusters and each cluster can be represented by a committee, i.e., a small number of objects with the highest membership for the cluster.

Before clustering, each type of object (e.g., document, user, and tag) is first represented by a combined vector. A document d_i is represented by D_i consisting of two components with one denoting user link vector and the other denoting tag link vector. $D_i = (D_i^{(U)}, D_i^{(T)})$, $D_i^{(U)} = (x_{ij}^{(U)} \mid j=1,2,...,|U|)$, $D_i^{(T)} = (x_{ik}^{(T)} \mid k=1,2,...,|T|)$, where $x_{ij}^{(U)}$ denotes the times that d_i is annotated by user u_j, $|U|$ denotes the total number of users, $x_{ik}^{(T)}$ denotes the times that d_i has been annotated with tag t_k, and $|T|$ denotes the total number of tags. User and tag can be represented likewise. Accordingly, the similarity between any two objects of the same type can then be computed by the linear combination of the similarity between their combined vectors. Our proposed fuzzy tripartite clustering algorithm is shown as follows.

Algorithm 1: The fuzzy tripartite clustering algorithm.

Input:

G=(D, U, T, R): the related social tagging data that document d and user u belong to;

N_{dc}, N_{uc}, N_{tc}: the predefined number of document clusters, user clusters, and tag clusters.

Output:

The fuzzy cluster assignments of documents, users, and tags: M_d^*, M_u^*, and M_t^*, where each object is affiliated with a list of membership values with respect to various clusters.

Method:

Initialize the fuzzy partition matrices of documents, users, and tags $M_d^{(0)} = \left[u_{(d)_{i,j}}\right]_{|D|\times N_{dc}}$, $M_u^{(0)} = \left[u_{(u)_{i,j}}\right]_{|U|\times N_{uc}}$ and $M_t^{(0)} = \left[u_{(t)_{i,j}}\right]_{|T|\times N_{tc}}$ randomly, such that $0 \leqslant u_{(d)_{i,j}}$, $u_{(u)_{i,j}}$, $u_{(t)_{i,j}} \leqslant 1$ and $\sum_p u_{(d)_{i,p}} = 1$, $\sum_p u_{(u)_{i,p}} = 1$, $\sum_p u_{(t)_{i,p}} = 1$. And then generate the initial committee of each cluster and set k = 1.

Repeat:

For each type of object (e.g. document, user, and tag) **do**

Calculate the centroid vector $c^{(k)}$ of each cluster based on the current committee of this cluster according to formula (1).

For each object **do**

Update the object's membership values $u_{i,j}^{(k)}$ to $u_{i,j}^{(k+1)}$ by the normalized Cosine similarity value between the i-th object and the centroid of the j-th fuzzy object cluster formed in the k-th iteration. Here the computation of the similarity value can be considered as the membership function.

End For

Regenerate the committee of each cluster.

End For

k = k + 1

Until $\max_{i,j}\{|u_{i,j}^{(k+1)} - u_{i,j}^{(k)}|\} < \varepsilon$ or k>specified threshold.

In the algorithm, $u_{i,j}$ denotes the membership value for the i-th object in the j-th cluster, ε is the termination criterion, which is set as 0.01 in this study. The threshold of maximum iteration number is set at k=50. Since the 'true' numbers of document clusters, user clusters, and tag clusters are hard to predict in advance, we simply set N_{dc}, N_{uc}, and N_{tc} to the square root of the total number of documents, users, and tags in the related social tagging data respectively. The committee of each cluster is determined by selecting 30 percent of objects which have the highest membership values for the cluster from all the objects of the same type. In the following demonstration, we will take documents as examples of objects.

Let C_d represent a fuzzy document cluster and $C_d(c)$ represent the committee of C_d ($C_d(c) \subset C_d$). Since each document can be represented by a user link vector and a tag link vector, we will first consider the user link vectors of these documents. The value of the centroid vector of the document cluster C_d at the user dimension u_μ can be calculated by formula (1).

$$Centroid_{C_{d_u}}^{(U)} = \frac{\sum_{d_i \in C_d(c), u_j \in C_u(c)} x_{ij}^{(U)}}{|C_d(c)| * |C_u(c)|}, \ (u_u \in C_u(c)) \qquad (1)$$

where $C_u(c)$ is the committee of a fuzzy user cluster for which user u_u has its highest membership value, u_j is any user in $C_u(c)$, and d_i is any document in $C_d(c)$. $x_{ij}^{(U)}$ denotes the times that d_i is tagged by u_j. The value of the centroid vector at the tag dimension can be calculated similarly. Accordingly, the similarity between a document d_i and the centroid of a fuzzy document cluster C_j can be calculated by the linear combination of the Cosine similarity between their user link vectors and tag link vectors.

After clustering, we get the cluster assignments of documents, users, and tags, where each document (user, tag) gets a membership value for each cluster. Next, the given document d is expanded to the document context $D_d^{(c)} = \{d, d_1, d_2, ..., d_m\}$ by adding m topic-related documents with highest membership value for the cluster that d belongs to most likely. Similarly, the intended user u is expanded to the user context $U_u^{(c)} = \{u, u_1, u_2, ..., u_n\}$ by adding n like-minded users with highest membership value for the cluster that u belongs to most likely. $D_d^{(c)}$ and $U_u^{(c)}$ are identified as the expanded social context, aiming to boost information shared by topic-related documents and users with similar interests for personalized summary generation. We will further discuss the variation of performance with different assignment of m and n in Section 4.

3.3 User interest discovery

As a common form of users' online behavior, users' social tagging activities are good at reflecting their interests about document's contents and expressing the general concepts of documents. Previous work has studied the utility of social tags for user interest modeling (Li et al., 2008) and confirmed that a set of semantically related tags can characterize users' interests well (Zhou et al., 2010).

Considering that a user may have diverse interest aspects on a given document and the combination of topic-related documents and like-minded users can provide rich global contextual clues, we propose to model the interests of a user u about a document d by the social tags which have been used to annotate the documents in the document context $D_d^{(c)}$ by the users from the user context $U_u^{(c)}$. The intuitive idea is that users who annotate similar documents may have common interests on the topic shared by these documents, so the tags used by these like-minded users may reveal the latent interests of the intended user about this kind of topic.

According to the output of the fuzzy tripartite clustering algorithm, the tags on $D_d^{(c)}$ annotated by $U_u^{(c)}$ may belong to different tag clusters with varying degrees of membership. So we assign these tags into the clusters for which they have highest membership values, and then we can model the intended user's interests by the tag clusters with each indicating one unique interest aspect of the user. Here each cluster consists of one or more semantically-related tags, corresponding to the committee of the relevant tag cluster.

Formally, the intended user's interests on the given document can be represented as UM_u, which can be regarded as multiple subtopics for modelling user's interest aspects. $UM_u = \{p_i \mid 1 \le i \le N_{tu}\}$, where N_{tu} is the number of interest aspects for user u, and p_i is the user's i-th interest aspect for the given document.

3.4 Personalized summary generation

Based on the identified interest aspects, we further adopt multi-manifold ranking algorithm to fuse the sentence relationships against different aspects in a unified ranking process, which has

performed successfully in the multi-subtopic summarization task (Wan, 2009). In this study, we collaboratively summarize the target document by multiple topic-related documents within the document context, since topic-related documents can provide more clues from global context to aid extracting salient summary sentences from the specified document.

Formally, given the sentence set $S=\{s_i \mid 1\le i\le n\}$ of the document context $D_d^{(c)}$ for document d and the k-th interest aspect p_k of user u, an affinity matrix $W_k = \left[w_{(k)_{i,j}}\right]_{(n+1)\times(n+1)}$ can be built firstly to represent both the relationships among all the n sentences in $D_d^{(c)}$ and the relationship between each sentence and p_k. Then W_k is symmetrically normalized by $S_k = D_k^{-1/2} \cdot W_k \cdot D_k^{-1/2}$. Here $w_{(k)_{i,j}}$ is computed by the Cosine similarity between the i-th sentence and the j-th sentence. D_k is the diagonal matrix with the (i,i)-element equal to the sum of the i-th row of W_k. In this study, there will be N_{tu} affinity matrices in total since the number of discovered interest aspects for user u is N_{tu}.

Let F represent a ranking function that assigns each element s_i ($0\le i\le n$) a ranking score f_i. It can be regarded as a vector $F = [f_0,\ldots, f_n]^T$. We also define a prior vector $Y = [y_0,\ldots, y_n]^T$, in which $y_0=1$ for the k-th interest aspect p_k and $y_i=0$ ($1\le i\le n$) for all the remaining sentences.

Next, we can rank all the sentences by adopting the multi-manifold ranking algorithm (Wan, 2009), in which the ranking function F is to be learned from W_k ($1\le k\le N_{tu}$) and Y. In this study, the constraints from S_k ($1\le k\le N_{tu}$) and Y are naturally fused in a regularized optimization framework defined by the following cost function.

$$Q(F) = \sum_{k=1}^{N_{tu}} \left[u_k \cdot \sum_{i,j=0}^{N_{tu}} \left(w_{(k)}\right)_{ij} \left| \frac{1}{\sqrt{(D_k)_{ii}}} f_i - \frac{1}{\sqrt{(D_k)_{jj}}} f_j \right|^2 \right] + \eta \cdot \sum_{i=0}^{n} |f_i - y_i|^2 \quad (2)$$

where u_k ($1\le k\le N_{tu}$) and η ($0<\eta\le 1$) are the trade-off between the smoothness constrains. $0<u_k$, $\eta<1$ and $\sum_{k=1}^{N_{tu}} u_k + \eta = 1$.

Based on the optimization framework, the optimal ranking function F^* can be achieved when Q(F) is minimized. In practice, the following iterative form shown in the formula (3) is more commonly used to get the ranking function, in which $F^{(0)}$ is set to Y and we have $F^{(*)} = \lim_{t\to\infty} F^{(t)}$.

$$F^{(t+1)} = \sum_{k=1}^{N_{tu}} u_k S_k F^{(t)} + (1 - \sum_{k=1}^{N_{tu}} u_k) Y \quad (3)$$

Through the above ranking process, the ranking scores, which denote the user-biased informativeness of sentences, can be obtained. Finally, those sentences highly overlapping with other informative sentences are penalized to remove redundancy (Wan and Yang, 2007), and the sentences with high overall scores are chosen from document d into the summary.

4 Experiments

4.1 Dataset

Since there is no benchmark dataset available for the task of personalized social summarization, we collected data from Delicious, one of the most popular social tagging websites. Specifically, we extracted a set of web documents, bookmark tags, and the users who bookmarked these documents to serve as the experimental dataset.

Starting with predefined seed tags, we extracted the top bookmarked documents for each tag and extracted the users and tags used to annotate each of the documents. The result is a collection consisting of 204 bookmarked documents and 2186 unique social tags that were used to annotate these documents by 1696 users. To guarantee the genre consistency, all the documents were crawled from news sources such as CNN, BBC, New York Times, etc.

4.2 Evaluation methods

In this paper, both manual evaluation method and automatic evaluation method are adopted. For each document in the dataset, we randomly select one to five users as the intended users from all the users who annotated the document with multiple social tags.

4.2.1 Manual evaluation

First, we must admit that it would be better to use personalized reference summaries for evaluation. However, it would be quite difficult to get personalized summaries from the actual users of Delicious. The alternative way is to get the external judgments from several judges and take average of their ratings so that we know that multiple people would consider that this summary is relevant or tailored for the intended user to a certain extent. How to develop a better test collection for personalized summarization from the perspective of social context is an important future direction of our research.

In this study, three evaluators are requested to express their judgments over all automatically generated summaries based on both the content they deem to be important for the target document and how "personal" each one is according to the interests of the intended user. We provide each evaluator the intended user's background knowledge collected by calling the official Delicious.com API and parsing its RSS feeds. The provided information includes all the open document bookmarks of intended users and all the tags they used to annotate the documents including the target document to be summarized. Evaluators can also access the content of the corresponding document by clicking the URL in each bookmark.

In the evaluation process, evaluators are instructed to give an overall score to each summary. The overall score reflects the comprehensive quality of a summary including not only the evaluation for the general content of the generated summary but also the degree of compliance with the intended user's personalized interests and foci.

All the judgment scores are rated in a 5-point scale, where "1" for "very poor", "2" for "poor", "3" for "barely acceptable", "4" for "good", and "5" for "very good". Evaluators are allowed to judge at any scores between 1 and 5, e.g. 3.5.

4.2.2 Automatic evaluation

Considering that manual evaluation is generally time consuming and labour-intensive, we also adopt automatic evaluation strategy.

For each intended user of the target document, we randomly divide the social tags he assigned to the document into two approximately equal parts: a training set and a test set. The former is used to generate personalized social summary on the document for the user, and the latter is used to evaluate the generated summary based on the recall against the tags in the test set, making sure to remove the tags occurring in the training set from the test set.

The idea of this kind of evaluation strategy is to look for overlaps between the generated personalized social summary and those unseen tags used by the intended user on the given document, since the tags in the test set correspond to an alternative, but previously unseen, point of interests for the intended user with respect to the target document.

The automatic evaluation experiments were conducted in the cross validation procedure, and the average recall score was recorded. Intuitively, the higher the average recall score is, the more the generated summaries are in line with the interests of the intended users.

4.3 Baselines

In the experiments, we compare our proposed approach with several baseline methods. For fair comparison, we conduct the same preprocessing for all the methods including sentence segmentation, word stemming, and redundancy removing.

Random: It extracts sentences randomly from each document.

OTS: It is an open source summarizer integrating shallow NLP techniques with statistical word frequency analysis for sentence scoring (Nadav, 2003).

MEAD: It ranks sentences according to the combination of features including centroid value, positional value, and first-sentence overlap (Radev et al., 2000).

LexRank: It first constructs a sentence affinity graph based on the Cosine similarity between sentences in a document, and then extracts a few informative sentences based on eigenvector centrality (Erkan and Radev, 2004).

DcontextLexRank: It is an extension of the original LexRank method by firstly ranking sentences on the document context which the target document belongs to, and then extracting sentences with highest ranking scores from the target document.

PSocialSum: It is our proposed approach using expanded social context to capture the intended user's interests, enrich the target document's content, and collaboratively summarize the target document in a personalized way.

4.4 Overall comparison results

4.4.1 Parameter settings

The parameters m and n, i.e., the number of expanded topic-related documents in the document context $D_d^{(c)}$ and the number of expanded like-minded users in the user context $U_u^{(c)}$, are set as

the number of elements in the corresponding committee of the cluster that document d or user u belongs to most likely.

In the multi-manifold ranking process of our approach, parameters u_k and η are the smoothness constraint and fitting constraint respectively, which control the trade-off between the impact from S_k (i.e., both the relationships among all the sentences in the document context and the relationship between each sentence and the k-th interest aspect of user u) and the impact from Y (i.e. the prior vector set for the k-th interest aspect and all the remaining sentences). In the following experiments, the regularization parameter η for the fitting constraint is fixed at 0.01, the same as in (Wan, 2009), and u_k is set to the normalized Cosine similarity between the corresponding vectors of p_k and $D_d^{(c)}$.

4.4.2 Experimental results

In the experiments, for each document, we generate multiple different personalized summaries for each of the intended users by our approach. For comparison purpose, each document in the dataset is also summarized using all the baseline methods described in Section 4.3.

First, we conducted the manual evaluation and the average overall scores of multiple evaluators on all the generated summaries are listed in Table 1.

Method	Average Overall Score
Random	1.2
OTS	2.1
MEAD	2.2
LexRank	2.3
DcontextLexRank	2.4
PSocialSum	3.5

TABLE 1 –The average overall scores of multiple evaluators.

From Table 1, it can be found that Random has the worst summarization performance.

LexRank and DcontextLexRank perform better than those of MEAD and OTS. This is mainly because both LexRank and DcontextLexRank make use of the inter-relationship between sentences to rank them globally, while MEAD and OTS only depend on the combination of some local features.

DcontextLexRank outperforms LexRank in our experiments, which indicates the use of appropriate document context for sentence ranking is an improvement over the use of single document alone which lacks the support of external clues from the similar documents.

Note that all these baseline methods generate the summary based on either the given document itself or the document context, regardless of the intended user's interests. Our proposed approach shows significantly better performance on evaluators' ratings. And the rating difference between PSocialSum and other baselines is significant at the 95% statistical confidence level in all cases. This indicates that consideration of user's interests is critical for generating a better personalized

summary, and the improvement achieved is mainly attributed to the personalization aspect as well as informative content.

We also find that the evaluator judgments on MEAD, LexRank, and DcontextLexRank are of little significant difference at the 95% confidence interval, which illustrates that the general summaries generated by these comparable baselines can convey the important information of a document, and different evaluators may have some agreement on the quality of its content, although all of these methods do not consider the intended user's interest at all.

Next, we conducted the automatic evaluation by computing the average recall scores against the tags in the corresponding test set for all the resulting summaries. The process is repeated across multiple different random splits of training and test set. The average recall scores are reported in Table 2.

Method	**Average Recall Score**
Random	0.194
OTS	0.282
MEAD	0.287
LexRank	0.292
DcontextLexRank	0.294
PSocialSum	0.338

TABLE 2 – The average recall scores.

From Table 2, we see that the summarization performance of PSocialSum is consistently better than those of other baselines. Such results also demonstrate that by leveraging part of the social tagging information of the intended users, we can generate better summaries which are more in accordance with the latent interests of them, compared to other summarizers which generate the static summaries ignoring the social contextual information.

4.5 Impact of parameters

In this section, to investigate how the size m and n of the expanded topic-related documents and the expanded like-minded users influence the performance of PSocialSum, we conduct the following experiments with different values.

Considering that m and n in this study are dynamically related to the predefined percentage of objects which have the highest membership values for the cluster from all the objects of the same type, in the experiment, we set the predefined percentage value related with m and n ranging from 10% to 80% with step length 10%, indicating the corresponding percentage of documents or users are selected for the expanded document set or user set.

Figure 2 shows the average recall scores against the tags in the test set for PSocialSum with different percentage values.

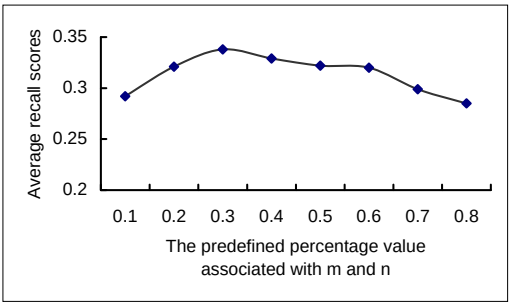


FIGURE 2 –The average recall scores of PSocialSum
vs. the predefined percentage value related with m and n.

From Figure 2, it can be seen that when the percentage value increases from 10% to 30%, the recall increases gradually, and reaches the global maximum when it is set to 30%. When we adjust the percentage value from 30% to 80%, the recall starts to decay. The result demonstrates that appropriate document context and user context are beneficial for improving personalized summarization performance, yet a large size of the expanded context may deteriorate the performance because it may include a lot of irrelevant information even noise.

Conclusion

In this paper, we present a study of personalized social summarization, and propose a novel unsupervised approach. The approach makes use of expanded social context to capture the intended user's interests, enrich the target document's content, and collaboratively summarize the target document in a personalized context-aware way. Preliminary experimental results demonstrate the effectiveness of the proposed approach.

In practice, the dimensions and variability of users, documents, and tags from most social network websites may be quite high, so in future work, we plan to combine link structure association analysis and feature selection to effectively deal with high-dimensional online tripartite clustering dynamically. And more social contextual information such as social relationships among users will also be investigated. For simplicity, this method represents each object with a vector of two sets of features, and this kind of representation would inevitably result in information loss to a certain extent. Therefore, we plan to try better alternatives such as hypergraph or tensor model, and make effort to improve the existing work on content or social network-based user interest modeling. Furthermore, it would be more convincing to resort to crowdsourcing technique to evaluate the proposed approach by a large number of real users on the social tagging websites and on larger-scale social data set.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (No. 61133012, 61173062, 61070082, 61070083, 61070243), the major program of the National Social Science Foundation of China (No. 11&ZD189), and the Post-70s Scholars Academic Development Program of Wuhan University.

References

Boydell, O. and Barry, S. (2007). From social bookmarking to social summarization: an experiment in community-based summary generation. In *Proceedings of the 12th International Conference on Intelligent User Interfaces (IUI 2007)*, pages 42-51, ACM, New York, NY.

Conroy, J.M. and Oleary, D.P. (2001). Text summarization via hidden markov models. In *Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2001)*, pages 406-407, ACM, New York, NY.

Díaz, A. and Gervás, P. (2007). User-model based personalized summarization. *Information Processing and Management*, 43(6):1715-1734.

Erkan, G. and Radev, D.R. (2004). LexPageRank: prestige in multi-document text summarization. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing (EMNLP 2004)*.

Hu, M.S., Sun, A.X., and Lim, E.P. (2008). Comments-oriented document summarization: understanding documents with users' feedback. In *Proceedings of the 31th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2008)*, pages 291-298, ACM, New York, NY.

Hu, P., Sun, C., Wu, L.F., Ji, D.H., and Teng, C. (2011). Social summarization via automatically discovered social context. In *Proceedings of the 5th International Joint Conference on Natural Language Processing (IJCNLP 2011)*, pages 483-490.

Li, X., Guo, L., and Zhao, Y.H. (2008). Tag-based social interest discovery. In *Proceedings of the 17th International Conference on World Wide Web (WWW 2008)*, pages 675-684, ACM, New York, NY.

Lin, C.Y. and Eduard, H. (2000). The automated acquisition of topic signatures for text summarization. In *Proceedings of the 17th Conference on Computational Linguistics (COLING 2000)*, pages 495-501, Association for Computational Linguistics, Stroudsburg, PA.

Lin, C.Y. and Eduard, H. (2003). Automatic evaluation of summaries using n-gram co-occurrence statistics. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology (NAACL 2003)*, pages 71-78, Association for Computational Linguistics, Stroudsburg, PA.

Lu, C.M., Chen, X., and Park, E. K. (2009). Exploit the tripartite network of social tagging for web clustering. In *Proceeding of the 18th ACM Conference on Information and Knowledge Management (CIKM 2009)*, pages 1545-1548, ACM, New York, NY.

Luhn, H. P. (1969). The automatic creation of literature abstracts. *IBM Journal of Research and Development*, 2(2):159-165.

Mei, Q.Z. and Zhai, C.X. (2008). Generating impact-based summaries for scientific literature.

In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL 2008)*, pages 816–824.

Mihalcea, R. and Tarau, P. (2004). TextRank: bringing order into texts. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing (EMNLP 2004)*.

Nadav, R. (2003). The open text summarizer. http://libots.sourceforge.net/.

Nomoto, T. and Matsumoto, Y. (2001). A new approach to unsupervised text summarization. In *Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2001)*, pages 26-34, ACM, New York, NY.

Qazvinian, V. and Radev, D.R. (2010). Identifying non-explicit citing sentences for citation-based summarization. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics (ACL 2010)*, pages 555-564, Association for Computational Linguistics, Stroudsburg, PA.

Qu, Y. and Chen, Q.X. (2009). Collaborative summarization: when collaborative filtering meets document summarization. In *Proceedings of the 23rd Pacific Asia Conference on Language, Information and Computation (PACLIC 2009)*, pages 474-483.

Sun, J.T., Shen, D., Zeng,H.J., Yang, Q., Lu, Y.C., and Chen, Z. (2005). Web-page summarization using clickthrough data. In *Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2005)*, pages 194-201, ACM, New York, NY.

Sun, P. (2008). Personalized summarization agent using non-negative matrix factorization. In *Proceedings of the 10th Pacific Rim International Conference on Artificial Intelligence (PRICAI 2008)*, pages 1034-1038.

Wan, X.J. and Yang, J.W. (2007). Single document summarization with document expansion. In *Proceedings of the 22nd National Conference on Artificial Intelligence (AAAI 2007)*, pages 931-936.

Wan, X.J. (2009). Topic analysis for topic-focused multi-document summarization. In *Proceedings of the 18th ACM Conference on Information and Knowledge Management (CIKM 2009)*, pages 1609-1612, ACM, New York, NY.

Yan, R., Nie, J.Y., and Li, X.M. (2011). Summarize what you are interested in: an optimization framework for interactive personalized summarization. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing (EMNLP 2011)*, pages 1342–1351.

Yang, Z., Cai, K.K., Tang, J., Zhang, L., Su, Z., and Li, J.Z. (2011). Social context summarization. In *Proceedings of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2011)*, pages 255-264, ACM, New York, NY.

You, O.Y., Li, W.J., Li, S.J., and Lu, Q. (2011). Applying regression models to query-focused multi-document summarization. *Information Processing and Management*, 47(2):227-237.

Zha, H.Y. (2002). Generic summarization and keyphrase extraction using mutual reinforcement principle and sentence clustering. In *Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2002)*, pages 113-

120, ACM, New York, NY.

Zhou, C., Ma, H., Lyu, M.R., and King, I. (2010). UserRec: a user recommendation framework in social tagging systems. In *Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence (AAAI 2010)*, pages 1486-1491.

Zhu, J.Y., Wang, C., He, X.F., Bu, J.J., Chen, C., Shang, S.J., Qu, M.C., and Lu, G. (2009). Tag-oriented document summarization. In *Proceedings of the 18th International Conference on World Wide Web (WWW 2009)*, pages 1195-1196, ACM, New York, NY.

Tweet Ranking Based on Heterogeneous Networks

Hongzhao HUANG[1] *Arkaitz ZUBIAGA*[1] *Heng JI*[1]
Hongbo DENG[2] *Dong WANG*[2] *Hieu LE*[2] *Tarek ABDELZAHER*[2]
Jiawei HAN[2] *Alice LEUNG*[3] *John HANCOCK*[4] *Clare VOSS*[5]
(1) COMPUTER SCIENCE DEPARTMENT AND LINGUISTICS DEPARTMENT,
QUEENS COLLEGE AND GRADUATE CENTER,
CITY UNIVERSITY OF NEW YORK, New York, NY, USA 11367
(2) COMPUTER SCIENCE DEPARTMENT, UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN,
Urbana-Champaign, IL, USA 61820
(3) RAYTHEON BBN TECHNOLOGIES, Cambridge, MA, USA 02138
(4) ARTISTECH INC., Fairfax, VA, USA 22030
(5) MULTILINGUAL COMPUTING BRANCH, ARMY RESEARCH LAB, Adelphi, MD, USA 20783
{hongzhaohuang, arkaitz.zubiaga, hengjicuny}@gmail.com

Abstract

Ranking tweets is a fundamental task to make it easier to distill the vast amounts of information shared by users. In this paper, we explore the novel idea of ranking tweets on a topic using heterogeneous networks. We construct heterogeneous networks by harnessing cross-genre linkages between tweets and semantically-related web documents from formal genres, and inferring implicit links between tweets and users. To rank tweets effectively by capturing the semantics and importance of different linkages, we introduce Tri-HITS, a model to iteratively propagate ranking scores across heterogeneous networks. We show that integrating both formal genre and inferred social networks with tweet networks produces a higher-quality ranking than the tweet networks alone. [1]

Title and Abstract in Chinese

基于异构网络的微信息排序

微信息排序是一个可以过滤由用户分享的大量信息的根本任务。在这篇文章中，我们探索利用异构网络来排序微信息。我们利用来源于正规类型并且与微信息语义相关的网络文本的跨类型联接,和推理微信息和用户之间的潜在联系来构造异构网络。为了有效地捕获不同联接的语义和重要性，我们提出了Tri-HITS, 一个能够跨网络循环传播排序分数的模型。我们证明了结合来自正规类型的信息，和推理隐含的社交网络可以取得比仅靠微信息网络本身更高的排序质量。[2]

Keywords: Tweet Ranking, Heterogeneous Networks, Iterative Propagation Model.

Keywords in Chinese: 微信息排序，异构网络，循环传播模型.

[1] Related resources and software are freely available for research purposes at http://nlp.cs.qc.cuny.edu/tweetranking.zip; the system demo is at http://nlp.cs.qc.cuny.edu/tweet_summary/ground-truth-demo.xhtml.

[2] 有关的资源和程序公布在如下地址和研究相关的应用分享: http://nlp.cs.qc.cuny.edu/tweetranking.zip; 系统演示在如下地址： http://nlp.cs.qc.cuny.edu/tweet_summary/ground-truth-demo.xhtml.

Proceedings of COLING 2012: Technical Papers, pages 1239–1256,
COLING 2012, Mumbai, December 2012.

1 Introduction

Twitter has become a popular service for online communication through short messages of up to 140 characters, known as tweets. Its users produce millions of tweets a day, enabling both individuals and organizations to disseminate information about current affairs and breaking news in a timely fashion. This information is sometimes posted by users on-site or in the vicinity of events, providing first-hand accounts from a wide variety of sources. However, the sheer volume of tweets sent during events of general interest is overwhelming and hence difficult to distill for the most relevant information, while also filtering out non-informative tweets.

To facilitate finding informative and trustworthy content in tweets, it is crucial to develop an effective ranking method. This is particularly useful in emerging situations. Eyewitnesses might be live-tweeting about anything happening at ongoing events (Diakopoulos et al., 2012) such as natural disasters. To assist in these situations, we aim to develop a ranking system that organizes tweets by informativeness, so that informative tweets are readily identified, while pointless and speculative observations are filtered out. However, the definition of informativeness might vary for different points of view. Twitter users can produce diverse content ranging from news and events, to conversations and personal status updates. While personal updates and conversations might be relevant to a specific group of people, we aim to find tweets on topics that are informative to a general audience, such as breaking news and real-time coverage of on-going events. For example, during Hurricane Irene in 2011, updates from a user living in New York City about her own safety might be very informative to her friends and relatives, but not so informative to others. To produce rankings that are as relevant to as many people as possible, we define informativeness as the extent to which a tweet meets the general interest of people involved with or tracking the event.

While previous research has relied on either the text of tweets or explicit features of social network such as retweets, replies, and follower-followee relationships, we believe that such networks can be enhanced by integrating information from a formal genre. On one hand, tweets from different sources tend to contain non-informative noise such as subjective comments and conversations. Therefore it is challenging to identify salient information from tweet content alone. On the other hand, events of general interest such as natural disasters or political elections are the topics of tweets sent by many users from multiple communities which are not connected to each other. In these situations, users are likely to be unaware of each other. As a result, they fail to connect with many others on topics of mutual interest. This lack of social interaction produces networks with few explicit linkages between users, and therefore between tweets and users. The sparsity of linkages would limit the effectiveness of features extracted from social network.

In this work, we introduce Tri-HITS, a novel propagation model that leverages global information iteratively computed across heterogeneous networks constructed from web documents, tweets, and users, to rank tweets on a topic by informativeness. The model addresses the two issues mentioned above (noisy tweets, limited social connections). Using Tri-HITS, we establish cross-genre linkages between tweets and web documents, filter informal writing and noise contained in tweets, and infer implicit tweet-user relations beyond the explicit ones, so that networks are enriched by connecting users that are sharing similar contents. We propose three high-level hypotheses that motivate the presented methods of constructing heterogeneous networks of tweets, users, and web documents. The proposed model, Tri-HITS, operates iteratively over all networks incorporating the semantics and importance of different linkages. By ranking tweets about the Hurricane Irene, we demonstrate that incorporating a formal genre such as web documents, inferring implicit social networks and performing effective ranking score propagation with the proposed model can significantly improve

the ranking quality.

2 Background

In this section, we describe the basic techniques used in the paper: information networks, the ranking approach TextRank, and a widely used method for redundancy removal.

2.1 Information Networks

We define an information network as a graph $G = (V, E)$ on $X = \{X_1, X_2, ..., X_Z\}$ for Z types of vertices, where $V(G) = X_1 \cup X_2 \cup ... \cup X_Z$ and $E(G) = \langle x_i, x_j \rangle$, for $x \in X$. An edge $\langle x_i, x_j \rangle$ is a binary relation between two vertices x_i and x_j. An information network is **heterogeneous** when the vertices are from multiple distinct types of sources ($Z \geq 2$). (Deng et al., 2011) defined a text-rich heterogeneous information network as an information network that integrates a set of text documents $D = \{d_1, d_2, ..., d_n\}$ with other types of vertices, so that $V(G) = D \cup X_1 \cup ... \cup X_{Z-1}$. In this work, we construct heterogeneous networks that include web documents, tweets, and users, as shown in Figure 1.

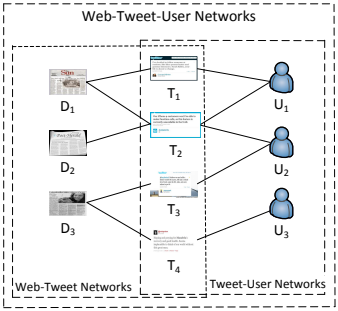


Figure 1: Web-Tweet-User heterogeneous networks

2.2 TextRank: Baseline Approach

Graph-based ranking algorithms have been widely used to generate rankings for vertices in graphs.. Adapted from PageRank (Page et al., 1998) to weighted graphs, TextRank (Mihalcea and Tarau, 2004) is a well-known ranking algorithm for homogeneous networks, which is defined as follows:

$$s(v_i) = (1-d) + d * \sum_{v_j \in In(v_i)} \frac{w_{ji} s(v_j)}{\sum_{v_k \in Out(v_j)} w_{jk}} \quad (1)$$

where v_i is a vertex with $s(v_i)$ as the ranking score, $In(v_i)$ as the set of incoming edges, and $Out(v_i)$ as the set of outgoing edges; w_{ij} is the weight for the edge between two vertices v_i and v_j. An edge exists between two vertices that represent text units when their computed shared content (cosine similarity) exceeds or equals a predefined threshold δ_{tt}.

Given its success when applied to sentence ranking for the task of extractive document summarization (Mihalcea, 2004), we choose TextRank as the **baseline** method to compute ranking scores in

tweet-only networks where edges between tweets are determined by their cosine similarity.

2.3 Redundancy Removal

Since users on Twitter can be tweeting similar information obliviously, and retweet and reply others' tweets, redundancy has been shown to be a pervasive phenomenon (Zanzotto et al., 2011). This issue has not been considered in previous works on tweet ranking (Duan et al., 2010; Huang et al., 2011). In this work, we perform a redundancy removal step to diversify top ranked tweets. To do so, we adopt the widely used greedy procedure (Carterette and Chandar, 2009; McDonald, 2007) to apply redundancy removal after the completion of each ranking method, as follows: tweet t_i in position i is removed when its cosine similarity with tweets $t_j \in [t_1, t_{i-1}]$ in more highly-ranked positions exceeds or equals a predefined threshold δ_{red}[3]

3 Motivations and Hypotheses

Next, we describe the motivational aspects and hypotheses in this work, which we aim to prove.

Hypothesis 1: *Informative tweets are more likely to be posted by credible users; and vice versa (credible users are more likely to post informative tweets).* (Duan et al., 2010; Huang et al., 2011) consider that users who have more followers, mentions, and retweets, and are listed more, are more likely to be authoritative. They used retweet, reply, user mention and follower counts to to compute the degree of authoritativeness of users; and showed that user account authority is a helpful feature for tweet ranking. However, for events of general interest involving multiple communities, users are more likely to be unaware of each other, and rarely interact. This makes it insufficient to rely on user-user networks constructed from retweet and reply interactions to compute user credibility scores. To overcome this problem, we apply a Bayesian approach to compute the credibility of users by incorporating the contents shared by them.

Hypothesis 2: *Tweets involving many users are more likely to be informative.* Having many users share similar tweets at the same time helps identify informative tweets. For example, in the context of Hurricane Irene, users were likely to share information about the Evacuation Zone when they found relevant news or events. The synchronization of information within groups has been successfully harnessed in other fields like financial trading, autonomous swarms of exploratory robots, and flocks of communicating software agents (Couzin, 2007; Saavedra et al., 2011). This idea has also been successfully exploited for event summarization from tweets (Zubiaga et al., 2012).

Hypothesis 3: *Tweets aligned with contents of web documents are more likely to be informative.* Tweets come from diverse sources, and can diverse content ranging from news and events, to conversations and personal status updates. Therefore, informative tweets tend to be interspersed with noisy and non-informative tweets. This differs from formal genres such as web documents, which tend to be cleaner. In the case of current events such as natural disasters or political elections, there are tight correlations between social media and web documents. Important information shared in social media tends to be posted in web documents. For example, the following informative tweets would rank highly because they are linked to informative web documents: " *New Yorkers, find your exact evacuation zone by your address here: http://t.co/9NhiGKG /via @user #Irene #hurricane #NY*" and "*Details of Aer Lingus flights affected by Hurricane Irene can be found at http://t.co/PCqE74V2̌01d*". As far as we know, this is the first work to integrate information from a formal genre such as web documents to enhance tweet ranking.

[3]We choose $\delta_{red} = 0.6$ as a threshold, obtained from our empirical studies with values from 0.1 to 1.0 in the development set.

4 Enhanced Approach: Tri-HITS

Based on the formulated hypotheses, we describe how Tri-HITS works.

4.1 Overview

Figure 2 depicts how Tri-HITS works. For a set of tweets on a specific topic, a rule-based filtering component is first applied to filter out a subset of non-informative tweets. For the remaining tweets, we define queries based on top terms in tweets, and use Bing Search API[4] to retrieve the titles[5] of the top m web documents for those queries ($m = 2$ for these experiments). Then we apply TextRank and a Bayesian approach that initialize ranking scores for tweets, web documents, and users. Finally, we iteratively propagate ranking scores for web documents, tweets, and users across the networks to refine the tweet ranking.

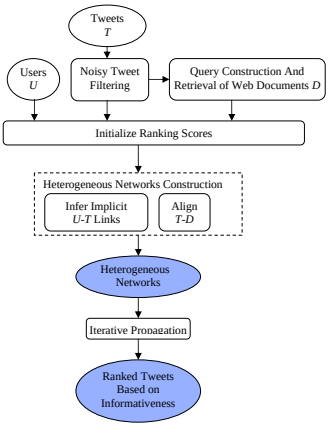


Figure 2: Overview of Tri-HITS

4.2 Filtering non-informative Tweets

Tweets are more likely to be shortened or informally written than texts from a formal genre such as web documents. Thus, a prior filtering step would clean up the set of tweets and improve the ranking quality. We observed that numerous non-informative tweets have some common characteristics, which help infer patterns to clean up the set of tweets. In our filtering method, we define several patterns to capture the characteristics of a non-informative tweet, i.e., very short tweets without a complementary URL, tweets with first personal pronouns, or informal tweets containing slang words[6]. These features have been shown to be effective in previous work on tweet ranking and information credibility (Duan et al., 2010; Castillo et al., 2011; Uysal and Croft, 2011). Our filtering component accurately filters out non-informative tweets, achieving 96.59% at precision.

[4] http://www.bing.com/toolbox/bingdeveloper/

[5] We rely on page titles, but it could be extended to the whole content of web documents straightforwardly.

[6] http://www.mltcreative.com/blog/bid/54272/Social-Media-Minute-Big-A-List-of-Twitter-Slang-and-Definitions

4.3 Initializing Ranking Scores

Initializing scores for tweets and web documents. For a set of tweets T, we first construct an undirected and weighted graph $G = (V, E)$. After removing stopwords and punctuations, the bag-of-words of each tweet t_i is represented as a vertex $v_i \in V$, and the weight for the edge between tweets is the cosine similarity using TF-IDF representations. Then, we use TextRank to compute initial scores. The same approach is used to initialize ranking scores for web documents.

Initializing user credibility scores. Based on Hypothesis 1, we define two approaches to compute initial user credibility scores. First, we construct a user network based on retweets, replies and user mentions as in (Duan et al., 2010). This results in a directed and weighted graph $G_d = (V, E)$, where V is the set of users and E is the set of directed edges. A directed edge exists from u_i to u_j if user u_i interacts with u_j (i.e., mentions, retweets, or replies to u_j). The weight of the edge is defined as N_{ij}, according to the number of interactions. In this case, we use TextRank to compute initial user credibility scores.

In addition, we also use the *Bayesian ranking* approach (Wang et al., 2011, 2012) that considers the credibility scores of tweets and users simultaneously based on Tweet-User networks. Given a set of users $U = \{u_1, u_2, ..., u_m\}$, and a set of claims $C = \{c_1, c_2, ..., c_n\}$ the users make (each claim corresponds to a cluster of tweets in this paper). We also define matrix W^{cu} where $w^{cu}_{ji} = 1$ if user u_i makes claim c_j, and is zero otherwise. Let u^t_i denote the proposition that *'user u_i speaks the truth'*. Let c^t_j denote the proposition that *'claim c_j is true'*. Also, let $P(u^t_i)$ and $P(u^t_i|W^{cu})$ be the prior and posterior probability that user u_i speaks the truth. Similarly, $P(c^t_i)$ and $P(c^t_i|W^{cu})$ are the prior and posterior probability that claim c_i is true. We define the credibility rank of a claim $Rank(c_j)$ as the increase in the posterior probability that a claim is true, normalized by prior probability $P(c^t_i)$. Similarly, the credibility rank of a user $Rank(u_i)$ is defined as the increase in the posterior probability that a user is credible, normalized by prior probability $P(u^t_i)$. In other words, we can get:

$$Rank(c_j) = \frac{P(c^t_j|W^{cu}) - P(c^t_j)}{P(c^t_j)} \quad (2)$$

$$Rank(u_i) = \frac{P(u^t_i|W^{cu}) - P(u^t_i)}{P(u^t_i)} \quad (3)$$

In our previous work, we showed that the following relations hold true regarding the credibility rank of a claim $Rank(c_j)$ and a user $Rank(u_i)$:

$$Rank(c_j) = \sum_{k \in Users_j} Rank(u_k) \quad (4)$$

$$Rank(u_i) = \sum_{k \in Claims_i} Rank(c_k) \quad (5)$$

where $Users_j$ is the set of users makes claim c_j, and $Claims_i$ is the set of claims the user u_i makes. From the above, the credibility of sources and claims can be derived as:

$$P(c^t_j|W^{cu}) = p^t_a(Rank(c_j) + 1) \quad (6)$$

$$P(u^t_i|W^{cu}) = p^t_s(Rank(u_i) + 1) \quad (7)$$

where p_a^t and p_s^t are initialization constants, which are the ratio of true claims to the total claims, and the ratio of credible users to the total users.

Then, Equation 7 is used to compute initial user credibility scores as our second approach.

4.4 Constructing Heterogeneous Networks

Next, we describe the two types of networks we build as constituent parts of heterogeneous networks:

Tweet-User networks. Based on Hypothesis 2, we expand the Tweet-User networks by inferring implicit tweet-user relations. If a user u_i posted a set of tweets T_i during a period of time, we say an implicit relation exists between u_i and a tweet t_j if the maximum cosine similarity between t_j and $t_i \in T_i$ exceeds or equals a threshold δ_{tu}.

Web-Tweet networks. Given a set of tweets T and a set of associated web documents D, we build a bipartite graph $G = T \cup D, E$, where an undirected edge with weight w_{ij}^{td} is added when the cosine similarity between $t_i \in T$ and $d_j \in D$ exceeds or equals δ_{td}. This approach creates cross-genre linkages between tweets and web documents on similar events (e.g., evacuation events).

In subsection 5.3, we will discuss the effects of parameters δ_{td} and δ_{tu}.

4.5 Iterative Propagation

We introduce a novel algorithm to incorporate both initial ranking scores and global evidence from heterogeneous networks. It propagates ranking scores across heterogeneous networks iteratively. Our algorithm is an extension of Co-HITS (Deng et al., 2009), which is limited to bipartite graphs. Co-HITS was designed to incorporate links of a bipartite graph with content from two types of objects. The intuition behind the score propagation is the mutual reinforcement to boost co-linked objects.

Let $G = (U \cup V, E)$ be a bipartite graph, in which the vertices are divided into two disjoint sets U and V, and each edge in E connects one vertex in U to another in V. We use w_{ij}^{uv} (or w_{ji}^{vu}) to denote the weight for the edge between u_i and v_j. To put all the weights between sets U and V together, we can use $W^{uv} \in \mathbb{R}^{|U|\times|V|}$ (or $W^{vu} \in \mathbb{R}^{|V|\times|U|}$) to denote the weight matrix between U and V. Note that $W^{uv} \in \mathbb{R}^{|U|\times|V|}$ is the transpose of $W^{vu} \in \mathbb{R}^{|V|\times|U|}$ as we have $w_{ij}^{uv} = w_{ji}^{vu}$. For each $u_i \in U$, a transition probability p_{ij}^{uv} is defined as the probability that vertex u_i in U reaches vertex v_j in V at the next step. Formally, it is defined as a normalized weight $p_{ij}^{uv} = \frac{w_{ij}^{uv}}{\sum_k w_{ik}^{uv}}$, such that $\sum_{j\in V} p_{ij}^{uv} = 1$. Similarly, we obtain the transition probability $p_{ji}^{vu} = \frac{w_{ji}^{vu}}{\sum_k w_{jk}^{vu}}$ and $\sum_{i\in U} p_{ji}^{vu} = 1$ for each $v_j \in V$. The Co-HITS algorithm is defined as follows:

$$s(u_i) = (1 - \lambda_u)s^0(u_i) + \lambda_u \sum_{j\in V} p_{ji}^{vu} s(v_j), \tag{8}$$

$$s(v_j) = (1 - \lambda_v)s^0(v_j) + \lambda_v \sum_{i\in U} p_{ij}^{uv} s(u_i), \tag{9}$$

where $\lambda_u \in [0, 1]$ and $\lambda_v \in [0, 1]$ are personalized parameters, $s^0(u_i)$ and $s^0(v_j)$ are initial ranking scores for u_i and v_j, and $s(u_i)$ and $s(v_j)$ denote updated ranking scores of vertices u_i and v_j. In this algorithm, the initial scores are normalized to $\sum_{i\in U} s^0(u_i) = 1$ and $\sum_{j\in V} s^0(v_i) = 1$, and the sum of updated $s(u_i)$ and $s(v_j)$ will be 1 as well.

The problem with Co-HITS in our experimental settings is the transition probability. As mentioned before, we choose cosine similarity as the weight for the edge between two vertices, and a similarity matrix W is obtained to denote the weight matrix where each entry w_{ij} is the similarity between vertex u_i and vertex v_j. Although the transition probability is a natural normalization for the weight between two vertices, it may not be suitable for similarity matrix. The reason is that the original similarity between different objects has already been normalized, so a further normalization from the similarity matrix to transition matrix may weaken or damage inherent meanings of the original similarity. For example, if a tweet u_i is aligned with one and only one document v_j with relatively low similarity weight, the transition probability w_{ij}^{uv} will be increased to 1 after normalization. Similarly, some higher similarity weights may be normalized to small transition probabilities.

By extending and adapting Co-HITS, we develop Tri-HITS to handle heterogeneous networks with three types of objects: users, tweets and web documents. Given the similarity matrices W^{dt} (between documents and tweets) and W^{tu} (between tweets and users), and initial ranking scores of $s^0(d)$, $s^0(t)$ and $s^0(u)$, we aim to refine the initial ranking scores and obtain the final ranking scores $s(d)$, $s(t)$ and $s(u)$. Starting from document $s(d)$, the update process considers both the initial score $s^0(d)$ and the propagation from connected tweets $s(t)$, which can be expressed as:

$$\begin{aligned} \hat{s}(d_i) &= \sum_{j \in T} w_{ji}^{td} s(t_j), \\ s(d_i) &= (1-\lambda_{td}) s^0(d_i) + \lambda_{td} \frac{\hat{s}(d_i)}{\sum_i \hat{s}(d_i)}, \end{aligned} \tag{10}$$

where W^{td} is the transpose of W^{dt}, and $\lambda_{td} \in [0,1]$ is the parameter to balance between initial and propagated ranking scores. Tri-HITS normalizes the propagated ranking scores $\hat{s}(d_i)$, while Co-HITS propagates normalized ranking scores by using the transition matrix instead of the original similarity matrix, potentially weakening or damaging the inherent meanings of the original similarity. Similarly, we define the propagation from tweets to users as:

$$\begin{aligned} \hat{s}(u_k) &= \sum_{j \in T} w_{jk}^{tu} s(t_j), \\ s(u_k) &= (1-\lambda_{tu}) s^0(u_k) + \lambda_{tu} \frac{\hat{s}(u_k)}{\sum_k \hat{s}(u_k)}, \end{aligned} \tag{11}$$

Each tweet $s(t_j)$ may be influenced by the propagation from both documents and users:

$$\begin{aligned} \hat{s}_d(t_j) &= \sum_{i \in D} w_{ij}^{dt} s(d_i), \\ \hat{s}_u(t_j) &= \sum_{k \in U} w_{kj}^{ut} s(u_k), \\ s(t_j) &= (1-\lambda_{dt}-\lambda_{ut}) s^0(t_j) \\ &\quad + \lambda_{dt} \frac{\hat{s}_d(t_j)}{\sum_j \hat{s}_d(t_j)} + \lambda_{ut} \frac{\hat{s}_u(t_j)}{\sum_j \hat{s}_u(t_j)}. \end{aligned} \tag{12}$$

where W^{ut} is the transpose of W^{tu}, λ_{dt} and λ_{ut} are parameters to balance between initial and propagated ranking scores. The λ variables define the networks being considered: (i) when λ_{dt} is

Input: A set of tweets (T), and users (U) on a given topic.
Output: Ranking scores (S_t) for T.

1: Use rule-based method to filter out noisy tweets (remaining $\hat{T}$ posted by users $\hat{U}$);
2: Retrieve relevant web documents D for $\hat{T}$;
3: Use TextRank and Bayesian Ranking to compute initial ranking scores S_t^0 for $\hat{T}$, S_d^0 for D and initial credibility scores S_u^0 for $\hat{U}$;
4: Construct heterogeneous networks across $\hat{T}$, $\hat{U}$ and D;
5: $k \leftarrow 0$, $diff \leftarrow 10e6$;
6: **while** $k <$ MaxIteration and $diff >$ MinThreshold **do**
7: Use Eq. (12) to compute S_t^{k+1};
8: Use Eq. (11) to compute S_u^{k+1};
9: Use Eq. (10) to compute S_d^{k+1};
10: Normalize S_t^{k+1},S_d^{k+1}, and S_u^{k+1};
11: $diff \leftarrow \sum(|S_t^{k+1} - S_t^k|)$;
12: $k \leftarrow k + 1$
13: **end while**

Algorithm 1: Tri-HITS: Tweet ranking using heterogeneous networks

set to 0, only Tweet-User networks are considered (Method 3 in Table 1); (ii) when λ_{ut} is set to 0, only Web-Tweet networks are considered (Method 4); (iii) when both λ_{dt} and λ_{ut} are different from 0, the entire heterogeneous Web-Tweet-User network is considered (Method 5). For methods relying on bipartite graphs, we define as *one-step propagation* when the propagation is performed in a single direction, while we call it *two-step propagation* when it is performed in both directions. The selection of one-step propagation and two-step propagation is controlled by λ parameters.

Model Convergence Proof: From Equation (10), and assuming $\lambda_{td} > 0$ (the ranking scores $s(d)$ for web documents would not change if $\lambda_{td} = 0$), we get:

$$\bar{s}(d_i) = \frac{1}{\lambda_{td}}[s(d_i) - (1 - \lambda_{td})s^0(d_i)] = \frac{\hat{s}(d_i)}{\sum_i \hat{s}(d_i)}. \quad (13)$$

$\bar{s}(d)$, the normalized score of $\hat{s}(d)$, is similar to the normalized authority or hub scores defined in HITS (Kleinberg, 1999), the difference being only the function to select vector norms. Kleinberg proved that $\bar{s}(d_i)$ converges as the iterative procedure continues, from which the convergence of the ranking scores $s(d)$ for web documents is guaranteed. The same assumption proves the convergence of ranking scores for tweets and users.

Algorithm 1 summarizes Tri-HITS.

5 Experiments

Next, we present the experiment settings and analyze the methods shown in Table 1.

Methods	Descriptions	Hypotheses
1. Baseline	TextRank based on tweet-tweet networks.	
2. 1+Filtering	Baseline with filtering included.	
3. 2+Tweet-User*	Propagation on explicit and implicit Tweet-User networks.	1 and 2
4. 2+Web-Tweet	Propagation on Web-Tweet networks.	3
5. 3+4 Web-Tweet-User*	Propagation on Web-Tweet-User networks.	all

Table 1: Description of methods (method with * make use of the Bayesian Approach to initialize user credibility scores.

Grade	5	4	3	2	1
Hour 1	65	48	93	119	847
Hour 2	135	159	255	164	458
Hour 3	129	102	162	123	602

Table 2: Tweet distribution by grade

5.1 Data

We use tweets on the Hurricane Irene from August 26 to September 2, 2011 for our experiments. Using the query terms *hurricane* or *irene* to monitor tweets, we collected 176,014 tweets posted by 139,136 users within that timeframe. For evaluation purposes, we segment the tweets into 153 hours with an average of 1,150 tweets in each hour.

We randomly chose tweets from three hours to be manually annotated as our reference. This subset contains 3,460 tweets posted on different days: August 27, 2011, August 28, 2011 and September 1, 2011. Following the annotation guidelines defined by (Huang et al., 2011), two annotators parallelly assigned each tweet a grade in a 5-star likert scale. Tweets with grade 5 are the most informative, while tweets with label 1 are the least informative. When the label difference between annotators was 1, the lower grade was selected. When the label difference was greater than 1, those tweets were re-annotated until the label difference did not exceed 1. Table 2 shows the distributions of all grades for each of the three hours of tweets.

5.2 Evaluation Metric

To evaluate tweet ranking, we rely on three-fold cross validation using *nDCG* as a measure (Jarvelin and Kekalainen, 2002), which considers both the informativeness, and the position of a tweet:

$$nDCG(\Phi, k) = \frac{1}{|\Phi|} \sum_{i=1}^{|\Phi|} \frac{DCG_{ik}}{IDCG_{ik}},$$

$$DCG_{ik} = \sum_{j=1}^{k} \frac{2^{rel_{ij}} - 1}{\log(1+j)},$$

where Φ is the set of documents in the test set, each document corresponding to an hour of tweets in our case, rel_{ij} is the human-annotated label for the tweet j in the document i, and $IDCG_{ik}$ is the DCG score for the ideal ranking. The average *nDCG* score for the top k tweets is: $Avg@k = \sum_{i=1}^{k} nDCG(\Phi, i)/k$. To favor diversity of top ranked tweets, redundant tweets are penalized to lower down the final score.

5.3 Effect of Parameters

We study the impact of different parameters on the training set. We present the most representative figures to show the effect, due to the lack of space. For TextRank, we explore δ_{tt} values from 0 to 1. For the enhanced approaches, we firstly perform one-step propagation of ranking scores from web documents to tweets by considering all pairs of δ_{td} and λ_{dt} from 0 to 1 with a step of 0.1. For each δ_{td}, the corresponding λ_{dt} and the best average nDCG scores for top 10 and 100 tweets are shown in Figure 3(a). We notice that when both initial tweet ranking scores and propagated ranking scores from web documents are considered (i.e., δ_{td} is set from 0 to 0.9 and $\lambda_{dt} > 0$), the ranking quality

outperforms that by simply considering initial ranking scores of tweets (i.e. $\delta_{td} = 1$). Secondly, for the ranking performance of double-step ranking scores propagation, we choose to set $\delta_{td} = 0.1$, $\lambda_{dt} = 0.4$ and test λ_{td} from 0 to 1. Figure 3(b) shows an encouraging improvement in the ranking quality, and more stable over the baseline and one-step propagation. This suggests that two-step propagation provides mutual improvement in the ranking quality. The reason is that the ranking of web documents may also be refined using tweet and user evidence thanks to the large volume and synchrony of tweeting (Zanzotto et al., 2011). Here, $\lambda_{td} = 0.2$ yields the best performance. The aforementioned process is followed for Tweet-User networks, finding the best performance for $\delta_{tu} = 0.1$, $\lambda_{ut} = 0.2$, and $\lambda_{tu} = 0.6$.

When validating on the test set, Method 4 based on Web-Tweet networks outperforms Method 3 relying on Tweet-User networks. Therefore, for Web-Tweet-User networks, we keep the above values, and explore λ_{ut} values from 0 to 0.6 (e.g., $1 - \lambda_{dt}$). Figure 3(c) shows that integrating web documents, tweets and users, the ranking quality improves over both Web-Tweet networks and Tweet-User networks.

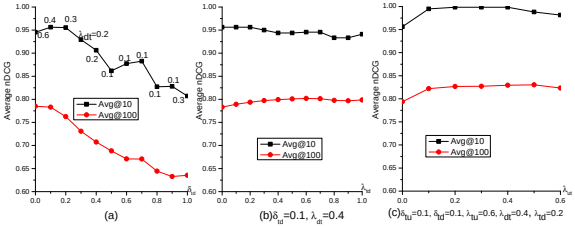


Figure 3: Effect of parameters: (a) δ_{td} and λ_{dt} for Web-Tweet networks, (b) λ_{td} for Web-Tweet networks, (c) λ_{dt} for Web-Tweet-User networks.

5.4 Performance and Analysis

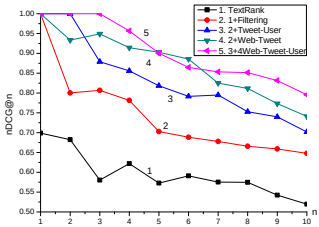


Figure 4: Performance comparison of ranking methods

Figure 4 shows the performance of ranking methods. The performance gain from Method 1 to Method 2 shows the need of filtering short and informal tweets. In this case, filtering reduced from 3,460 to 1,765 tweets ($\sim$ 49% reduction). Table 3 shows the distribution of labels for filtered tweets: a great majority of 91.75% had been annotated as 1, while only 0.11% had been annotated as 5.

Methods 3, 4 and 5, which integrate heterogeneous networks after filtering, outperform the baseline TextRank. When tweets are aligned with web documents (Method 4), the ranking quality improves

Grade	5	4	3	2	1
Percentage	0.11%	0.17%	3.13%	4.84%	91.75%

Table 3: Grade distributions for filtered tweets.

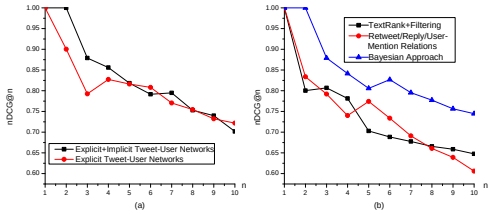


Figure 5: (a) Explicit vs Inferred Implicit Tweet-User Relations to Construct Tweet-User Networks; (b) TextRank vs One-step Propagation on Explicit Tweet-User Networks Using Bayesian Approach and Retweet/Reply/User Mention Relations.

significantly, proving that web documents can help infer informative tweets adding support from a formal genre. The fact that tweets with low initial ranking scores are aligned with web documents helps improve their ranking positions (Hypothesis 3). For example, the ranking of the tweet "*Hurricane Irene: City by City Forecasts http://t.co/x1t122A*" is improved compared to TextRank, helped by the fact that 10 retrieved web documents are about this topic.

Integrating users (Method 5) further improves performance. This indicates that Web-Tweet and Tweet-User networks may complement each other in improving ranking. For example, the tweet "*A social-media guide to dealing with Hurricane Irene http://t.co/0XBEnEJ*" is not top-ranked when only using Web-Tweet networks, since none of the retrieved web documents is related to it. However, similar tweets appear with high frequency in the tweet set. Hence, inferring implicit tweet-user relations and propagating information through the tweet-user network also improves the ranking.

Figure 5(a) shows that inferring implicit tweet-user relationships outperforms the only use of explicit tweet-user relations, especially for top positions. Looking into lower positions, we find that the redundancy removal performs better for the only use of explicit relations. However, both approaches can still perform similarly in positions $5 \sim 10$. This corroborates the synchronous behavior of users as an indicator of informative contents (Hypothesis 2). Since it is likely that a large set of users only tweet once within a short timeframe, limiting to explicit tweet-user relations results in sparse links, and ranking quality cannot be bootstrapped. Interestingly, inferring implicit tweet-user relations can capture synchronous behavior of users, which indicates subjects that users are concerned about.

Figure 5(b) shows that initializing user credibility scores with the Bayesian approach and performing one-step ranking score propagation from users to tweets based on the explicit tweet-user networks also outperforms TextRank. This corroborates our hypothesis that credible users are more likely to post informative tweets (Hypothesis 1). In addition, using only retweets, replies, and user mentions to compute initial user ranking scores, the performance does not improve over TextRank. The reason is that for an event of general interest like the Hurricane Irene, users from different communities rarely interact with each other.

Finally, Figure 6 shows that Tri-HITS significantly outperforms Co-HITS over bipartite graphs, with the only exception of position $n = 2$ for the Web-Tweet network. This corroborates that normalizing the similarity matrix weakens semantic relations between different objects, and that

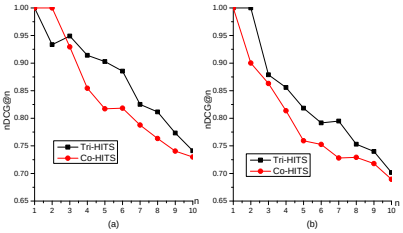


Figure 6: Co-HITS vs Tri-HITS on (a) Web-Tweet Networks, (b) Tweet-User Networks

capturing inherent meanings of cross-genre linkages is crucial for information propagation.

5.5 Remaining Error Analysis

Tri-HITS shows encouraging improvements in ranking quality with respect to a state-of-the-art model like TextRank. However, there are still some issues to be addressed for further improvements.

(i) Topically-relevant tweet identification. We tracked tweets containing the keywords "Hurricane" and "Irene". Using such a query to follow tweets might also return tweets that are not related to the event being followed. This may occur either because the terms are ambiguous, or because of spam being injected into trending conversations to make it visible. For example, the tweet "*Hurricane Kitty: http://t.co/cdIexE3*" is an advertisement, which is not topically related to Irene.

(ii) Non-informative tweet identification. Our rule-based filtering component achieves high precision (96.59%) on the identification of non-informative tweets, while there are still a number of false positives with a 70.7% recall. Performing deeper linguistic analysis, such as exploring subjectivity, might help clean up the tweet set by identifying additional non-informative tweets. For example, an analysis of writing styles would help identify the tweet "*Hurricane names hurricane names http://t.co/iisc7UY ;)*" as informal because it contains repeated phrases. And the tweet "*My favorite parts of Hurricane coverage is when the weathercasters stand in those 100 MPH winds right on the beach. Good stuff.*" is clearly subjective commentary that may entertain but will not meet the general interest of people involved with or tracking the event.

(iii) Deep semantic analysis of the content. Users may rely on distinct terms to refer to the same concept. More extensive semantic analyses of text could help identify those terms, possibly enhancing the propagation process. For example, information extraction tools can be used to extract entities and events, and their coreferential relations, such as "NYC" and "New York City", or "MTA closed" and "subway shutting down". Likewise, existing dictionaries such as WordNet (Miller, 1995) can be utilized to mine synonym/hypernym/hyponym relations, and Brown clusters (Brown et al., 1992) can be explored to mine other types of relations.

6 Related Work

We discuss relevant research on tweet ranking, information credibility for tweets, and the use of graphical models.

Previous research on tweet ranking has relied on the analysis of content (Sankaranarayanan et al., 2009), user credibility (Golder et al., 2009; Weng et al., 2010; Yamaguchi et al., 2010; Hannon et al., 2010; Uysal and Croft, 2011) and URL availability, or combinations of them (Duan et al., 2010;

Huang et al., 2011). In addition, (Huang et al., 2011) also exploited content similarity to propagate evidence within the tweet genre. Most work has been based on supervised learning models such as RankSVM, Naive-Bayes classifier, and Linear Regression. (Inouye and Kalita, 2011) compared various unsupervised methods to rank tweets for summarization purposes, but only used lexical-level content analysis features.

In analyzing the information credibility of tweets, (Castillo et al., 2011) relied on various levels of features (i.e., message-based, user-based, topic-based and propagation-based features) and supervised learning models for information credibility assessment in Twitter, which (Gupta et al., 2012) extended by capturing relations among events, tweets, and users. (Wang et al., 2011, 2012) proposed a Bayesian interpretation to assess tweet credibility. However, it remains as a preliminary approach due to the linear assumption made in the iterative algorithm of the basic fact-finding scheme. Intensive research has also been conducted on information credibility analysis (cf. (Gupta and Han, 2011)).

Graphical models have been effectively used in document summarization (Mihalcea, 2004; Sornil and Greeu, 2006; Sharifi et al., 2010) demonstrating their power of propagating information across linked instances. However, most of these models, such as TextRank (Mihalcea and Tarau, 2004), as originally developed apply only to homogeneous networks. In contrast to existing research, we introduce Tri-HITS, a novel method that incorporates evidence from multiple genres, by exploiting semantically-related links to external web documents and inferring the implicit tweet-user relations. Following a different method for linking tweets and web documents, (Dong et al., 2010) used outgoing links from tweets to improve recency ranking for a search engine.

7 Conclusions and Future Work

We have introduced Tri-HITS, a novel propagation model that makes use of heterogeneous networks composed of tweets, users, and web documents to rank tweets. To the best of our knowledge, this is the first approach to integrating tweets with formal genres that improves tweet ranking quality. Using propagation models to define ranking scores, we have shown that information from the formal genre of web documents can help improve the ranking quality. By introducing this new propagation model, studying the integration of different genres, presenting a way of inferring implicit tweet-user relations, and exploring the impact of parameters, this work sheds light on the challenging task of ranking tweets that are written informally by a diverse community of users.

Our next step is to develop metrics to predict ranking confidence so that we can remove low-confidence results and outliers from the evidence propagation. In addition, ranking tweets (and later, news) by their informativeness within a given time frame, will help in identifying elements of information for inclusion in a summary. More ambitiously, in future work, we plan to generate automatic summaries from the information jointly provided by tweets and web documents.

Acknowledgments

This work was supported by the U.S. Army Research Laboratory under Cooperative Agreement No. W911NF-09-2-0053, the U.S. NSF Grants IIS-0953149, IIS-1144111, IIS-0905215, CNS-0931975 and the U.S. DARPA BOLT program, the U.S. Air Force Office of Scientific Research MURI award FA9550-08-1-0265. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation here on.

References

Brown, P. F., deSouza, P. V., Mercer, R. L., Pietra, V. J. D., and Lai, J. C. (1992). Class-based n-gram models of natural language. *Computational Linguistics*, 18:467–479.

Carterette, B. and Chandar, P. (2009). Probabilistic models of ranking novel documents for faceted topic retrieval. In *Proceedings of the 18th ACM conference on Information and knowledge management*, CIKM '09, pages 1287–1296, New York, NY, USA. ACM.

Castillo, C., Mendoza, M., and Poblete, B. (2011). Information credibility on twitter. In *Proceedings of the 20th international conference on World wide web*, pages 675–684. ACM.

Couzin, I. (2007). Collective minds. *Nature*, 445.

Deng, H., Han, J., Zhao, B., Yu, Y., and Lin, C. (2011). Probabilistic topic models with biased propagation on heterogeneous information networks. In *Proc. ACM SIGKDD2011*, pages 1271–1279. ACM.

Deng, H., Lyu, M. R., and King, I. (2009). A generalized co-hits algorithm and its application to bipartite graphs. In *Proc. ACM SIGKDD2009*.

Diakopoulos, N., De Choudhury, M., and Naaman, M. (2012). Finding and assessing social media information sources in the context of journalism. In *Proc. Conference on Human Factors in Computing Systems (CHI)*.

Dong, A., Zhang, R., Kolari, P., Bai, J., Diaz, F., Chang, Y., Zheng, Z., and Zha, H. (2010). Time is of the essence: improving recency ranking using twitter data. In *Proceedings of the 19th international conference on World wide web*, WWW '10, pages 331–340, New York, NY, USA. ACM.

Duan, Y., Jiang, L., Qin, T., Zhou, M., and Shum, H.-Y. (2010). An empirical study on learning to rank of tweets. In *Proceedings of the 23rd International Conference on Computational Linguistics*, COLING '10, pages 295–303, Stroudsburg, PA, USA. Association for Computational Linguistics.

Golder, S. A., Marwick, A., and Yardi, S. (2009). A structural approach to contact recommendations in online social networks. In *Proc. SIGIR2009 Workshop on Search in Social Media*.

Gupta, M. and Han, J. (2011). Heterogeneous network-based trust analysis: a survey. *SIGKDD Explor. Newsl.*, 13(1):54–71.

Gupta, M., Zhao, P., and Han, J. (2012). Evaluating event credibility on twitter. In *SDM*, pages 153–164.

Hannon, J., Bennett, M., and Smyth, B. (2010). Recommending twitter users to follow using content and collaborative filtering approaches. In *Proceedings of the fourth ACM conference on Recommender Systems*.

Huang, M., Yang, Y., and Zhu, X. (2011). Quality-biased ranking of short texts in microblogging services. In *Proceedings of 5th International Joint Conference on Natural Language Processing*, pages 373–382, Chiang Mai, Thailand. Asian Federation of Natural Language Processing.

Inouye, D. and Kalita, J. K. (2011). Comparing twitter summarization algorithms. In *IEEE SocialCom 2011*.

Jarvelin, K. and Kekalainen, J. (2002). Cumulated Gain-based Evaluation of IR Techniques. *ACM Transactions on Information Systems*, 20.

Kleinberg, J. M. (1999). Authoritative sources in a hyperlinked environment. *J. ACM*, 46(5):604–632.

McDonald, R. (2007). A study of global inference algorithms in multi-document summarization. In *Proceedings of the 29th European conference on IR research*, ECIR'07, pages 557–564, Berlin, Heidelberg. Springer-Verlag.

Mihalcea, R. (2004). Graph-based ranking algorithms for sentence extraction, applied to text summarization. In *Proc. ACL2004*.

Mihalcea, R. and Tarau, P. (2004). Textrank: Bringing order into texts. In *Proceedings of EMNLP*, volume 4. Barcelona: ACL.

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

Page, L., Brin, S., Motwani, R., and Winograd, T. (1998). The pagerank citation ranking: Bringing order to the web. In *Proc. the 7th International World Wide Web Conference*.

Saavedra, S., Hagerty, K., and Uzzi, B. (2011). Synchronicity, Instant Messaging and Performance among Financial Traders. *PNAS*.

Sankaranarayanan, J., Samet, H., Teitler, B. E., Lieberman, M. D., and Sperling, J. (2009). Twitterstand: News in tweets. In *Proceedings of the 17th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*.

Sharifi, B., Hutton, M.-A., and Kalita, J. K. (2010). Experiments in microblog summarization. In *IEEE Second International Conference on Social Computing (SocialCom)*.

Sornil, O. and Greeu, K. (2006). An automatic text summarization approach using content-based and graph-based characteristics. In *IEEE Conference on Cybernetics and Intelligent Systems*.

Uysal, I. and Croft, W. B. (2011). User oriented tweet ranking: A filtering approach to microblogs. In *Proc. CIKM2011 (Poster)*.

Wang, D., Abdelzaher, T., Ahmadi, H., Pasternack, J., Roth, D., Gupta, M., Han, J., Fatemieh, O., Le, H., and Aggrawal, C. (2011). On bayesian interpretation of fact-finding in information networks. In *Proc 14th International Conference on Information Fusion (Fusion '11)*.

Wang, D., Le, H., Kaplan, L., and Abdelzaher, T. (2012). On truth discovery in social sensing: A maximum likelihood estimation approach. In *Proc. 11th ACM/IEEE Conference on Information Processing in Sensor Networks (IPSN)*.

Weng, J., Lim, E.-P., Jiang, J., and He, Q. (2010). Twitterrank: Finding topic-sensitive influential twitterers. In *Proc. WSDM2010*.

Yamaguchi, Y., Takahashi, T., Amagasa, T., and Kitagawa, H. (2010). Turank: Twitter user ranking based on user-tweet graph analysis. In *Proc. WISE2010*.

Zanzotto, F. M., Pennacchiotti, M., and Tsioutsiouliklis, K. (2011). Linguistic redundancy in twitter. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '11, pages 659–669, Stroudsburg, PA, USA. Association for Computational Linguistics.

Zubiaga, A., Spina, D., Amigó, E., and Gonzalo, J. (2012). Towards real-time summarization of scheduled events from twitter streams. In *Proceedings of the 23rd ACM conference on Hypertext and social media*, pages 319–320. ACM.

Improved Combinatory Categorial Grammar Induction with Boundary Words and Bayesian Inference

Yun HUANG[1,2] *Min ZHANG*[2] *Chew Lim TAN*[1]

[1]Department of Computer Science
National University of Singapore
13 Computing Drive, Singapore
{huangyun,tancl}@comp.nus.edu.sg

[2]Human Language Department
Institute for Infocomm Research
1 Fusionopolis Way, Singapore
mzhang@i2r.a-star.edu.sg

ABSTRACT

Combinatory Categorial Grammar (CCG) is an expressive grammar formalism which is able to capture long-range dependencies. However, building large and wide-coverage treebanks for CCG is expensive and time-consuming. In this paper, we focus on the problem of unsupervised CCG induction from plain texts. Based on the baseline model in (Bisk and Hockenmaier, 2012), we propose following two improvements: (1) we utilize boundary part-of-speech (POS) tags to capture lexical information; (2) we perform nonparametric Bayesian inference based on the Pitman-Yor process to learn compact grammars. Experiments on English Penn treebank demonstrate the effectiveness of our boundary model and Bayesian learning.

TITLE AND ABSTRACT IN ANOTHER LANGUAGE (CHINESE)

基于边界词和贝叶斯模型改进的组合范畴语法推导

组合范畴语法(CCG)是一种具有丰富表达能力的语法形式，它可以捕获长距离的依赖关系。但是，构建大规模、覆盖面广的组合范畴语法语料库既昂贵又耗时。在本文中，我们研究如何从普通文本中无监督地推导出组合范畴语法。基于现有的工作(Bisk and Hockenmaier, 2012)，我们提出以下两个改进：(1) 我们使用边界词的词性标记把词汇化信息引入模型中；(2) 我们使用基于Pitman-Yor过程的非参数贝叶斯模型来学习简洁的文法。在英语宾州树库上的实验结果显示了我们提出的边界词模型和贝叶斯模型的有效性。

KEYWORDS: Grammar Induction, Combinatory Categorial Grammar, Boundary Words, Bayesian Model.

KEYWORDS IN CHINESE: 语法推导, 组合范畴语法, 边界词, 贝叶斯模型.

Proceedings of COLING 2012: Technical Papers, pages 1257–1274,
COLING 2012, Mumbai, December 2012.

1 Introduction

Unsupervised grammar induction has attracted research interests for a long time. The induced grammars can be used to construct large treebanks (van Zaanen, 2000), study language acquisition (Jones et al., 2010), etc. In recent years, numerous approaches have been introduced to automatically induce hierarchical structures from plain strings. Some approaches focus on the constituency grammar induction: the constituent-context model (Klein and Manning, 2002), the data-oriented parsing (Bod, 2006), the common cover link model (Seginer, 2007), and the tree-substitution grammars (TSG) (Cohn et al., 2009). The other mainstream is the dependency grammar induction: the dependency model with valence (DMV) (Klein and Manning, 2004; Headden III et al., 2009; Cohen and Smith, 2009), TSG model for dependency (Blunsom and Cohn, 2010), etc.

Among these grammar formalisms, the Combinatorial Categorial Grammar (CCG) is a lexicalized, mildly-context-sensitive model (Steedman, 2000). In the formal grammar theory, CCGs are known to be weekly equivalent to Linear Indexed Grammars, Tree-adjoining Grammars, and Head Grammars (Vijay-Shanker and Weir, 1994). The CCG formalism provides a transparent interface between syntax and semantics, such that the underlying semantics could be naturally defined over syntactical derivations, including the long-range dependencies, the coordination structure, and the extraction phenomenon (Bos et al., 2004; Zettlemoyer and Collins, 2007). As a mildly context-sensitive grammar, CCG can be efficiently parsed in polynomial time, which makes them practical in real parsing tasks[1]. The wide-coverage combinatorial categorial grammars have been used in many NLP tasks, such as the lexical acquisition (Blunsom and Baldwin, 2006), the parsing tasks (Hockenmaier and Steedman, 2002; Clark and Curran, 2003), and the statistical machine translation (Hassan et al., 2007; Zhang and Clark, 2011). These supervised CCG models highly depend on the annotated training corpus, e.g. the CCGbank (Hockenmaier and Steedman, 2007). However, building large and wide-coverage treebanks for CCG is expensive and time-consuming. Therefore, how to induce CCG lexicons and grammars from unlabeled sentences has great values.

Some unsupervised CCG induction models have been proposed (Osborne and Briscoe, 1997; Watkinson and Manandhar, 1999; Ponvert, 2007; Boonkwan and Steedman, 2011; Bisk and Hockenmaier, 2012). Most of these approaches define probabilistic models over CCG rules and use the Expectation-Maximization (EM) algorithm to estimate parameters. The generative process generates grammar rules independently given their parents, without regard to the lexical information. However, the constituents and contexts have been proven useful for grammar induction (Klein and Manning, 2002; Headden III et al., 2009). Another issue of the EM-based models is that the EM algorithm tends to overfit the training data, which requires carefully smoothing (Headden III et al., 2009).

In this paper, we propose to incorporate the lexical information in the unsupervised CCG induction in order to capture more complex language aspects. Specifically, an additional *boundary model*, which defines probability distributions over the boundary part-of-speech tags, is introduced during the probability calculation for parse trees. Furthermore, we present the nonparametric Bayesian inference to alleviate the overfitting problem of EM. The Pitman-Yor process (Pitman and Yor, 1997) is used to encourage rule reuse, resulting in compact grammars. Although the boundary words and Bayesian inference have been used in other grammar

[1] Given grammar G, the parsing complexity of CCG is $O(n^3|G|)$ for the sentences with length n. Clark and Curran (2007) report their supervised parser could parse $20-30$ sentences/second using the treebank grammar.

induction models, so far as we know they are used in CCG induction for the first time. Experimental results show that both the boundary model and the Bayesian inference outperform the baseline CCG induction system significantly.

This paper is structured as follows. First we give a brief overview of the combinatorial categorial grammars in Section 2. Then we present the grammar generation step in Section 3. In Section 4, we describe the baseline model and propose the boundary model and the nonparametric Bayesian learning framework. Experimental results and related work are shown in Section 5 and Section 6 respectively. We conclude our work in the final section.

2 Combinatory Categorial Grammar

Combinatory Categorial Grammar (CCG) is a linguistically expressive lexicalized grammar formalism (Steedman, 2000). In CCG, words and nonterminals are associated with rich syntactic categories which capture the basic word order and subcategorization. Specifically, the categories in CCG are defined recursively: (1) There are some atomic categories, e.g. `S`, `N`; (2) Complex categories take the form `X/Y` or `X\Y`, representing the syntactical function that takes the input category `Y` and outputs the result category `X`. The forward slash (`/`) and the backward slash (`\`) indicate the input category `Y` follows or precedes the complex category respectively. Note that `X` and `Y` themselves may be complex categories too. Parentheses can be used to specify the order of function applications if needed. By default, the slashes are left-associated, e.g. "`X\Y/Z`" is the shorthand of "`(X\Y)/Z`". If the order of categories is not important in some cases, we use the symbol "`|`" to represent either the forward slash or the backward slash. The following examples show some common categories in English grammars: `S` for sentences, `N` for nouns and noun phrases[2], `(S\N)/N` for transitive verbs, `N/N` for determiners and adjectives, etc.

The *derivation* of CCG is the sequence of CCG rule applications. There are a few kinds of rule templates defined in CCG. The simplest rules are the forward application (>) and the backward application (<), where the complex category consumes the whole input category:

`X/Y`	`Y`	$\Rightarrow$	`X`	(>)
`Y`	`X\Y`	$\Rightarrow$	`X`	(<)

The input category could be complex category as well, forming the composition rules:

`X/Y`	`Y|Z`	$\Rightarrow$	`X|Z`	(>B^1)
`Y|Z`	`X\Y`	$\Rightarrow$	`X|Z`	(<B^1)

Note that the above composition rules could reduce the categories `X/Y` and `Y\Z` to the category `X\Z`, using the so-called *cross composition rule*. These rules give CCG the ability to deal with the crossed dependencies in some languages such as Dutch or German[3].

Higher order composition rules can be defined similarly:

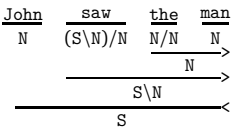
`X/Y`	$Y\|Z_1\|\ldots\|Z_n$	$\Rightarrow$	$X\|Z_1\|\ldots\|Z_n$	(>B^n)
$Y\|Z_1\|\ldots\|Z_n$	`X\Y`	$\Rightarrow$	$X\|Z_1\|\ldots\|Z_n$	(<B^n)

In a sense, the application rules (> and <) can be regarded as the zero-order case of composition rules (>B^0 and <B^0).

[2]In formal English grammars, `NP` is often used to represent noun phrases(Hockenmaier and Steedman, 2007). Following (Bisk and Hockenmaier, 2012), we do not distinguish noun phrase from nouns in this paper for efficiency. This simple treatment causes some problems, e.g. the determiners would be treated as adjuncts and then regarded optional, but actually they are needed for singular count nouns. We leave this problem to future work.

[3]See the example 71 (a German sentence) in (Steedman and Baldridge, 2011)

The following examples show the CCG derivations of a declarative sentence:



(1)

In this example, the lexical category (S\N)/N for the transitive verb "saw" restricts that the verb must first consume an object noun (N) on the right to become the intransitive verb category S\N, then take another noun (N) on the left as the subject to form the sentence S. We can see that the lexicons encode rich lexical information as well as the syntactic restrictions.

For coordination, only the same categories can be conjuncted to yield a single category of the same type in CCG. In detail, the CCG includes a ternary conjunction rule (&).

X conj X ⇒ X (&)

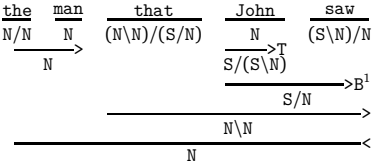
For the parsing algorithms (e.g. the bottom-up CKY algorithm) that require binary rules, we often use the binarized conjunction rules (>& and <&):

X X[conj] ⇒ X (>&)
conj X ⇒ X[conj] (<&)

The type-raising rules are also included in CCG, which turn arguments into functions over functions-over-such-arguments:

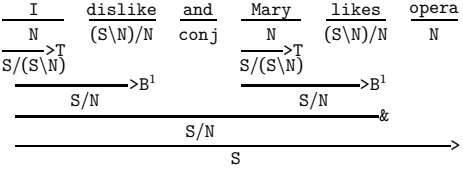
X ⇒ T/(T\X) (>T)
X ⇒ T\(T/X) (<T)

These rules are needed to form some unusual constituents, such as the constituent "John saw" in the example (2). In this example, there is no argument on the right to transitive verb "saw" due to the clause structure, so the noun "John" has to be type-raised.



(2)

Another example of type-raising is the uncommon coordination case (example (3)), in which the two uncommon subject-verb categories S/N are conjuncted.



(3)

In example (1) and (2), it should be emphasised that the same words have the same lexical categories, although the sentence structures are totally different. This elegant treatment for coordination and extraction structures in CCG allows easy recovery of the long-range dependencies and semantics.

Following (Bisk and Hockenmaier, 2012), we group the complex categories into the following two types according to their linguistic functions.

- **Modifier category**. Categories in the form of X|X are modifier categories. In other words, modifier categories take one category as input and output the same category as result. Some modifier category examples are the noun modifier "N/N", the sentence modifier "S\S", and the more complex category "(S\S)/(S\S)".
- **Functor category**. In contrast, the functor category takes one category as input but output a different category as result, i.e. in the form of X|Y, where X and Y are different categories. In the example (2), the follows are all functor categories: the transitive verb "(S\N)/N", the uncommon constituent "S/N", the type-raised category "S/(S\N)", and the relative pronoun "(N\N)/(S/N)".

The modifier categories and the functor categories play different linguistic roles. Using the dependency terminology, the modifier category X|X acts as the dependent and modifies its head X, while the functor category X|Y acts as the head of its dependent Y. We will revisit this issue in the CCG evaluation (section 5.1).

In this paper, we focus on the problem of unsupervised CCG induction, the task to infer meaningful grammars and tree structures from plain texts. We will first describe the grammar generation step in Section 3, and propose the boundary model and the Bayesian learning method in Section 4.

3 Grammar Generation

The combinatory categorial grammars explicitly encode the head-modifier and head-argument dependencies into rich syntactic categories. The first step of our CCG induction method is the lexicon generation step. Bisk and Hockenmaier (2012) propose a simple iterative lexicon generation algorithm from the golden part-of-speech tags. Due to the simplicity and effectiveness of this method, we adopt it to generate lexicons in our method. We rephrase their algorithm with minor modifications in this section.

Only two atomic categories, N (nouns or noun phrases) and S (sentences) are allowed in the grammar. Conjunction words (usually with part-of-speech tag CC in the Penn Treebank) are expanded from a special conjunction category conj. All trees are generated from a special start symbol TOP. Following (Bisk and Hockenmaier, 2012), we assume all strings are either nouns or sentences, i.e. they are generated from one of the following two unary rules:

$$\texttt{TOP} \rightarrow \texttt{N} \qquad\qquad \texttt{TOP} \rightarrow \texttt{S}$$

In addition, we restrict that: (1) strings containing at least one verb must be sentences, i.e. parsed with the TOP → S rule; and (2) strings without any verb must be parsed with the TOP → N rule[4]. Note that the above assumptions are not always true for real sentences. For instance, there exist some sentence fragments and exclamatory sentences in the English treebank. In this paper, we just follow previous work and have no special consideration on these cases for simplicity.

[4]Only the first restriction is used in (Bisk and Hockenmaier, 2012).

The initial CCG lexicon $\mathscr{L}^{(0)}$ is created manually. We associate the noun POS tags with the atomic category N, the verb POS tags with the atomic category S, and the conjunction POS tag with the special category conj. For the POS tag set of the English Penn treebank (Marcus et al., 1993), the initial CCG lexicons are shown as follows:

$$\begin{aligned} \texttt{N} &: \{\texttt{DT}, \texttt{NN}, \texttt{NNS}, \texttt{NNP}, \texttt{NNPS}, \texttt{PRP}\} \\ \texttt{S} &: \{\texttt{MD}, \texttt{VB}, \texttt{VBD}, \texttt{VBG}, \texttt{VBN}, \texttt{VBP}, \texttt{VBZ}\} \\ \texttt{conj} &: \{\texttt{CC}\} \end{aligned}$$

Note that the tag NNPS (representing plural proper nouns) and the tag VBP (representing verbs of non-3rd person singular and in present tense) are missed in (Bisk and Hockenmaier, 2012), but these two tags are found in the Penn treebank tag set.

The lexicon for atomic categories remains fixed after the initial lexicon $\mathscr{L}^{(0)}$ has been created. However, the POS tags may acquire more syntactic categories in the lexicon generation stage. In each induction step, we first assign each POS tag with the categories induced in the previous step, then create new candidates for adjacent POS tags of the training sentences.

- **Modifier category**. Assuming the i^{th} POS tag in some given sentence has been associated with category X, we create new modifier category X/X and X\X, if X satisfies one of the following conditions (items with [c]):
  [c] X is an atomic category;
  [c] X is a modifier itself.
  The newly created categories are inserted to the candidate set of the $(i-1)^{th}$ POS tag and the $(i+1)^{th}$ POS tag respectively.
- **Functor category**. For each pair of adjacent POS tags in the i^{th} and the $(i+1)^{th}$ position in each training sentence, and for each category X and Y associated with the two POS tags, we consider that X may take Y as argument to form the functor category X/Y, and Y may also take X as argument resulting in the functor Y\X. The new categories are valid if the head H (input category) and the argument A (result category) satisfy one of the following conditions (items with [c]) and violate none of the following restrictions (items with [r]):
  [c] H is a modifier or in the form of "(S|...)", and A is the atomic category "N" or "S";
  [c] H is "S" and A is "N", i.e. the categories "S/N" and "S\N" are allowed;
  [r] A is not a modifier, i.e. any non-modifier (atoms and functors) may be argument;
  [r] H is different from A, otherwise the result category is the modifier category rather than the functor category;
  [r] H is not "N", since the atomic category "N" is assumed to take no arguments.
  If the categories X/Y (or Y\X) passes the above tests, it is inserted to the candidate set of the i^{th} POS tag (and the corresponding $(i+1)^{th}$ POS tag).

After creating the lexicons for one sentence, we parse it with the created lexicons and remove categories that can not lead to a successful parse. The rest categories are inserted to the lexicon $\mathscr{L}^{(i)}$ for the i^{th} induction step. We perform this step twice to get the final lexicon $\mathscr{L}^{(2)}$.

The above induction procedure is almost the same as the algorithm described in (Bisk and Hockenmaier, 2012). One additional induction step they used is the "derived" induction step, in which adjacent constituents that can be derived from the existing lexicon are combined. However, their experiments do not show significant improvement of this lexicon generation method, so we omit this step in our experiments.

4 Improved CCG Induction Models

4.1 Basic Probabilistic Model

The *basic* model in this paper is the baseline model described in (Hockenmaier and Steedman, 2002), which is also used in (Bisk and Hockenmaier, 2012). There are four types of CCG rules:

1. the lexical (W) rules which generate terminal words;
2. the unary (U) rules which could be the root rules or the type-raising rules;
3. the left-headed (L) rules with the first child symbol as the head category, e.g. the forward composition rules;
4. the right-headed (R) rules with the second child symbol as the head category, e.g. the backward composition rules.

Binary trees are generated top-down recursively from the start symbol TOP. For each unexpanded nonterminal P, the basic model first generates the expansion type $\texttt{exp} \in \{\texttt{W},\texttt{U},\texttt{L},\texttt{R}\}$ according to $P_e(\texttt{exp}|\texttt{P})$. Then for each expansion type, the model generates either terminal word w or head child H and possible non-head child N:

$$\begin{aligned}
\text{Lexical:} \quad & P_e(\texttt{exp}=\texttt{W}|\texttt{P})\, P_w(\texttt{w}|\texttt{P},\texttt{exp}=\texttt{W}) \\
\text{Unary:} \quad & P_e(\texttt{exp}=\texttt{U}|\texttt{P})\, P_U(\texttt{H}|\texttt{P},\texttt{exp}=\texttt{U}) \\
\text{Left:} \quad & P_e(\texttt{exp}=\texttt{L}|\texttt{P})\, P_L(\texttt{H}|\texttt{P},\texttt{exp}=\texttt{L})\, P_l(\texttt{N}|\texttt{P},\texttt{H},\texttt{exp}=\texttt{L}) \\
\text{Right:} \quad & P_e(\texttt{exp}=\texttt{R}|\texttt{P})\, P_R(\texttt{H}|\texttt{P},\texttt{exp}=\texttt{R})\, P_r(\texttt{N}|\texttt{P},\texttt{H},\texttt{exp}=\texttt{R})
\end{aligned}$$

After the lexicon generation step, each POS tag acquires a lexicon of CCG categories. These lexicons and CCG rules are used to parse the training corpus. We use the Expectation Maximization (EM) algorithm to estimate model parameters for the basic model. The Inside-Outside algorithm (Lari and Young, 1990) is used to collect the expected counts in the E-step of the EM algorithm.

4.2 Boundary Models

Boundary POS tags have been proven useful for detecting phrase boundaries in the supervised setting (Xiong et al., 2010) and in the unsupervised grammar induction (Golland et al., 2012; Huang et al., 2012). We introduce this idea to the unsupervised combinatory categorial grammar induction. Since the system inputs are the golden POS tags in the treebank, we use the boundary words and the boundary POS tags interchangeably in this paper.

Particularly, in some parse tree T, we consider the boundary POS tags of each constituent and define the new probabilistic model as

$$\begin{aligned}
P(T) &= P_{CCG}(T) P_{BDR}(T) \\
&= \prod_{\text{rule:}r\in T} P_{CCG}(r) \prod_{\text{span:}\langle i,j\rangle\in T} P_{BDR}(\sigma_{\langle i,j\rangle}|B)
\end{aligned} \tag{4}$$

where distribution P_{CCG} is the basic CCG model defined in section 4.1, P_{BDR} is the proposed boundary model, $\sigma_{\langle i,j\rangle}$ means the boundary words of the constituent covered by span $\langle i,j\rangle$, and B is a special nonterminal representing the constituent spans. We denote this model as *basic+bdr*. Figure 1 shows a tree example. The boundary probability of this parse tree is

$$\begin{aligned}
P_{BDR}(T) = {} & P(\texttt{DT_DT}|B) \times P(\texttt{NNS_NNS}|B) \times P(\texttt{VBD_VBD}|B) \times P(\texttt{RB_RB}|B) \\
& \times P(\texttt{DT_NNS}|B) \times P(\texttt{VBD_RB}|B) \times P(\texttt{DT_RB}|B)
\end{aligned} \tag{5}$$

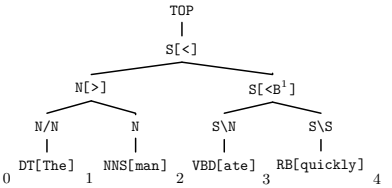


Figure 1: A tree example used to illustrate the boundary probability. The CCG rule types are given in the square brackets next to each nonterminal. Although only POS tags are considered in the induction model, we also show the words for clarity.

Currently, we restrict that the single special nonterminal B generates all boundary tag pairs. We have tried to let the boundary pairs depend on the category of tree nodes. For instance, for span $\langle 2,4\rangle$ of tree in Figure 1, we model $P(\texttt{VBD\_RB}|S)$ rather than $P(\texttt{VBD\_RB}|B)$. However, this category-dependent boundary model performs poor in experiments (not reported in this paper). The reason might be the data sparsity problem, since there are quite a lot of induced categories in the grammar.

4.3 Bayesian Models

The EM algorithm may overfit the training data, so we propose the Bayesian model to infer grammars and tree structures. In the Bayesian models, the generative process is often formulated as the Chinese Restaurant process (CRP) or the Pitman-Yor process (PYP) to encourage rule reuse and learn compact models (Teh et al., 2006; Pitman and Yor, 1997). Since the PYP is a generalization of the CRP and has more elegant and controllable behavior over the "long tail" of probability distributions, we focus on the PYP in this paper.

For each nonterminal category A in CCG, we maintain a cache to store the total number n of rules expanded with A as parent, the total number of different rule types m, and the counts n_k of each rule that has been generated for $k = 1,\ldots,m$. Initially, all caches are empty, i.e. with $n = m = 0$. The parse trees are generated in sequence. For each sentence, the PYP generates trees in the top-down fashion. For each nonterminal label to be expanded, we consult the cache associated with that nonterminal and decide whether to choose the k^{th} expanded rule in the cache, or generate a new rule. The probabilities of these two cases are

$$P_t(z|z_{i<n}) = \begin{cases} \frac{ma+b}{n+b} & \text{if } z_{n+1} = m+1 \\ \frac{n_k-a}{n+b} & \text{if } z_{n+1} = k, k \in \{1,\cdots,m\} \end{cases} \tag{6}$$

where z_i is the cache index of the i^{th} generated rule, $a \in [0,1]$ and $b \geq 0$ are two category-associated parameters naming the discount and concentration parameters respectively. Note that different labels may have different values of a and b. If we decide to generate a new rule, the new rule is sampled from the base multinomial distribution P_0. We also put a Dirichlet prior on the base distribution and sample the base rule probabilities from the Dirichlet distribution: $\boldsymbol{\theta} \sim \text{Dir}(\boldsymbol{\theta}|\boldsymbol{\alpha})$. The above sampling procedures are performed recursively down until all frontier labels have been expanded to terminals. For CCG induction models described in previous sections, PYP priors are put on all factored models, although they may have different hyperparameters.

The joint probability of a particular sequence of indexes $\boldsymbol{z}$ with cached counts $(n_1, \ldots, n_m)$ under the Pitman-Yor process is

$$PY(\boldsymbol{z}|a,b) = \frac{\prod_{k=1}^{m}(a(k-1)+b)\prod_{j=1}^{n_k-1}(j-a)}{\prod_{i=0}^{n-1}(i+b)}. \tag{7}$$

The above generative process demonstrates the "rich get richer" dynamics, i.e. previous sampled rules would be sampled more likely in following procedures. It is easy to verify that any permutation of $z_1, \ldots, z_n$ has the same probability in the Pitman-Yor process, so the Pitman-Yor process is *exchangeable*, resulting in efficient sampling methods. Given the parse tree set $\mathscr{T}$, we could integrate out the base distribution probabilities to get the joint PYP probability[5]:

$$P(\mathscr{T}|\boldsymbol{a},\boldsymbol{a},\boldsymbol{b}) = \prod_{\mathtt{X}\in\mathscr{N}} \frac{\mathrm{Beta}(\boldsymbol{a}_{\mathtt{X}} + \boldsymbol{f}_{\mathtt{X}})}{\mathrm{Beta}(\boldsymbol{a}_{\mathtt{X}})} PY(\boldsymbol{z}(\mathscr{T})|\boldsymbol{a},\boldsymbol{b}) \tag{8}$$

where $\mathscr{N}$ is the set of nonterminal categories, $\boldsymbol{f}_{\mathtt{X}}$ is the vector containing the number of occurrences that rules r with $\mathtt{X}$ as parent in $\mathscr{T}$, and Beta means the Beta function.

To infer trees and parameters of the PYP model, we apply the collapsed Metropolis-Hastings algorithm (Hastings, 1970; Johnson et al., 2007) to sample trees from the parse forests. In detail, we iteratively draw samples for each yield in training corpus in sequence. Assuming the current tree of the i^{th} sentence is T_i, we first remove this tree from the whole tree set to obtain $\mathscr{T}_{-i}$, the set of sampled trees except the i^{th} one. Then we draw new tree T_i' from some proposal distribution $Q(T_i'|\mathscr{T}_{-i})$, and accept the new sampled tree with probability

$$A(T_i, T_i') = \min\left\{1, \frac{P(T'|\alpha,\boldsymbol{a},\boldsymbol{b})\, Q(T_i|\mathscr{T}_{-i})}{P(T|\alpha,\boldsymbol{a},\boldsymbol{b})\, Q(T_i'|\mathscr{T}_{-i})}\right\}. \tag{9}$$

In theory, Q could be any distribution if it never assigns zero probability. In practice, the proposal distribution should be close enough to the true distribution to avoid high rejection rate. We use the following proposal distribution in experiments:

$$Q(T_i|\mathscr{T}_{-i}) = \frac{1}{Z(\mathscr{T}_{-i})} \prod_{\text{rule}:r\in T_i} P_t(z_r|z_{\mathscr{T}_{-i}})\, P_0(r|\alpha)^{\delta(r\notin\mathscr{T}_{-i})} \tag{10}$$

in which P_t is the conditional index probability in Equation 6, and the model needs to consult the base distribution P_0 if it encounters a new rule ($\delta(r \notin \mathscr{T}_{-i}) = 1$). We do not need to calculate the normalization constant $Z(\mathscr{T}_{-i})$ since it would be cancelled in Equation 9. The proposal distribution differs from the true distribution in the sense that: the caches are updated immediately after calculating probabilities of each rule in T_i under the true distribution, while the caches stay fixed in the proposal distribution evaluation. In experiments, we observe that only a tiny fraction (less than 1%) of proposals are rejected. This provides evidence that the proposal distribution works well enough. We use the sampling algorithm described in (Blunsom and Osborne, 2008) to draw a parse tree from the parse forest according to the proposal distribution Q.

[5]For simplicity, we omit probability factorization as if there is only one model. The complete probability expression is the product of multiple factored PYP probabilities.

5 Experiments

5.1 Datasets and Settings

We carry out experiments on the Wall Street Journal portion of the Penn English Treebank (Marcus et al., 1993). As the standard data split, we use sections 02-21 as the training set, section 00 as the development set, and section 23 as the final test set. We remove punctuations and null elements in treebank, as the standard preprocessing step in the previous unsupervised grammar induction approaches (Klein and Manning, 2002; Cohn et al., 2010; Bisk and Hockenmaier, 2012). For comparison, we build datasets with sentence lengths no more than 10 and 20 words after removing punctuations. As the standard machine learning pipeline, we perform learning and inference on the training set, select model with best performance on the development set, and report the result of the selected model on the test set. For efficiency, we only train and tune parameters on sentences with length no more than 10, but report performance on longer sentences as well. Table 1 gives the statistics of each dataset.

Dataset	Train		Dev		Test	
	# sent	# word	# sent	# word	# sent	# word
PTB10	5,899	41,701	265	1,875	398	2,649
PTB20	-	-	-	-	1,286	16,591

Table 1: Data statistics

For evaluation, the script of CoNLL 2008 shared task[6] is used to calculate the Unlabeled Attachment Score (UAS) of the system outputs, using the treebank dependency structures as golden standards. We perform the McNemar's significant test to compare our proposed models with the baseline model. Since the original treebank only has the phrase-structure trees, we use (Johansson and Nugues, 2007)'s code[7] to convert the treebank to dependency structures. In order to compare with existing approaches, we follow (Bisk and Hockenmaier, 2012) to convert the CCG trees to dependency trees: (1) the modifier categories are treated as the dependents of their heads; (2) the head of the sentence is treated as a dependent of a root node at position 0; (3) the left part of conjunction is treated as the head of `conj`, and the `conj` is treated as the head of the right part. Figure 2 shows an example.

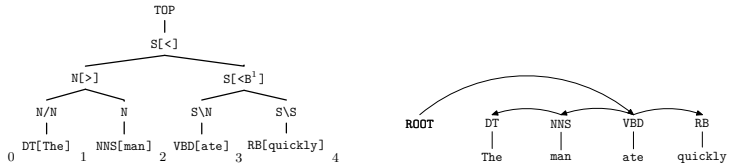


Figure 2: Left: a CCG tree example. Right: the converted dependency tree.

To reduce the model complexity, we restrict that the maximal order of composition rule is 2. The rule probabilities are initialized uniformly. For EM-based models (basic and basic+bdr), we add fixed value to expected counts in each E-step as smoothing. We perform maximal 40 EM iterations while stop earlier if the development score starts to drop. In the Bayesian

[6]Available at: `http://barcelona.research.yahoo.net/dokuwiki/doku.php?id=conll2008:software`
[7]Available at: `http://nlp.cs.lth.se/software/treebank_converter`

inference, we run sampler through the whole training sentences for 400 iterations and use the last sampled grammars to parse fresh sentences. To model the uncertainty of hyperparameters, we put an uninformative Beta$(1,1)$ prior on $\boldsymbol{a}$ and a "vague" Gamma$(10, 0.1)$ prior on $\boldsymbol{b}$ instead of setting them empirically. After each iteration, we resample each of hyperparameters from the posterior distribution of hyperparameters using a slice sampler (Neal, 2003).

5.2 Results

Before presenting the final results, we first examine the effect of smoothing values for EM models. We test smoothing values from $\{1, 10, 20, 30, 40, 50, 60, 70, 80, 90, 100\}$ and evaluate the unlabeled attachment scores (UAS) of the basic model and the basic+bdr model on both the development set and the test set. Note the performance on the test set is only used for references, the final smoothing value is selected as the one with best performance on the development set. Experimental results are plotted in Figure 3. We can easily find that the best smoothing value (with highest dev-score) is 20 for both of these models. The basic+bdr model achieves significant (at $p < 10^{-3}$ level) better results (dev: 66.3, tst: 66.7) than the basic model (dev: 63.3, tst: 62.9) on both the development and test sets when the optimal smoothing values are selected.

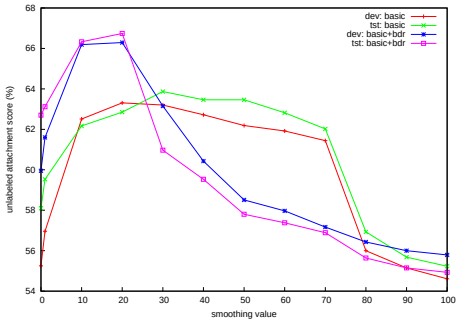


Figure 3: The effect of smoothing value on development and test set of PTB10

The final results of basic and basic+bdr models, using EM or Pitman-Yor process (PYP) are shown in Table 2 for comparison. Some existing results (copied from Figure 4(a) in (Bisk and Hockenmaier, 2012)) are also given in this table. Comparing within the four models described in this paper, we draw following conclusions:

1. The basic+bdr model achieves significant better results than the basic model under the EM learning. The boundary words capture lexical information about constituents and show complementary effectiveness for CCG induction.
2. The Bayesian framework outperforms the EM baseline significantly. This provides evidence that the compact models are preferred in the unsupervised CCG induction.
3. The combination of the boundary model and the Bayesian inference only show slightly better results than individual components. The reason might be that both the boundary words and the Bayesian model have the same effects and give high probabilities to those parse trees with more reused rules.

4. For longer sentences, the proposed methods still outperform baseline model with a large gap, demonstrating the robustness of our method.

Model		PTB10	PTB20
(Klein and Manning, 2004)		47.5	-
(Headden III et al., 2009)		68.8	-
(Spitkovsky et al., 2010)		65.3*	53.8*
(Cohn et al., 2010)		65.9	58.3
(Bisk and Hockenmaier, 2012)		71.5	60.3
(Naseem et al., 2010)		71.9	50.4*
EM	basic	62.9	49.9
	basic+bdr	66.7$^+$	54.0$^+$
PYP	basic	66.0$^+$	53.9$^+$
	basic+bdr	66.7$^+$	55.1$^+$

Table 2: Comparison results on the test set with various length limits. Results of existing approaches are copied from (Bisk and Hockenmaier, 2012). Results with (*) were obtained with additional training data. Results with ($^+$) outperform the baseline (basic with EM) results significantly at $p < 10^{-3}$ level according to the McNemar's significant test.

Compared with existing approaches, our models stay in the intermediate level. Headden III et al. (2009) use rich contexts, words as well as POS tags, and sophisticated smoothing techniques, which might explain their higher performance than ours on short sentences. Naseem et al. (2010) manually specify some dependency rules in experiments, while we only use some coarse restrictions on the lexicon and grammar generation. Our models are mainly based on the previous work (Bisk and Hockenmaier, 2012). The full-EM model in (Bisk and Hockenmaier, 2012) corresponds to the basic model in our paper. Their reported results of full-EM are around 55−60 on short sentences, lower than our implementation. However, the best model in their paper outperforms our models on both short and long sentences. They achieve the state-of-the-art performance using the k-best EM learning. It should be emphasised that the k-best EM learning strategy is still applicable to our proposed basic+bdr model, which we leave for future work.

5.3 Discussion and Future work

Our method and many previous approaches (Klein and Manning, 2002, 2004; Bisk and Hockenmaier, 2012) take the golden part-of-speech tags as input. This practice may reduce data sparsity problem caused by directly modelling words. However, this may also lose useful lexical information. As reported in (Headden III et al., 2009), incorporating words with high frequencies (greater than 100 times in their experiments) as well as the POS tags could improve the induction accuracy for dependency models. In CCG, words may also help to distinguish lexical categories. For example, the transitive verbs are often tagged as (S\N)/N and the intransitive verbs often have category S\N. However, these syntactic differences are not encoded in the Penn treebank POS tags, in which they may both have the POS tag $\texttt{VB}_x$ depending on the tenses. We leave this extension for future work.

Although the simple additive smoothing methods could improve EM results (see Figure 3), sophisticated smoothing schemes are also applicable (Headden III et al., 2009). Currently, the

final probability is the product of basic CCG model and boundary model. This simple strategy already shows effectiveness in our experiments. In future work, different interpolation or back-off methods will be investigated. In addition, the context words have been proved useful for constituency tree induction (Klein and Manning, 2002; Golland et al., 2012). We could also integrate the context information to help CCG induction.

The performance of Bayesian model is somehow below our expectation, especially for the basic+bdr model. Currently, we simply use the grammars sampled at the last iteration to parse test sentences. Johnson and Goldwater (2009) propose the *maximum marginal decoding* technique to obtain more stable results, which we will explore in the future. Furthermore, we do not elaborately tune hyperparameters of Bayesian model, such as the prior distribution of $\boldsymbol{a}, \boldsymbol{b}$, the value of $\boldsymbol{\alpha}$, etc. Finally, tree nodes tend to be labeled with common and simple categories in CCGbank (Hockenmaier and Steedman, 2007). We could define probability models over the number of the categories and the internal arity of categories, and put sparse priors to enforce compact model.

6 Related work

Unsupervised dependency grammar induction has attracted a lot of research interests. The dependency model with valence (DMV) (Klein and Manning, 2004; Headden III et al., 2009; Cohen and Smith, 2009) is one of representative work, in which the valence is explicitly modelled. In contrast, the CCG formalism encodes the functor arity and word orders via syntactic categories, providing a more syntax-meaningful representation especially for long-range dependencies (Steedman, 2000). Since our work is based on CCG induction, we only present CCG-related work.

Osborne and Briscoe (1997) propose an unsupervised learning model for CCG induction. They create a labeled binary tree for each part-of-speech tag sequences in a greedy, bottom-up, incremental manner. The label of each inner node is the label of either the left or right sub-node. To avoid overfitting, they apply the Minimum Description Length (MDL) principle to learn compact grammars with minimal length of hypothesis and minimal length of data encoded in the hypothesis. While our model uses the alternative Bayesian learning method to learn compact grammars. Watkinson and Manandhar (1999) describe a CCG induction model based on linguistic lexicon generation. The learner is provided with a set of manually defined English oriented CCG categories, such as the verb-subcategorization. Compared to their work, we initialize lexicons with more general categories and learn complex categories and grammar rules automatically. Ponvert (2007) presents a generic algorithm to learn CCG categories. However, the reported experiments do not show much promising results. Naseem et al. (2010) use manually-specified linguistic-motivated rules in dependency grammar induction. Variational Bayesian method is used to estimate the parameters.

The most related work is (Bisk and Hockenmaier, 2012). The grammar generation step described in our paper is almost the same as the one in their paper. They compare various EM settings (full EM, Viterbi EM, and k-best EM) and find that the k-best EM could achieve best performance. They report the state-of-the-art results for unsupervised dependency grammar induction. Instead of the k-best EM, we perform the Bayesian inference and use sampling to estimate parameters. In addition, we exploit the use of rich lexical information and propose the boundary model to improve CCG induction. It is worth noting that the k-best EM can be also used for our boundary model, which we leave for future work.

Conclusion

In this paper, we have proposed to incorporate lexical information in the unsupervised CCG induction. Specifically, an additional boundary model is defined to capture complex language aspects, in which boundary words are generated from a special symbol independently for each span covered by tree nodes. Furthermore, we describe the nonparametric Pitman-Yor process to encourage rule reuse, resulting in compact grammars. Experimental results demonstrate that both the boundary model and the Bayesian inference outperform the baseline CCG induction system.

Acknowledgments

We would like to thank Zhixiang Ren for insightful discussions. We would also thank Professor Julia Hockenmaier for her LaTeX package to draw CCG derivations. Thank the anonymous reviewers for their helpful comments and suggestions.

References

Bisk, Y. and Hockenmaier, J. (2012). Simple robust grammar induction with combinatory categorial grammar. In *Proceedings of the Twenty-Sixth Conference on Artificial Intelligence (AAAI-12)*, pages 1643–1649, Toronto, Canada.

Blunsom, P. and Baldwin, T. (2006). Multilingual deep lexical acquisition for HPSGs via supertagging. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, pages 164–171, Sydney, Australia.

Blunsom, P. and Cohn, T. (2010). Unsupervised induction of tree substitution grammars for dependency parsing. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 1204–1213, Cambridge, MA.

Blunsom, P. and Osborne, M. (2008). Probabilistic inference for machine translation. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 215–223, Honolulu, Hawaii.

Bod, R. (2006). An all-subtrees approach to unsupervised parsing. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 865–872, Sydney, Australia.

Boonkwan, P. and Steedman, M. (2011). Grammar induction from text using small syntactic prototypes. In *Proceedings of 5th International Joint Conference on Natural Language Processing*, pages 438–446, Chiang Mai, Thailand.

Bos, J., Clark, S., Steedman, M., Curran, J. R., and Hockenmaier, J. (2004). Wide-coverage semantic representations from a CCG parser. In *Proceedings of Coling 2004*, pages 1240–1246, Geneva, Switzerland.

Clark, S. and Curran, J. (2003). Log-linear models for wide-coverage CCG parsing. In *Proceedings of the 2003 Conference on Empirical Methods in Natural Language Processing*, pages 97–104.

Clark, S. and Curran, J. R. (2007). Wide-coverage efficient statistical parsing with CCG and log-linear models. *Computational Linguistics*, 33(4):493–552.

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

Cohn, T., Blunsom, P., and Goldwater, S. (2010). Inducing Tree-Substitution grammars. *Journal of Machine Learning Research*, 11:3053–3096.

Cohn, T., Goldwater, S., and Blunsom, P. (2009). Inducing compact but accurate tree-substitution grammars. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 548–556, Boulder, Colorado.

Golland, D., DeNero, J., and Uszkoreit, J. (2012). A feature-rich constituent context model for grammar induction. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 17–22, Jeju Island, Korea.

Hassan, H., Sima'an, K., and Way, A. (2007). Supertagged phrase-based statistical machine translation. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 288–295, Prague, Czech Republic.

Hastings, W. K. (1970). Monte Carlo sampling methods using Markov chains and their applications. *Biometrika*, 57(1):97–109.

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

Hockenmaier, J. and Steedman, M. (2002). Generative models for statistical parsing with combinatory categorial grammar. In *Proceedings of 40th Annual Meeting of the Association for Computational Linguistics*, pages 335–342, Philadelphia, Pennsylvania, USA.

Hockenmaier, J. and Steedman, M. (2007). CCGbank: A corpus of CCG derivations and dependency structures extracted from the Penn Treebank. *Computational Linguistics*, 33(3):355–396.

Huang, Y., Zhang, M., and Tan, C. L. (2012). Improved constituent context model with features. In *Proceedings of the 26th Pacific Asia Conference on Language, Information and Computation (to appear)*, Bali, Indonesia.

Johansson, R. and Nugues, P. (2007). Extended constituent-to-dependency conversion for English. In *Proceedings of the 16th Nordic Conference of Computational Linguistics*, pages 105–112, Tartu, Estonia.

Johnson, M. and Goldwater, S. (2009). Improving nonparameteric bayesian inference: experiments on unsupervised word segmentation with adaptor grammars. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 317–325, Boulder, Colorado.

Johnson, M., Griffiths, T. L., and Goldwater, S. (2007). Adaptor grammars: A framework for specifying compositional nonparametric bayesian models. In *Advances in Neural Information Processing Systems 19*, pages 641–648, Cambridge, MA.

Jones, B. K., Johnson, M., and Frank, M. C. (2010). Learning words and their meanings from unsegmented child-directed speech. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 501–509, Los Angeles, California.

Klein, D. and Manning, C. (2004). Corpus-based induction of syntactic structure: Models of dependency and constituency. In *Proceedings of the 42nd Meeting of the Association for Computational Linguistics (ACL'04), Main Volume*, pages 478–485, Barcelona, Spain.

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

Lari, K. and Young, S. J. (1990). The estimation of stochastic context-free grammars using the Inside-Outside algorithm. *Computer Speech and Language*, 4:35–56.

Marcus, M. P., Marcinkiewicz, M. A., and Santorini, B. (1993). Building a large annotated corpus of English: the Penn Treebank. *Computational Linguistics*, 19(2):313–330.

Naseem, T., Chen, H., Barzilay, R., and Johnson, M. (2010). Using universal linguistic knowledge to guide grammar induction. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 1234–1244, Cambridge, MA.

Neal, R. M. (2003). Slice sampling. *Annals of Statistics*, 31(3):705–767.

Osborne, M. and Briscoe, T. (1997). Learning stochastic categorial grammars. In *Proceedings of CoNLL97: Computational Natural Language Learning*, pages 80–87.

Pitman, J. and Yor, M. (1997). The two-parameter Poisson-Dirichlet distribution derived from a stable subordinator. *Annals of Probability*, 25:855–900.

Ponvert, E. (2007). Inducing combinatory categorial grammars with genetic algorithms. In *Proceedings of the ACL 2007 Student Research Workshop*, pages 7–12, Prague, Czech Republic.

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

Spitkovsky, V. I., Alshawi, H., Jurafsky, D., and Manning, C. D. (2010). Viterbi training improves unsupervised dependency parsing. In *Proceedings of the Fourteenth Conference on Computational Natural Language Learning*, pages 9–17, Uppsala, Sweden.

Steedman, M. (2000). *The Syntactic Process*. Cambridge, MA, USA.

Steedman, M. and Baldridge, J. (2011). Combinatory categorial grammar. In Borsley, R. and Börjars, K., editors, *Non-Transformational Syntax: Formal and Explicit Models of Grammar*, pages 181–224.

Teh, Y. W., Jordan, M. I., Beal, M. J., and Blei, D. M. (2006). Hierarchical Dirichlet processes. *Journal of the American Statistical Association*, 101(476):1566–1581.

van Zaanen, M. (2000). ABL: Alignment-based learning. In *Proceedings of the 18th International Conference on Computational Linguistics (Coling 2000)*, volume 2, pages 961–967, Saarbrücken, Germany.

Vijay-Shanker, K. and Weir, D. (1994). The equivalence of four extensions of context-free grammars. *Mathematical Systems Theory*, 27(6):511–546.

Watkinson, S. and Manandhar, S. (1999). Unsupervised lexical learning with categorial grammars using the LLL corpus. In *Proceedings of the 1st Workshop on Learning Language in Logic*.

Xiong, D., Zhang, M., and Li, H. (2010). Learning translation boundaries for phrase-based decoding. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 136–144, Los Angeles, California.

Zettlemoyer, L. and Collins, M. (2007). Online learning of relaxed CCG grammars for parsing to logical form. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 678–687, Prague, Czech Republic.

Zhang, Y. and Clark, S. (2011). Syntax-based grammaticality improvement using CCG and guided search. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 1147–1157, Edinburgh, Scotland, UK.

Mining Rules for Rewriting States in a Transition-based Dependency Parser for English

Akihiro Inokuchi and Ayumu Yamaoka
The Institute of Scientific and Industrial Research, Osaka University
8-1 Mihogaoka, Ibaraki, Osaka 567-0047, JAPAN
{inokuchi, yamaoka}@ar.sanken.osaka-u.ac.jp

ABSTRACT
Recently, methods for mining graph sequences have attracted considerable interest in data-mining research. A graph sequence is a data structure used to represent changing networks. The aim of graph sequence mining is to enumerate common changing patterns appearing more frequently than a given threshold in graph sequences. Dependency analysis is recognized as a basic process in natural language processing. In transition-based parsers for dependency analysis, a transition sequence can be represented by a graph sequence, where each graph, vertex, and edge corresponds to a state, word, and dependency, respectively. In this paper, we propose a method for mining rules to rewrite states reaching incorrect final states to those reaching correct final states, from transition sequences of a dependency parser using a beam search. The proposed method is evaluated using an English corpus, and we demonstrate the design of effective feature templates based on knowledge obtained from the mined rules.

KEYWORDS: Dependency Parsing, Graph Sequence, Frequent Pattern Mining.

Proceedings of COLING 2012: Technical Papers, pages 1275–1290,
COLING 2012, Mumbai, December 2012.

1 Introduction

Data mining is the process of mining useful knowledge from large datasets. Recently, methods for mining graph sequences (dynamic graphs (Borgwardt et al., 2006) or evolving graphs (Berlingerio et al., 2009)) have attracted considerable interest from researchers in the field of data mining (Inokuchi and Washio, 2008). For example, human networks can be represented as a graph, where each vertex and edge corresponds, respectively, to a human and a relationship in the network. If a human joins or leaves the network, the numbers of vertices and edges in the graph increase or decrease, respectively. A graph sequence is one of the data structures used to represent a changing network. Figure 1(a) shows a graph sequence consisting of four steps, five vertices, and various edges between the vertices. The aim of graph sequence mining is to enumerate subgraph subsequence patterns, an example of which is shown in Fig. 1(b), appearing more frequently than a given threshold in graph sequences. Since the development of methods for mining graph sequences, these methods have been applied, for example, to social networks in Web services (Berlingerio et al., 2009), article-citation networks (Ahmed and Karypis, 2011), and e-mail networks (Borgwardt et al., 2006).

Dependency parsing is considered a basic process in natural language processing (NLP), and a number of studies have been reported (Kudo and Matsumoto, 2002; Nivre, 2008). One reason for the increasing popularity of this research area is the fact that dependency-based syntactic representations seem to be useful in many applications of language technology (Kubler et al., 2009), such as machine translation (Culotta and Sorensen, 2004) and information extraction (Ding and Palmer, 2004). Broadly speaking, dependency parsers can be categorized as transition-based, graph-based, and grammar-based dependency parsers. Transition-based dependency parsers, which are data-driven methods, transit between states in a deterministic way using local state information. If the parser adds an incorrect dependency between words once, it never reaches the correct final state. To reduce such incorrect decisions, the parser can keep track of multiple candidate outputs using the beam search principle, thus avoiding making decisions too early (Zhang and Clark, 2008).

In a transition-based parser, a transition sequence can be represented by a graph sequence, where each graph, vertex, and edge, corresponds to a state, word, and dependency, respectively. By mining characteristic patterns from transition sequences for sentences analyzed incorrectly by a parser, it is possible to design new parsers and generate effective feature templates in the machine learner of the parser to avoid incorrect dependency structures. In this paper, we demonstrate the application of graph sequence mining to dependency parsing in NLP.

We propose a method for mining rewriting rules from transition sequences of an arc-eager dependency parser integrated with the beam search principle for English sentences. The mined rewriting rules can shed light on why incorrect dependency structures are returned by this type of parser. We also present effective feature templates designed according to knowledge obtained from the mined rules, and show the improvement of the parser's attachment score, which is a measure of the percentage of words with the correct heads. To mine such rules, the rules should be human-readable, since they are to be used as inspiration for the engineering

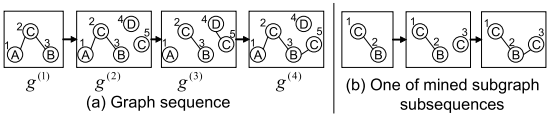


Figure 1: Examples of a graph sequence and a mined frequent pattern

Parse$(x = \langle w_0, w_1, \cdots, w_n \rangle)$
1) $c \leftarrow c_s(x)$
2) while $c \notin C_F$
3) $c \leftarrow [o(c)](c)$
4) return c

Figure 2: Dependency parser based on a transition system

Transitions	
Left	$(\sigma \vert i, j \vert \beta, A) \Rightarrow (\sigma, j \vert \beta, A \cup \{(j,i)\})$
Right	$(\sigma \vert i, j \vert \beta, A) \Rightarrow (\sigma \vert i \vert j, \beta, A \cup \{(i,j)\})$
Reduce	$(\sigma \vert i, \beta, A) \Rightarrow (\sigma, \beta, A)$
Shift	$(\sigma, j \vert \beta, A) \Rightarrow (\sigma \vert j, \beta, A\})$
Preconditions	
Left	$i \neq 0 \wedge \nexists k$ s.t. $(k,i) \in A$
Right	$\nexists k$ s.t. $(k,j) \in A$
Reduce	$\exists k$ s.t. $(k,i) \in A$

Figure 3: Transitions for an arc-eager parser

of new feature templates of the parser.

2 Transition-based Dependency Parsing

In this paper, we focus on dependency analysis using an "arc-eager parser" (Nivre, 2008), which is a parser based on a transition system, for "English sentences". However, the principle of the method proposed in this paper can basically be applied to any parser based on a transition system for sentences in any language (Inokuchi et al., 2012).

The aim of dependency parsing of a sentence is to output its dependency graph.

Definition 1 *The dependency graph for a sentence* $x = \langle w_0, \cdots, w_n \rangle$ *is represented as a graph* $g = (V, E)$*, where* $V = \{0, \cdots, n\}$ *and* $E \subset V \times V$. ■

Definition 2 *A dependency graph* (V, E) *is well-formed, if the following conditions are satisfied:*

- $\nexists x \in V$ *such that* $(x, 0) \in E$ *(root condition),*
- $\nexists x \in V$ *such that* $(x, y) \in E \wedge x \neq x'$*, when* $(x', y) \in E$ *(single-head condition), and*
- $\nexists \{(v_0, v_1), (v_1, v_2), \cdots, (v_{l-1}, v_l)\} \subseteq E$ *such that* $v_0 = v_l$ *(acyclicity condition).* ■

A dependency graph satisfying these conditions is a forest. In addition, if a dependency graph is connected, the graph is a tree.

We define a transition-based dependency parser with input $x = \langle w_0, w_1, \cdots, w_n \rangle$ and output $g = (V, E)$.

Definition 3 *A transition-based parser consists of* $S = (C, T, c_s, C_F)$*, where*

- $C = \{(\sigma, \beta, A)\}$ *is a set of states, with* σ*,* β*, and* A *a stack, a buffer, and a set of edges, respectively,*
- T *is a set of transitions, with* $t \in T$ *a partial function such that* $t : C \rightarrow C$*,*
- c_s *is an initial function satisfying* $c_s(x) = ([0], [1, 2, \cdots, n], \emptyset)$*, and*
- $C_F \subseteq C$ *is a set of final states* $\{c \in C \mid c = (\sigma, [\,], A)\}$. ■

A transition sequence for $x = \langle w_0, w_1, \cdots, w_n \rangle$ on $S = (C, T, c_s, C_F)$ is represented as $C_{1,m} = \langle c^{(1)}, \cdots, c^{(m)} \rangle$, satisfying (1) $c^{(1)} = c_s(x)$, (2) $c^{(m)} \in C_F$, and (3) $\exists t \in T$ for $c^{(i)}$ $(1 \leq i < m)$, $c^{(i+1)} = t(c^{(i)})$. We denote β and A for state c as β_c and A_c, respectively.

Figure 2 gives the algorithm for a transition-based dependency parser, where o denotes an oracle for selecting $t = [o(c)]$ to transit to the next state in a deterministic way. In particular,

the arc-eager parser, which is a transition-based parser, selects either Left, Right, Reduce, or Shift to analyze sentences, as shown in Fig. 3, where the operator | is taken to be left-associative for the stack and right-associative for the buffer. If o selects Left or Right, then edge (j,i) or (i,j) is added to A to transit from c to $t(c)$, respectively. If o selects Reduce, then i is popped from the stack to transit from c to $t(c)$. Otherwise, j is popped from the buffer and j is pushed onto the stack to transit from c to $t(c)$. Since o is a function that determines the transition from Left, Right, Reduce, and Shift, it is implemented using a multi-class classifier for feature vectors characterizing the state c (Kubler et al., 2009).

Although we defined each state using a stack and buffer in a similar way to that reported in most of the literature, we now redefine it using a graph to link dependency parsing to graph sequence mining.

Definition 4 *A state* $c = (\sigma, j|\beta, A)$ *s.t.* $\sigma = [s_{|\sigma|}, s_{|\sigma|-1}, \cdots, s_1]$ *is represented as a graph* $c_g = (N, A, N')$*, where* $N = \{0, 1, \cdots, j\}$ *is a set of vertices,* A *is a set of edges, and* $N' = \{s_{|\sigma|}, s_{|\sigma|-1}, \cdots, s_1, j\} \subseteq N$.

The graph $c_g = (N, A, N')$ is a forest of ordered trees, where $N' \subseteq N$ are vertices that are not reduced on the rightmost path of each tree in the forest. If o selects Left or Right, then edge (j,i) or (i,j), where i and j $(i < j)$ are the largest vertices in N', is added to transit from c to $t(c)$, respectively. If o selects Shift, the smallest vertex that does not exist in N is added to c_g to transit from c to $t(c)$. Otherwise, the second largest vertex in N' is removed from N' to transit from c to $t(c)$. Since an arc-eager dependency parser is incremental (Kubler et al., 2009), the numbers of vertices and edges in c_g increase monotonically.

Example 1 *Figure 4 shows the transition sequence from the initial state to the final state for the sentence* $\langle \$, I, saw, him, . \rangle$*, where* $w_0 = \$$ *is a special root vertex. In the sequence, Shift, Left, Right, Right, Reduce, and Right are selected in order by o. In this figure, shaded vertices in each state belong to* N'.

Figure 5 shows the search space T for sentence $\langle w_0, w_1, w_2, w_3 \rangle$. The words in each state and all states whose final states are not trees are omitted owing to lack of space. The search space T for the algorithm given in Figs. 2 and 3 is depicted as a single-rooted directed acyclic graph, where states C on $S = (C, T, c_s, C_F)$ are nodes, initial state $c^{(1)}$ is the root node, final states $C_F \subseteq C$ are leaves, and transitions between the states are branches. As shown in Fig. 5, there is only one or a few transition sequences from the initial state to each leaf. Therefore, if an incorrect dependency is added between words once, the parser never reaches the correct final state. To reduce the possibility of such an incorrect decision, the parser can keep track of multiple candidate outputs using the beam search principle and avoid making decisions too early (Zhang and Clark, 2008). Nevertheless, an arc-eager parser incorporating the beam search principle sometimes reaches an incorrect final state.

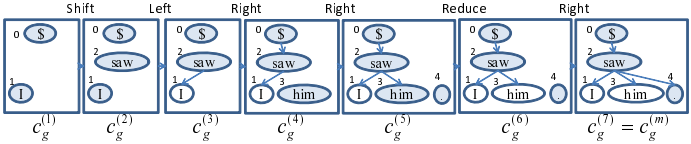


Figure 4: Transition sequence for the sentence $\langle \$, I, saw, him, . \rangle$

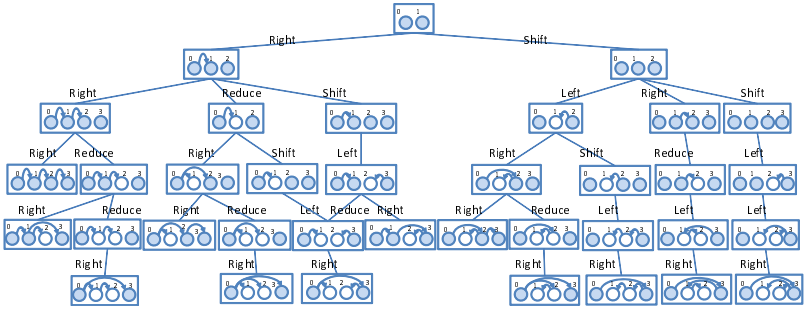
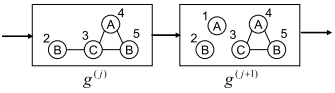


Figure 5: Search space for sentence $\langle w_0, w_1, w_2, w_3 \rangle$



$g^{(j)}$ $g^{(j+1)}$

Figure 6: Change between two successive graphs

In this paper, we propose a method for mining rewriting rules from transition sequences of an arc-eager dependency parser incorporating the beam search principle for English sentences. The rewriting rules to be mined are human-readable rules for rewriting states reaching incorrect final states to those reaching the correct final states. The rewriting rules correspond to bypasses between states in the search space shown in Fig. 5. To describe the proposed method, we first discuss GTRACE for mining graph sequences corresponding to transition sequences in the next section.

3 Graph Sequence Mining

Figure 1(a) shows an example of a graph sequence. Graph $g^{(j)}$ is the j-th labeled graph in the sequence. The problem we address in this section is how to mine patterns that appear more frequently than a given threshold from a set of graph sequences. We proposed using transformation rules to represent graph sequences compactly under the assumption that "change is gradual" (Inokuchi and Washio, 2008). In other words, only a small part of the structure changes, while the other part remains unchanged between successive graphs $g^{(j)}$ and $g^{(j+1)}$ in a graph sequence. For example, the change between successive graphs $g^{(j)}$ and $g^{(j+1)}$ in the graph sequence shown in Fig. 6 is represented as an ordered list of two transformation rules $\langle vt^{(j)}_{[1,A]}, ed^{(j)}_{[(2,3),\bullet]} \rangle$. This list denotes that a vertex with ID 1 and label A is inserted (vi), and then the edge between the vertices with IDs 2 and 3 is deleted (ed). By assuming that the change in each graph is gradual, we can represent a graph sequence compactly, even if the graph in the graph sequence has many vertices and edges. We also proposed a method, called GTRACE (Graph TRAnsformation sequenCE mining), for mining all frequent patterns from ordered lists of transformation rules. A transition sequence in the dependency parser is represented as a graph sequence. In addition, since any change between two successive graphs in the graph sequence comprises at most two changes, the assumption holds.

A labeled graph g is represented as $g = (V, E, L, l)$, where $V = \{1, \cdots, n\}$ is a set of vertices, $E \subseteq V \times V$ is a set of edges, and L is a set of labels such that $l : V \cup E \to L$. In addition, a graph sequence is an ordered list of labeled graphs and is represented as $d = \langle g^{(1)}, \cdots, g^{(z)} \rangle$.

Table 1: TRs used to represent a graph sequence

Vertex Insertion $vi^{(j,k)}_{[u,l]}$	Insert vertex u with label l into $g^{(j,k)}$ to transform to $g^{(j,k+1)}$.
Vertex Deletion $vd^{(j,k)}_{[u,\bullet]}$	Delete an isolated vertex u in $g^{(j,k)}$ to transform to $g^{(j,k+1)}$.
Vertex Relabeling $vr^{(j,k)}_{[u,l]}$	Relabel the label of vertex u in $g^{(j,k)}$ as l to transform to $g^{(j,k+1)}$.
Edge Insertion $ei^{(j,k)}_{[(u_1,u_2),l]}$	Insert an edge with label l between vertices u_1 and u_2 in $g^{(j,k)}$ to transform to $g^{(j,k+1)}$.
Edge Deletion $ed^{(j,k)}_{[(u_1,u_2),\bullet]}$	Delete an edge between vertices u_1 and u_2 in $g^{(j,k)}$ to transform to $g^{(j,k+1)}$.
Edge Relabeling $er^{(j,k)}_{[(u_1,u_2),l]}$	Relabel a label of an edge between vertices u_1 and u_2 in $g^{(j,k)}$ as l to transform to $g^{(j,k+1)}$.

To represent a graph sequence compactly, we focus on the differences between two successive graphs $g^{(j)}$ and $g^{(j+1)}$ in the sequence.

Definition 5 *The differences between graphs $g^{(j)}$ and $g^{(j+1)}$ in d are interpolated by a virtual sequence $d^{(j)} = \langle g^{(j,1)}, \cdots, g^{(j,m_j)} \rangle$, where $g^{(j,1)} = g^{(j)}$ and $g^{(j,m_j)} = g^{(j+1)}$. The graph sequence d is represented by $d = \langle d^{(1)}, \cdots, d^{(z-1)} \rangle$.* ■

The order of graphs $g^{(j)}$ represents the order of the graphs in an observed sequence. On the other hand, the order of graphs $g^{(j,k)}$ is the order of graphs in the artificial graph sequences, and there can be various artificial graph sequences between graphs $g^{(j)}$ and $g^{(j+1)}$. We limit the artificial graph sequences to be compact and unambiguous by taking the one with the shortest length in terms of the graph edit distance to reduce both the computational and spatial costs.

Definition 6 *Let the transformation of a graph by either insertion, deletion, or relabeling of a vertex or an edge be a unit, and let each unit have edit distance 1. A graph sequence $d^{(j)} = \langle g^{(j,1)}, \cdots, g^{(j,m_j)} \rangle$ is defined as an artificial graph sequence in which the edit distance between any two successive graphs is 1 and the edit distance between any two graphs is the minimum.* ■

Transformations are represented in this paper by the following "transformation rule (TR)".

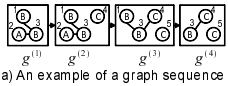
Definition 7 *A TR transforming $g^{(j,k)}$ to $g^{(j,k+1)}$ is represented by $tr^{(j,k)}_{[o_{jk},l_{jk}]}$, where*

- *tr is the transformation type that is either insertion, deletion, or relabeling of a vertex or edge,*
- *o_{jk} is the vertex or edge to which the transformation is applied, and*
- *$l_{jk} \in L$ is a label to be assigned to the vertex or edge in the transformation.* ■

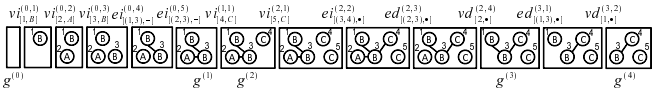
For the sake of simplicity, we simplify $tr^{(j,k)}_{[o_{jk},l_{jk}]}$ to $tr^{(j,k)}_{[o,l]}$ using the six TRs listed in Table 1. In summary, we define a transformation sequence as follows.

Definition 8 *A graph sequence $d^{(j)} = \langle g^{(j,1)}, \cdots, g^{(j,m_j)} \rangle$ is represented by $seq(d^{(j)}) = \langle tr^{(j,1)}_{[o,l]}, \cdots, tr^{(j,m_j-1)}_{[o,l]} \rangle$. Moreover, a graph sequence $d = \langle g^{(1)}, \cdots, g^{(z)} \rangle$ is represented by a transformation sequence $seq(d) = \langle seq(d^{(0)}), \cdots, seq(d^{(z-1)}) \rangle$.* ■

The notation for transformation sequences is far more compact than the original graph-based representation since only differences between two successive graphs in d are kept in the sequence. In addition, any graph sequence can be represented by the six TRs in Table 1.



(a) An example of a graph sequence d.



(b) Representation of the graph sequence d by using TRs.

Figure 7: Graph sequence and its TRs

Example 2 *The graph sequence d in Fig. 7(a) can be represented by a sequence of insertions and deletions of vertices and edges, as shown in Fig. 7(b). The transformation sequence is represented as* $\langle vi^{(0,1)}_{[1,B]} vi^{(0,2)}_{[2,A]} vi^{(0,3)}_{[3,B]} ei^{(0,4)}_{[(1,3),-]} ei^{(0,5)}_{[(2,3),-]} vi^{(1,1)}_{[4,C]} vi^{(2,1)}_{[5,C]} ei^{(2,2)}_{[(3,4),-]} ed^{(2,3)}_{[(2,3),\bullet]} vd^{(2,4)}_{[2,\bullet]} ed^{(3,1)}_{[(1,3),\bullet]} vd^{(3,2)}_{[1,\bullet]} \rangle$*, where "–" denotes an edge label.*

When transformation sequence s'_d is a subsequence of transformation sequence s_d, denoted as $s'_d \sqsubseteq s_d$, there is a mapping ϕ from vertex IDs in s'_d to those in s_d. We omit a detailed definition thereof owing to lack of space (see (Inokuchi and Washio, 2008) for the details). Given a set of graph sequences $DB = \{d \mid d = \langle g^{(1)}, \cdots, g^{(z)} \rangle\}$, we define a support $sup(s_p)$ of transformation sequence s_p as $sup(s_p) = |\{d \mid d \in DB, s_p \sqsubseteq seq(d)\}|/|DB|$. We call a transformation sequence whose support is no less than the minimum support sup', a frequent transformation subsequence (FTS). Given a set of graph sequences, GTRACE enumerates a set of all FTSs from the set according to the anti-monotonic property of the support. GTRACE-RS (Inokuchi et al., 2012) which is an extended version of GTRACE first mines FTSs each of which consists of a TR. It then mines FTSs by recursively adding one TR to the mined FTS.

4 Mining Rules for Rewriting States

As mentioned in Section 2, if the parser shown in Fig. 2 adds an incorrect dependency between words once, it never reaches the correct final state. Even if the parser keeps track of multiple candidate outputs using the beam search principle, sometimes all the multiple candidates reach incorrect final states. In this study, we set out to discover rules for rewriting states reaching incorrect final states to those reaching the correct final states from a corpus $D = \{(x, g)\}$ consisting of sentences $x = \langle w_0, w_1, \cdots, w_n \rangle$ and their dependency graphs g. The rewriting rules state that "if state c_g contains graph p as a subgraph, the state is transformed to another state using a certain TR". The rewriting rules contain knowledge about why the dependency parser outputs the incorrect final states and what should be done to fix these incorrect states. Since p and c_g can also be represented as TRs, each rewriting rule is represented as a sequences of TRs.

To mine these rewriting rules, it needs to be determined how to generate the input graph sequences for GTRACE from transition sequences of the dependency analysis. For the sake of simplicity, we assume that the beam width of the parser is 1. Let $C_{1,m}(x) = \langle c_g^{(1)}, c_g^{(2)}, \cdots, c_g^{(m)} \rangle$ be a transition sequence from the initial state, $c_g^{(1)}$, to the correct final state, $c_g^{(m)} = g$, for sentence $(x, g) \in D$. In addition, let $C'_{1,m'}(x) = \langle c_g^{(1)}, \cdots, c_g^{(k-1)}, c_g'^{(k)}, c_g'^{(k+1)}, \cdots, c_g'^{(m')} \rangle$ be another transition sequence for x, where the parser selects the incorrect transition between $c_g^{(k-1)}$ and $c_g'^{(k)}$ for some $k > 1$. One way of generating a graph sequence from transition sequence $C'_{1,m'}(x)$ is to append the correct final state g to the transition sequence after the parser outputs the incorrect final state; i.e., $d_A = \langle c_g^{(1)}, \cdots, c_g^{(k-1)}, c_g'^{(k)}, c_g'^{(k+1)}, \cdots, c_g'^{(m')}, g \rangle$.

We define rewriting rules R to be mined from the graph sequences in the form of d_A as FTSs, each of which contains a subsequence of $seq(C'_{1,m'}(x))$ and TRs to transform $c_g'^{(m')}$ to g; i.e.,

$$R = \{s_1 \Diamond s_2 \mid s_1 \sqsubseteq seq(C'_{1,m'}(x)),\ s_2 \sqsubseteq seq(\langle c_g'^{(m')}, g\rangle),\ (x,g) \in D,\ sup(s_1 \Diamond s_2) \geq sup'\}, \quad (1)$$

where sup' and $s_1 \Diamond s_2$ are the minimum support threshold and the concatenation of transformation sequences s_1 and s_2, respectively. We refer to s_1 of r in R as the precondition of rewriting rule r. Another way of generating a graph sequence from the transition sequence is to append the correct state $c_g^{(k)}$ to its transition subsequence immediately after the parser selects the incorrect transition; i.e., $d_B = \langle c_g^{(1)}, \cdots, c_g^{(k-1)}, c_g'^{(k)}, c_g^{(k)}\rangle$. Then, similar to the first approach, rewriting rules are mined from the graph sequences in the form of d_B. Since $f \sqsubseteq seq(d_B) \Rightarrow f \sqsubseteq seq(d_A)$, all FTSs mined from graph sequences generated in the second approach are also mined from those in the first approach. However, the converse does not hold, since d_A contains information about vertices and edges that is not included in d_B. Since this information is not used in feature vectors characterizing $c_g^{(k-1)}$ to select the next transition at state $c_g^{(k-1)}$, this may explain why incorrect dependency graphs are returned by transition-based dependency parsers. In addition, the first approach may contain "maximal" information gathered from the whole dependency graph that is not available during transitions of the conventional parser (Attardi and Ciaramita, 2008). Therefore, we use the first approach to generate graph sequences. In addition, to distinguish TRs in s_1 of Eq. (1) from those in s_2, we assign label l_2 to edges in g that are not in $c_g'^{(m')}$, and label l_1 to all other edges.

Example 3 *Figure 8 shows a graph sequence generated by appending the correct final state g to the transition sequence for the sentence in Example 1, where the parser selects an incorrect transition from $c_g^{(5)}$ to $c_g'^{(6)}$. Since edge $(2,4)$ is not in $c_g'^{(6)}$ but is in g, label l_2 is assigned to the edge. The transformation sequence of the graph sequence is given as* $\langle vi^{(0,1)}_{[0,\$]} vi^{(0,2)}_{[0,root]} vi^{(0,3)}_{[1,I]} vi^{(0,4)}_{[1,PRP]} vi^{(1,1)}_{[2,saw]} vi^{(1,2)}_{[2,VBD]} ei^{(2,1)}_{[(2,1),l_1]} vr^{(2,2)}_{[1,reduced]} ei^{(3,1)}_{[(0,2),l_1]} vi^{(3,2)}_{[3,him]} vi^{(3,3)}_{[3,PRP]} ei^{(4,1)}_{[(2,3),l_1]} vi^{(4,2)}_{[4,.]} vi^{(4,3)}_{[4,'.']} ed^{(5,1)}_{[(2,3),\bullet]} ei^{(5,2)}_{[(2,4),l_2]}\rangle$*, where ROOT, PRP, VBD, and '.' are the parts of speech (POSs) of the corresponding words*[1].

Let $r = \langle vi^{(0,1)}_{[0,root]} vi^{(1,1)}_{[2,VBD]} ei^{(3,1)}_{[(0,2),l_1]} ei^{(4,1)}_{[(2,3),l_1]} vi^{(4,2)}_{[4,.]} ei^{(5,1)}_{[(2,4),l_2]}\rangle$ be a rewriting rule. If the parser has the rewriting rule r and is in state $c_g'^{(6)}$ in Example 3, the method proposed in this paper deletes edge $(3,4)$ from $c_g'^{(6)}$, and adds edge $(2,4)$ to $c_g'^{(6)}$, by applying r to transit to another state g in Fig. 4 that can reach the correct final state, since the transformation sequence of transition sequence $\langle c_g^{(1)}, \cdots, c_g^{(5)}, c_g'^{(6)}\rangle$ contains the precondition of r as a subsequence. Therefore, the

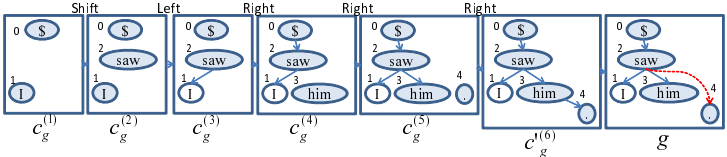


Figure 8: Graph sequence for a transition sequence

[1] We have *a priori* knowledge that each vertex in a state has at most one parent. Therefore, the fact that a TR t for inserting an edge labeled l_2 exists in transformation sequence s indicates that another TR for deleting an edge whose dependent is identical to t must exist in s. For this reason, in our implementation, we do not include TRs for deleting edges in s to reduce the computation time of GTRACE, which increases exponentially with the average length of the transformation sequences in its input.

rewriting rule that transforms from $c_g'^{(6)}$ to g corresponds to a bypass between states in the search space.

GTRACE mines a vast set of FTSs, R, in Eq. (1) from the graph sequences. Next, we discuss how to select certain FTSs from these. Desirable rewriting rules are those that rewrite states reaching incorrect final states to those reaching the correct final states. By rewriting states, the attachment score for a parser using rewriting rules should be better than that for a parser without rewriting rules. This is achieved by selecting the rewriting rule satisfying the following equation.

$$r = \arg\max_{r \in R} \left[\frac{1}{|D|} \sum_{x \in D} \#p(S_r, x) - \#p(S, x) \right], \tag{2}$$

where S_r and S are transition-based parsers with and without rewriting rule r, and $\#p$ is the number of words with correct parents in the dependency graph returned by the parser for sentence x. If $r = s_1 \Diamond s_2$ is a rule in R as given in Eq. (1), $|D| \times \sup(s_1 \Diamond s_2)$ and $|D| \times \big(\sup(s_1) - \sup(s_1 \Diamond s_2)\big)$ are the expected numbers of sentences in D, correctly and incorrectly rewritten by r, respectively. Thus, we obtain the following approximation.

$$\frac{1}{|D|} \sum_{x \in D} \#p(S_r, x) - \#p(S, x) \simeq \sup(s_1 \Diamond s_2) - \big(\sup(s_1) - \sup(s_1 \Diamond s_2)\big) = 2\sup(s_1 \Diamond s_2) - \sup(s_1). \tag{3}$$

We select certain FTSs that maximize the right-hand side of Eq. (3) from R. In addition, we limit the mined rewriting rules such that the number of TRs in s_2 is 1, which is denoted as $|s_2| = 1$, for the following reason. If there is a rewriting rule $r = s_1 \Diamond s_2$ that correctly rewrites a state c into c', where $s_2 = \langle tr_1 tr_2 \rangle$ consists of two TRs, we divide r into $r_1 = s_1 \Diamond tr_1$ and $r_2 = s_1' \Diamond tr_2$, where $s_1' = s_1 \Diamond tr_1$. If r_1 rewrites a state c into another state c'', the state c'' is rewritten into c' by r_2. Even if we limit the mined rewriting rules such that $|s_2| = 1$, we can mine r_1 and r_2, and the state c can be rewritten correctly by r_1 and r_2. In addition, by the limitation, we efficiently mine all rewriting rules because mining rewriting rules is a combinatorial problem of TRs.

We propose a method for mining rewriting rules from transition sequences traversed by a dependency parser. For the sake of simplicity of explanation, we first explain the basic algorithm for mining rewriting rules from transition sequences generated by a transition-based parser with beam width 1, and then expand it using a dependency parser incorporating the beam search principle. The left part of Fig. 9 gives the pseudo-code for mining the set of rewriting rules R from the transition sequences. Let D be a corpus $D = \{(x, g)\}$ consisting of sentences $x = \langle w_0, w_1, \cdots, w_n \rangle$ and their dependency graphs g. In line 6a, ParseWithRules returns a transition sequence d by parsing sentence x using the rewriting rules R. Next, in line 7a, after appending g to the tail of d, denoted by $\langle d \Diamond g \rangle$, $\langle d \Diamond g \rangle$ is added to DB. Subsequently, in line 8a, the attachment score is updated after comparing the final state $c_g^{(m)}$ with the correct dependency g of sentence x. In line 9a, if the attachment score for $R \cup \{r\}$ is not greater than that for R, R is returned. Otherwise, r is added to R. In line 13a, a rewriting rule r satisfying Eqs. (2) and (3) is mined from the FTSs enumerated by GTRACE from DB under the minimum support threshold sup'.

The right part of Fig. 9 gives the pseudo-code for parsing sentence x using the rewriting rules R to return a transition sequence for x. The procedures, except for those in lines b6 to b13, are similar to those in Fig. 2. In line b5, the last state in the transition sequence d is substituted

RuleMiner(D, sup')

1a) $R \leftarrow \emptyset$
2a) $r \leftarrow null$
3a) while
4a) $DB \leftarrow \emptyset$
5a) for sentence $(x = \langle w_0, \cdots, w_n \rangle, g) \in D$
6a) $d \leftarrow$ ParseWithRules $(x, R \cup \{r\})$, where $d = \langle c_g^{(1)}, \cdots, c_g^{(m)} \rangle$
7a) $DB \leftarrow DB \cup \{\langle d \Diamond g \rangle\}$
8a) $evaluate(c_g^{(m)}, g)$
9a) if $R \neq \emptyset$ and the attachment score is saturated,
10a) return R
11a) if $r \neq null$
12a) $R \leftarrow R \cup \{r\}$
13a) $r \leftarrow MineRewritingRule(DB, sup')$
14a) if $r = null$
15a) return R

ParseWithRules$(x = \langle w_0, \cdots, w_n \rangle, R)$

1b) $C_{andidates} \leftarrow \{\langle c_s(x) \rangle\}$
2b) while $\{last(d) \mid d \in C_{andidates}\} \not\subseteq C_F$
3b) $A_{genda} \leftarrow \emptyset$
4b) for each $d \in C_{andidates}$
5b) $c_g = (N, A, N') \leftarrow last(d)$
6b) for each $r = s_1 \Diamond s_2 \in R$
7b) $(a, b) \leftarrow (v_1, v_2)\ s.t.\ s_2 = ei^{(j)}_{[(v_1, v_2), l_2]}$
8b) if $s_1 \sqsubseteq seq(d)$, where $\phi : ID(s_1) \rightarrow ID(seq(d))$
9b) $(i, j) \leftarrow (\phi(a), \phi(b))$
10b) $c_g \leftarrow (N, A \cup \{(i, j), N'\})$
11b) if $\exists i'$ s.t. $(i', j) \in A \wedge i' \neq i$
12b) $c_g \leftarrow (N, A \setminus \{(i', j), N'\})$
13b) $d \leftarrow \langle d \Diamond c_g \rangle$
14b) $A_{genda} \leftarrow A_{genda} \cup \{\langle d \Diamond t(c_g) \rangle \mid t \in \{\text{Left, Right, Reduce, Shift}\}\}$
15b) $C_{andidates} \leftarrow best(A_{genda})$
16b) return $C_{andidates}$

Figure 9: Algorithms for mining rewriting rules and parsing using the rules (beam width is 1)

into c_g by function $last$. All possible transition sequences for c_g are generated in line 14b, and the best of these is selected by function $best$ in line 14b. If there is a rewriting rule whose precondition is contained in $seq(d)$ and its mapping ϕ from vertex IDs in the precondition of r to vertex IDs in $seq(d)$, state $c_g = (N, A, N')$ is rewritten in line 10b or 12b and the parser transits to another state. In line 10b, edge (i, j) corresponding to (a, b), is added to A. In addition, if the j-th word has another parent i' different from i, edge (i', j) is deleted from A in line 12b.

In the case of a parser using a beam search with width b, ParseWithRules returns a set of transition sequences $=\{d_i \mid i = 1, \ldots, b\}$, instead of a single transition sequence. We assume that d_1 is the transition sequence whose final state has the best score of all $\{d_i\}$. If the final state of d_1 is isomorphic to g, we do not append g to d_i for any i, because we do not need any rewriting rules to transform states in transition sequences for sentences for which the parser returns correct final states. Otherwise, after appending g to the tail of each d_i, denoted by $\langle d_i \Diamond g \rangle$, $\langle d_i \Diamond g \rangle$ is added to DB similarly to the case in line a7. Therefore, DB consists of $|D| \times b$ graph sequences. On the other hand, the code for ParseWithRules incorporating the beam search principle is almost the same as the original. In line 15b, the b best transition sequences are selected from A_{genda} by function $best$.

5 Experiments

We evaluated the proposed method using English Penn Treebank data. We used Yamada's head rules to convert the phrase structure to a dependency structure (Yamada and Matsumoto, 2003). We also used the averaged perceptron algorithm with early-update strategy (Collins and Roark, 2004), where weights are updated whenever the gold-standard action-sequence falls off the beam, while the rest of the sequence is ignored. The idea behind this strategy is that later mistakes are often caused by earlier ones, and are irrelevant if the parser is already on the wrong track (Huang and Sagae, 2010).

We split the Wall Street Journal part of the corpus into sections 02-11 for training, sections

12-21 for mining rewriting rules, and section 22 for development. The set of feature templates in Zhang and Clark (2008) characterizing states in the parser was used. Figures 10, 13, and 14 show three of the few dozen rewriting rules mined using the proposed method under a minimum support threshold of 0.05% and beam width of 4, where h, i, j, k, and l are word IDs satisfying $h < i < j < k < l$, and the terms in each circle are words or their POS-tags. The minimum support threshold was set through trial and error using the development data. The supports $sup(s_1)$ and $sup(s_1 \Diamond s_2)$ of the rule shown in Fig. 10 are 1.22% and 0.95%, respectively. In addition, Fig. 11 shows some of the sentences in the corpus whose transition sequences are correctly rewritten by the rule. In Fig. 11, the annotations i, j, and h after words correspond to the respective word IDs in Fig. 10. Here, we explain the rule using concrete examples. The two sentences

- "$ Dozens of workers were injured." and
- "$ Dozens of workers were injured, authorities said."

and their correct dependency graphs, are shown on the left and right sides of Fig. 12, respectively. If the parser is in state $c_g^{(6)}$, where $\sigma = [0]$ and $\beta = [4, 5, \cdots, n]$, when parsing the first sentence, it usually selects Right to transit to the next state. Similarly, the parser incorrectly selects Right, instead of Shift, to transit to the next state from $c_g^{(6)}$ when parsing the second sentence. This is because when the parser is in state $c_g^{(6)}$, it does not know whether the phrase "authorities said." occurs at the end of the sentence. As mentioned in Section 2, if the parser shown in Fig. 2 adds an incorrect dependency between words once, it never reaches the correct final state. This rule rewrites the fifth state $g^{(5)}$ by deleting edge (i, j) and adding edge (h, j) without backtracking. Since we limit the mined rewriting rules to $r = s_1 \Diamond s_2$ where $|s_2| = 1$, this rule does not include deleting edge (h, i) and adding edge (j, i) in $g^{(5)}$ in the rewrite. However, we obtained another rewriting rule to do this.

The rewriting rule shown in Fig. 13 suggests that our parser incorrectly parses sentences containing "because of NN". The supports, $sup(s_1)$ and $sup(s_1 \Diamond s_2)$, of the rule are 0.71% and 0.54%, respectively. We assume that the parser is in state $(\sigma, \beta, A) = ([\cdots, h, i], [j, \cdots], A)$, where w_h ="because", w_i ="of", the POS-tag for w_j is NN (noun, singular or mass), and $(h, i) \in A$. Since the parser using the feature templates in Zhang and Clark (2008) does not know what the word for the parent of the stack top is, the parser incorrectly selects Right, instead of Reduce, to transit to the next state. This rule rewrites state $g^{(5)}$ by deleting edge (i, j) and adding edge (h, j), after the parser adds an incorrect edge.

The supports, $sup(s_1)$ and $sup(s_1 \Diamond s_2)$, of the rule shown in Fig. 14 are 0.053%. After mining the rewriting rule, we investigated words corresponding to vertices h, i, and j by scanning the corpus. The words, (w_h, w_i, w_j), were either (Procter, &, Gamble), (Peabody, &, Co.), (Ogilvy, &, Mather), (Young, &, Rubicam), (Shea, &, Gould), (Standard, &, Poor), (Bausch, &, Lomb), or (Dun, &, Bradstreet). The trigram of words comprising each triplet is a compound proper noun. The rule suggests incorrect parsing of words consisting of proper compound nouns. Since the parser does not know that the triplets are proper compound nouns, it cannot

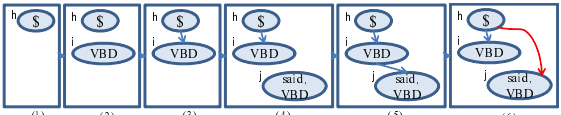


Figure 10: Mined rule #1

$(h) Dozens of workers were(i) injured, authorities said(j).
$(h) But the Fed move was(i) a small gesture, traders said(j).
$(h) Mr. Agnos declined(i) the invitations, the White House said(j).
$(h) They continued(i) to represent that to the board," said(j) Mr. Lloyd.
$(h) Some laggard food issues attracted(i) bargain-hunters, traders said(j).

Figure 11: Sentences for which the rule in Fig. 10 correctly rewrites transition sequences

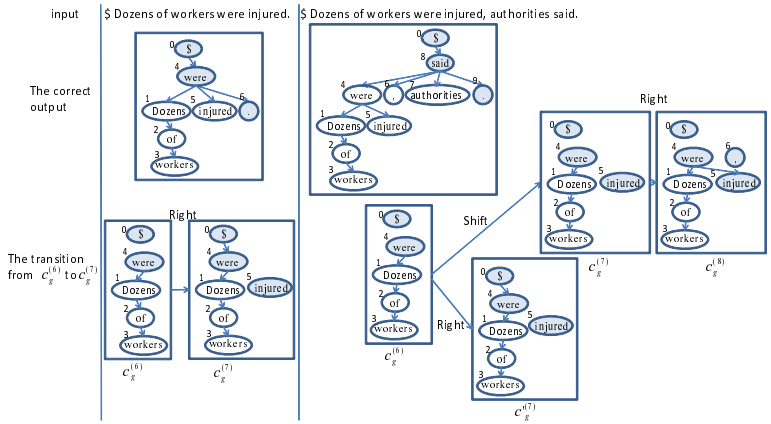


Figure 12: Transition of two similar sentences

correctly parse sentences containing "A & B's NN", where "A & B" is a proper compound noun. This rule rewrites state $g^{(9)}$ by deleting edge (l, j) and adding edge (i, j).

As shown above, the proposed method has the benefits that the rules mined by the method are human-readable and easily understandable. In addition, the rewriting rules contain context that is more complex and detailed than a set of features of the conventional parser, because of the use of the graph representation. Furthermore, if the mined rules are valid grammatically, and a dependency structure obtained by the proposed method, after being rewritten by the rules, is different from a dependency structure in the corpus made by humans, the latter dependency structure may contain incorrect dependencies. The proposed method is therefore also useful for rectifying human errors in the corpus.

According to the knowledge obtained from the mined rewriting rules, we designed three new types of feature templates characterizing states in the parser. First, according to the knowledge obtained from the first rewriting rule, we added feature templates N0w∘Flag and N0t∘Flag, where N0w and N0t denote the word at the top of the stack and its POS-tag in the parser, respectively, Flag is either true or false, and N0w∘Flag and N0t∘Flag are their concatenations. Flag is true, if one of the words "said", "say", or "says" appears after N0w in the sentence. Flag can be calculated in time linear to the length of the sentence during preprocessing. Since features of these templates provide the parser with information on whether a sentence contains "said", "say" or "says" at the end, the parser has a high probability of correctly parsing such sentences. Second, according to knowledge obtained from the second rewriting rule, we added feature templates SPTw∘STw∘N0t and SPTw∘STt∘N0t, where STw, STt, and STPw are the word at the top of the stack, its POS tag, and the word at the parent of the stack top, respectively. The

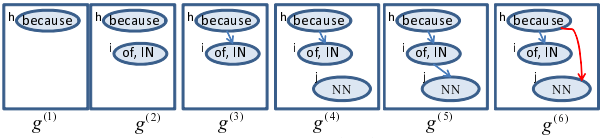


Figure 13: Mined rule #2

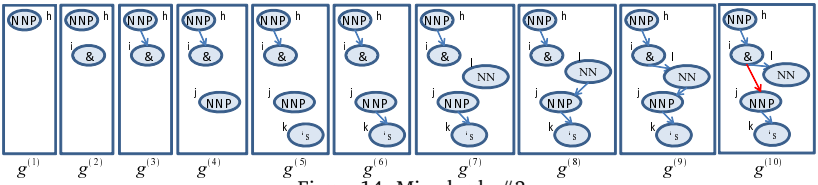


Figure 14: Mined rule #3

arc-eager shift -reduce parser using these feature templates has word information on the parent of the stack top, and is expected to parse sentences containing "because of NN" correctly. Third, according to knowledge obtained from the third rewriting rule, we added feature templates N-2w∘N-1w∘N0w, N-1w∘N0w∘N1w, and N0w∘N1w∘N2w, where N0w is the word at the buffer top, and N-2w, N-1w, N1w, and N2w are words before and after that word. Therefore, N-2w∘N-1w∘N0w, N-1w∘N0w∘N1w, and N0w∘N1w∘N2w are trigrams containing N0w in a sentence. The arc-eager shift-reduce parser using these feature templates is expected to parse sentences containing proper compound nouns correctly.

Figure 15 shows (1) the attachment scores for an arc-eager shift-reduce parser using only the conventional feature templates in Zhang and Clark (2008), (2) that using the feature templates given above as well as the conventional feature templates, and (3) that using the feature templates given above as well as the conventional feature templates and mined rewriting rules[2] for various beam widths. In these experiments, we split the WSJ part of the corpus into sections 02-21 for training, section 22 for development, and section 23 for testing. The figure shows that attachment scores are improved by adding the new feature templates to the conventional feature templates. In particular, the degree of improvement is large when the beam width is small. Although we used knowledge obtained from the three rewriting rules, greater improvement is expected by mining more rewriting rules and designing more feature templates based on the knowledge obtained from these rules. Table 2 reproduces the attachment scores for various dependency parsers given in Hayashi et al. (2012) including those of our proposed method. The table shows that our method is comparable to parsers of the latest studies with respect to the attachment score.

6 Conclusion

This paper proposed a method for mining rewriting rules from transition sequences of an arc-eager dependency parser incorporating the beam search principle for English sentences. The rewriting rules mined by the proposed method are human-readable, and it is possible for us to design new parsers and to generate feature templates for the machine learner of the parser to avoid producing incorrect dependency graphs. In this study, we used GTRACE to analyze transition sequences, although there are other data structures for representing graph sequences,

[2]Result using rewriting rules with beam with 64 was not obtainable due to memory overflow.

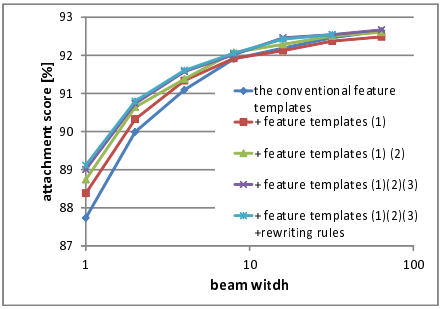


Figure 15: Attachment scores of arc-eager shift-reduce parsers using various feature templates

Table 2: Attachment scores for various methods

method	attachment score [%]
McDonald (McDonald and Pereira, 2006)	91.5
Koo (Koo and Collins, 2010)	93.04
Hayashi (Hayashi et al., 2011)	92.89
Goldberg (Goldberg and Elhadad, 2010)	89.7
Kitagawa (Kitagawa and Tanaka-Ishii, 2010)	91.3
Zhang (Sh beam 64) (Zhang and Clark, 2008)	91.4
Zhang (Sh+Graph beam 64) (Zhang and Clark, 2008)	92.1
Huang (beam+DP) (Huang and Sagae, 2010)	92.1
Zhang (beam 64) (Zhang and Nivre, 2011)	93.07
Hayashi (beam 32+pred 5+DP) (Hayashi et al., 2012)	92.5
Hayashi (beam 32+pred 5+DP+FIRST) (Hayashi et al., 2012)	92.6
Our method (Sh beam 64+additional features)	92.67

such as dynamic graphs (Borgwardt et al., 2006) and evolving graphs (Berlingerio et al., 2009), and algorithms for mining the graphs. Since insertions of vertices cannot be represented by dynamic graphs, and a vertex in an evolving graph always comes with an edge connected to the vertex, these data structures cannot be used to analyze transition sequences in transition-based parsers to mine rewriting rules. Compared with dynamic graphs and evolving graphs, the class of graph sequences is, therefore, general enough to apply to the analysis of transition sequences.

Revision rules proposed in Ahmed and Karypis (2011) transform only edges in dependency structures output by the dependency parser in post-processing. Since it is assumed that revision rules do not remove or add any vertices in the dependency structures, the revision rules cannot be applied to phrase structure analysis. On the other hand, rewriting rules transform states in the transition-based dependency parser. Since the method proposed in this paper can basically be applied to any transition systems whose internal states are represented by graphs, it can be applied to the phrase structure analysis. In addition, rewriting rules are more human-readable than revision rules. Kudo et al. (2005) proposed a method for extracting features represented by trees that are human-readable to obtain high attachment scores. Using the features, the transition-based parser selects a correct transition in each state. Compared with that method, using our method proposed in this paper, we obtain knowledge about why incorrect dependency graphs are returned by transition-based dependency parsers and knowledge about how we transform incorrect states into correct states.

References

Agrawal, R., and Srikant, R. 1994. Fast Algorithms for Mining Association Rules in Large Databases. *Proc. of Int'l Conf. on Very Large Data Bases (VLDB)*, 487–499.

Ahmed, R., and Karypis, G. 2011. Algorithms for Mining the Evolution of Conserved Relational States in Dynamic Networks. *Proc. of IEEE Int'l Conf. on Data Mining (ICDM)*, 1–10.

Attardi, G. and Ciaramita, M. 2008. Tree Revision Learning for Dependency Parsing. *Proc. of Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics, (HLT-NAACL)*, 388–395.

Berlingerio, M., et al., 2009. Mining Graph Evolution Rules. *Proc. of Euro. Conf. on Principles and Practice of Knowledge Discovery in Databases (PKDD)*, 115–130.

Borgwardt, K. M., et al., 2006. Pattern Mining in Frequent Dynamic Subgraphs. *Proc. of IEEE Int'l Conf. on Data Mining (ICDM)*, 818–822.

Collins, M., and Roark, B. 2004. Incremental parsing with the perceptron algorithm. *Proc. of the 42nd Annual Meeting of the Association for Computational Linguistics(ACL)*, pp. 111-118.

Culotta, A., and Sorensen, J. 2004. Dependency Tree Kernels for Relation Extraction. *Proc. of Annual Meeting of Association for Comp. Linguistics (ACL)*, 423–429.

Ding, Y., and Palmer, M. 2004. Automatic Learning of Parallel Dependency Treelet Pairs. *Proc. of Int'l Joint Conf. on Natural Language Processing*, 233–243.

Garey, M., and Johnson, D. 1979. Computers and Intractability: A Guide to the Theory of NP-Completeness. W. H. Freeman and Company, New York.

Goldberg, Y. and Elhadad, M. 2010. An efficient algorithm for easy-first non-directional dependency parsing. *Proc of Human Language Technologies: Conference of the North American Chapter of the Association of Computational Linguistics* (HLT-NAACL), pp. 742–750.

Hayashi, K. et al., 2011. The third-order variational reranking on packed-shared dependency forests. *Proc. of Conference on Empirical Methods in Natural Language Processing* (EMNLP), pp. 1479–1488.

Hayashi, K., et al., 2012. Head-driven Transition-based Parsing with Top-down Prediction. *Proc. of the Annual Meeting of the Association for Computational Linguistics(ACL)*.

Huang, L., and Sagae, K. 2010. Dynamic Programming for Linear-Time Incremental Parsing. *Proc. of Annual Meeting of the Association for Computational Linguistics (ACL)*, 1077-1086.

Inokuchi A., and Washio, T. 2008. A Fast Method to Mine Frequent Subsequences from Graph Sequence Data. *Proc. of IEEE Int'l Conf. on Data Mining (ICDM)*, 303–312.

Inokuchi, A., et al., 2012. Efficient Graph Sequence Mining Using Reverse Search. *IEICE Transactions* 95-D(7), 1947–1958

Inokuchi, A., et al., 2012. Mining Rules for Rewriting States in a Transition-Based Dependency Parser. *Proc of Pacific Rim International Conference on Artificial Intelligence* (PRICAI), 133–145.

Iwatate, M., et al., 2008. Japanese Dependency Parsing Using a Tournament Model. *Proc. of Int'l Conf. on Comp. Linguistics (COLING)*, 361–368.

Kitagawa, K. and Tanaka-Ishii, K. 2010. Tree-based deterministic dependency parsing - an application to Nivre's method. *Proc. of Annual Meeting of the Association for Computational Linguistics* (ACL), Short Papers, pp. 189–193.

Koo, T. and Collins, M. 2010. Efficient third-order dependency parsers. *Proc. of Annual Meeting of the Association for Computational Linguistics* (ACL), pp. 1–11.

Kubler, S., et al., 2009.*Dependency Parsing*. Morgan and Claypool Publishers.

Kudo, T., and Matsumoto, Y. 2002. Japanese Dependency Analysis using Cascaded Chunking. *Proc. of Conf. on Comp. Natural Language Learning (CoNLL)*, 63–69.

Kudo, T. et al. 2005. Boosting-based Parse Reranking with Subtree Features. *Proc. of Annual Meeting of the Association for Computational Linguistics (ACL)*.

Makinen, E. 1989. On the Subtree Isomorphism Problem for Ordered Trees. *Information Processing Letters* 32(5), 271–273.

McDonald, R. and Pereira, F. 2006. Online learning of approximate dependency parsing algorithms. *Proc. of Conference of European Chapter of the Association for Computational Linguistics* (EACL), pp. 81–88.

Nivre, J. 2008. Algorithms for Deterministic Incremental Dependency Parsing. *Comp. Linguistics* 34(4), 513–553.

Pei, J., et al., 2001. PrefixSpan: Mining Sequential Patterns by Prefix-Projected Growth. *Proc. of Int'l Conf. on Data Engineering (ICDE)*, 2–6.

Yamada, H., and Matsumoto, M. 2003. Statistical dependency analysis with support vector machines. *Proc. of the International Workshop on Parsing Technologies (IWPT)*, pp 195–206.

Zhang, Y., and Clark, S. 2008. A Tale of Two Parsers: Investigating and Combining Graph-based and Transition-based Dependency Parsing. *Proc. of Conf. on Empirical Methods in Natural Language Processing (EMNLP)*, 562-571.

Zhang, Y. and Nivre, J. 2011. Transition-based dependency parsing with rich non-local features. *Proc. of Annual Meeting of the Association for Computational Linguistics* (ACL), pp. 188–193.

Coreference Resolution with ILP-based Weighted Abduction

Naoya Inoue[1] *Ekaterina Ovchinnikova*[2]
Kentaro Inui[1] *Jerry Hobbs*[2]
(1) Tohoku University, 6-6-05 Aoba, Aramaki, Aoba-ku, Sendai, 980-8579, Japan
(2) USC/ISI, 4676 Admiralty Way, Marina del Rey, CA 90292, USA
naoya-i@ecei.tohoku.ac.jp,katya@isi.edu,inui@ecei.tohoku.ac.jp,hobbs@isi.edu

ABSTRACT
This paper presents the first systematic study of the coreference resolution problem in a general inference-based discourse processing framework. Employing the mode of inference called weighted abduction, we propose a novel solution to the overmerging problem inherent to inference-based frameworks. The overmerging problem consists in erroneously assuming distinct entities to be identical. In discourse processing, overmerging causes establishing wrong coreference links. In order to approach this problem, we extend Hobbs et al. (1993)'s weighted abduction by introducing weighted unification and show how to learn the unification weights by applying machine learning techniques. For making large-scale processing and parameter learning in an abductive logic framework feasible, we employ a new efficient implementation of weighted abduction based on Integer Linear Programming. We then propose several linguistically motivated features for blocking incorrect unifications and employ different large-scale world knowledge resources for establishing unification via inference. We provide a large-scale evaluation on the CoNLL-2011 shared task dataset, showing that all features and almost all knowledge components improve the performance of our system.

KEYWORDS: weighted abduction, coreference resolution, Integer Linear Programming.

Proceedings of COLING 2012: Technical Papers, pages 1291–1308,
COLING 2012, Mumbai, December 2012.

1 Introduction

In this paper, we explore coreference resolution in a discourse processing framework based on a mode of inference called *weighted abduction* (Hobbs et al., 1993). This framework is appealing because it is a realization of the observation that we understand new material by linking it with what we already know. It instantiates in natural language understanding the more general principle that we understand our environment by coming up with the best explanation for the observables in the environment. Hobbs et al. (1993) show that the lowest-cost abductive proof provides the solution to a whole range of natural language pragmatics problems, such as word sense disambiguation, anaphora and metonymy resolution, interpretation of noun compounds and prepositional phrases and detection of discourse relations. For examples of the application of weighted abduction to discourse processing see (Charniak and Goldman, 1991; Inoue and Inui, 2011; Ovchinnikova et al., 2011; Ovchinnikova, 2012).

If weighted abduction is applied to discourse processing, coreference links naturally follow as a by-product of constructing best explanations. In weighted abduction, coreference resolution is equal to unification of predications; see Sec. 3.1. Similarly, if deductive model building is applied to discourse interpretation, coreference links result from the model minimality. Both inference approaches are based on the idea that predications having the same predicates describe the same situation and therefore their arguments can be assumed to be equal if no logical contradictions follow. If the necessary knowledge is missing from the knowledge base, both the deductive and the abductive procedures are likely to miss relevant coreference links and establish wrong links (overmerge entities). The *overmerging* problem is a serious obstacle in applying reasoning to discourse processing, because it leads to a large number of incorrect inferences; see (Ovchinnikova, 2012) for examples. There have been attempts to employ semantic similarity for merging predications in a deductive framework (Dellert, 2011) and attempts to use linguistically motivated constraints in order to prohibit incorrect unification in an abductive framework (Ovchinnikova et al., 2011; Ovchinnikova, 2012). However, the issue of overmerging was never systematically studied and the proposed solutions were never evaluated. In this paper, we investigate whether adding linguistically motivated features can help to block incorrect links in an inference-based framework.

A lot of effort in NLP was put into coreference resolution systems ranging from rule-based (Lee et al., 2011, etc.) to machine learning-based resolvers (Soon et al., 2001; Ng and Cardie, 2002; Fernandes et al., 2012, etc.); see (Ng, 2010) for a detailed survey. Coreference resolution may require deep understanding of text, access to world knowledge, and inference ability. For example, (Levesque, 2011) considers twin sentences such as *Ed shouted at Tim because he crashed the car* and *Ed shouted at Tim because he was angry*. In order to resolve coreference in these sentences one requires world knowledge about people shouting when being angry and people shouting at someone who made a mistake, e.g., crashed a car. Surprisingly, most of the contemporary coreference resolution systems including the winners of the CoNLL-2011 and CoNLL-2012 shared tasks (Lee et al., 2011; Fernandes et al., 2012) do not exploit any world knowledge. There exist attempts to resolve coreference based on world knowledge resources such as WordNet hierarchy, Wikipedia, semantic similarity, narravite chains (Ponzetto and Strube, 2006; Ng, 2007; Irwin et al., 2011; Rahman and Ng, 2012). Unfortunately, the corresponding resolvers were either not evaluated in large-scale challenges or did not show convincing performance in the challenges. Thus, the question remains open whether employing world knowledge can improve coreference resolution in large unfiltered corpora. In this paper, we investigate whether adding world knowledge for establishing more coreference links can

improve coreference resolution. In the world knowledge employed, our work is most similar to the study on twin sentences presented in (Rahman and Ng, 2012). However, instead of using world knowledge for generating features in a machine learning framework, we explore inference-based discourse processing. Regarding inference, our method may seem related to the coreference resolution research based on Markov Logic Networks (MLNs) (Poon and Domingos, 2008; Song et al., 2012). However, previous MLN-based work on coreference resolution does not incorporate inference rules based on world knowledge.

The key contributions of our work are the following. First, we propose a novel solution to the overmerging problem in an inference-based framework. We extend (Hobbs et al., 1993)'s weighted abduction in order to accommodate unification weights and show how to learn the weights by applying machine learning techniques. For making large-scale processing and parameter learning in an abductive logic framework feasible, we employ a new efficient implementation of weighted abduction based on the Integer Linear Programming technique (Inoue and Inui, 2011).[1] Second, we propose several linguistically motivated features for blocking incorrect unifications and we employ different large-scale world knowledge resources for establishing unification via inference. Third, we report on a large-scale evaluation showing that all features and knowledge components improve the performance.

The structure of this paper is as follows. In Sec. 2, we introduce weighted abduction and its ILP-based implementation. Section 3 describes our discourse processing pipeline based on weighted abduction and discusses the overmerging problem, our solution to it, and types of knowledge we employ for generation of features and axioms. Section 4 presents the experiments on coreference resolution. The final section concludes the paper.

2 Abductive Inference

2.1 Weighted Abduction

Abduction is inference to the best explanation. Formally, logical abduction is defined as follows:

Given: Background knowledge B, observations O, where both B and O are sets of first-order logical formulas,

Find: A hypothesis H such that $H \cup B \models O, H \cup B \not\models \bot$, where H is a set of first-order logical formulas. We say that p is *hypothesized* if $H \models p$, and that p is *explained* if $(\exists q)\ q \rightarrow p \in B$ and q is hypothesized or explained.

Typically, there exist several hypotheses H explaining O. Each of them is called a *candidate hypothesis*. To rank candidate hypotheses according to plausibility, we use the framework of *weighted abduction* as defined by Hobbs et al. (1993). In this framework, observation O is a conjunction of propositions existentially quantified with the widest possible scope. Each proposition has a positive real-valued cost. We use the notation $P^{\$c}$ to indicate that proposition P has cost c and $cost(P)$ to represent the cost of P.

The background knowledge B is a set of first-order logic formulas of the form $P_1^{w_1} \wedge ... \wedge P_n^{w_n} \rightarrow Q_1 \wedge ... \wedge Q_m$. All variables occurring in the antecedent of such axioms are universally quantified with the widest possible scope. Other variables are existentially quantified within the scope of the universal quantifiers. Propositions in the antecedents are assigned positive real-valued *weights*. We use the notation P^w to indicate that proposition P has weight w.

[1] There has been work on applying ILP to coreference (Finkel and Manning, 2008; Denis and Baldridge, 2009), but with no relationship with logical inference.

The two main inference operations in weighted abduction are backward chaining and unification. *Backward chaining* is the introduction of new assumptions given an observation and background knowledge. For example, given $O = \exists x(q(x)^{\$10})$ and $B = \{\forall x(p(x)^{1.2} \rightarrow q(x))\}$, there are two candidate hypotheses: $H_1 = \exists x(q(x)^{\$10})$ and $H_2 = \exists x(p(x)^{\$12})$. In weighted abduction, a *cost function* f is used to calculate assumption costs. The function takes two arguments: costs of the propositions backchained on and weight of the assumption. Usually, a multiplication function is used, i.e. $f(c, w) = c \cdot w$, where c is the cost of the propositions backchained on and w is weight of the corresponding assumption. For example, if $q(x)$ costs \$10 and w of p is 1.2 in the example above, then assuming p in H_2 costs \$12.

Unification is the merging of propositions with the same predicate name by assuming that their arguments are same. For example, $O = \exists x, y(p(x)^{\$10} \wedge p(y)^{\$20} \wedge q(y)^{\$10})$. There is a candidate hypothesis $H = \exists x, y(p(x = y)^{\$10} \wedge x = y^{\$0} \wedge q(x = y)^{\$10})$, where $p(x)^{\$10}$ and $p(y)^{\$20}$ are merged by assuming $x = y$ (called *variable unification assumption*). Hobbs et al. (1993) assign the smallest cost to the result of the unification (i.e. \$10), and zero cost to the variable unification assumption. This principle often causes incompatible entities to be identified (e.g., a dog and a cat) on the basis of slender evidence, since unification always reduces the cost of hypothesis. In order to address this issue, we propose to assign a cost to the variable unification assumption. We use a weighted feature function to assign the cost, where the appropriate weights are learnable from the dataset (see Sec. 3 for further details).

Both operations (backchaining and unification) can be applied to an observation as many times as possible to generate a possibly infinite set of candidate hypotheses. Henceforth, we denote $\mathscr{H}_O$ to represent a set of all possible candidate hypotheses for O. Weighted abduction defines a *cost* of candidate hypothesis H as $cost(H) = \sum_{h \in H} cost(h)$, where h is an atomic conjunct in H also called an *elemental hypothesis* (e.g., $p(x)$ in the above H). In this framework, minimum-cost explanations are best explanations.

2.2 ILP-based Weighted Abduction

Recently, an implementation of weighted abduction based on Integer Linear Programming (ILP) was developed by Inoue and Inui (2011). In this approach the abductive reasoning problem is formulated as an ILP optimization problem. We adopt this solution since (i) the ILP-based reasoner is significantly more efficient than existing implementations of weighted abduction (Inoue and Inui, 2011), and (ii) its declarative nature makes it is highly extensible (Sec. 3).

Given B and O, the framework first enumerates set P of *potential elemental hypotheses* (atomic assumptions). Then it generates ILP variables and constraints based on this set to represent all possible *candidate hypotheses*. The four main ILP variables are $h_p \in \{0,1\}$, $r_p \in \{0,1\}$, $u_{p,q} \in \{0,1\}$, and $s_{x,y} \in \{0,1\}$, where p, q are potential elemental hypotheses and x, y are first-order logical variables or constants used in P. h_p is used to represent whether p is hypothesized ($h_p = 1$) or not ($h_p = 0$). r_p is used to represent whether p pays its cost ($r_p = 0$) or not (p is explained, $r_p = 1$). The ILP objective function is as follows.

$$\text{min. } cost(H) = \sum_{p \in \{p | p \in P, h_p = 1, r_p = 0\}} cost(p) \tag{1}$$

Thus, the cost of H is the sum of the costs of $p \in P$, such that p is included in the hypothesis ($h_p = 1$) and is *not* explained ($r_p = 0$). That is, the backchaining bottoms out in p.

The space of candidate hypotheses is restricted by several ILP constraints. For example, one of the constraints allows us to set $r_p = 1$ (p does not pay its cost) only if at least one proposition $q \in Q$, where Q is a set of propositions that explain p, is hypothesized ($h_q = 1$). The ILP formulation of this constraint is $r_p \leq \sum_{q \in Q} h_q$.

In order to represent unification of two propositions p and q we introduce variables u and s, such that $u_{p,q} = 1$ if p and q are unified and $u_{p,q} = 0$ otherwise; $s_{x,y} = 1$ if variables x and y are set to be equal and $s_{x,y} = 0$ otherwise. Additional constraints are defined on these variables. For example, $p(x_1, x_2, ..., x_n)$ and $p(y_1, y_2, ..., y_n)$ can be unified ($u_{p(x_1,x_2,...,x_n),p(y_1,y_2,...,y_n)} = 1$) only if their corresponding arguments are assumed to be equal (for all $i \in \{1, 2, ..., n\}$, $s_{x_i,y_i} = 1$). This is captured by the following ILP constraint: $n \cdot u_{p(x_1,x_2,...,x_n),p(y_1,y_2,...,y_n)} \leq \sum_{i=1}^{n} s_{x_i,y_i}$.

Formulation of the ILP constraints corresponding to variable inequality is rather straightforward.[2] For each pair of variables x and y such that $x \neq y \in P$, the following equality is introduced: $s_{x,y} = 0$.

3 Coreference Resolution in ILP-based Abductive Framework

3.1 Abduction for Discourse Processing

Abductive reasoning can be used to recover implicit information from natural language texts. The implicit information includes semantic relations between discourse entities, anaphoric relations, character's intentions, etc; see (Hobbs et al., 1993) for detailed examples.

A logical form (LF) of a text represents observations, which need to be explained by background knowledge. In our discourse processing pipeline, a text is first input to the English parser *Boxer* (Bos, 2008). For each segment, the parse produced by *Boxer* is a first-order fragment of the DRS language used in Discourse Representation Theory (Kamp and Reyle, 1993). An add-on to *Boxer* converts the DRS into a logical form in the style of (Hobbs, 1985).

The LF is a conjunction of propositions, which have generalized eventuality arguments that can be used for showing relationships among the propositions. According to (Hobbs, 1985), any predication in the logical notation has an extra argument, which refers to the "condition" of that predication being true. Thus, in the logical form $John(e_1, j) \wedge run(e_2, j)$ for the sentence *John runs*, e_2 is a running event by John and e_1 is a condition of j being named "John".

In the context of discourse processing, we call a hypothesis explaining a logical form *an interpretation* of this LF. The interpretation of the text is carried out by an abductive system. The system tries to prove the logical form of the text, allowing assumptions where necessary. Where the system is able to prove parts of the LF, it is anchoring it in what is already known from the overall discourse or from a knowledge base. Where assumptions are necessary, it is gaining new information.

Let us illustrate the procedure with an example implying coreference resolution. Suppose we need to interpret the text *John gave Bill a book; he was happy to have it*. A simplified logical form of this sentence is as follows:

$John(e_1, x_1) \wedge give(e_2, x_1, x_2, x_3) \wedge Bill(e_3, x_2) \wedge book(e_4, x_3) \wedge he(e_5, x_4) \wedge have(e_6, x_4, x_5) \wedge it(e_7, x_5)$

Suppose our knowledge base contains the following axioms

[2] See (Inoue and Inui, 2012) for the ILP representation of negated propositions.

(1) $give(e_1, x_1, x_2, x_3) \rightarrow get(e_2, x_2, x_3)$
(2) $get(e_1, x_1, x_2) \rightarrow have(e_2, x_1, x_2)$

Given these axioms, we can backchain on $have(e_6, x_4, x_5)$ to $give(e_0, u, x_5, x_4)$. After unifying this proposition with $give(e_2, x_1, x_2, x_3)$, we can infer the equality $x_2 = x_4$ and $x_3 = x_5$, which corresponds to linking *he* to *Bill* and *it* to *book* in the sentence.

3.2 Weighted Unification

Frequently, the lowest-cost interpretation results from identifying two entities with each other, so that their common properties only need to be proved or assumed once. This feature of the algorithm is called "unification", and is one of the principal methods by which coreference is resolved. A naive approach to coreference in an inference-based framework is to unify propositions having the same predicate names unless it implies logical contradictions (Hobbs et al., 1993; Bos, 2011). However, in situations when knowledge necessary for establishing contradictions is missing, the naive procedure results in *overmerging*. For example, given $O = animal(e_1, x) \wedge animal(e_2, y)$, weighted abduction incorrectly assumes x equals y even when $dog(e_3, x)$ and $cat(e_4, y)$ are observed. For *John runs and Bill runs*, with the observations $O = John(e_1, x) \wedge run(e_2, x) \wedge Bill(e_3, y) \wedge run(e_4, y)$, weighted abduction assumes John and Bill are the same individual just because they are both running. If we had complete knowledge about disjointness (*dog* and *cat* are disjoint, people have unique first names), the overmerging problem might not occur because of logical contradictions. However, it is not plausible to assume that we would have an exhaustive knowledge base.

In this study, we impose costs on variable unification assumptions in order to avoid the overmerging. If the unification weights are introduced, unification does not always reduce the overall cost of the hypothesis anymore, which loosens the assumption that all propositions with the same predicate names are coreferential.

How can we define unification weights? The cost of a variable unification assumption, say $x = y$, depends on the properties of x and y. For example, it depends on lexico-syntactic properties. If *x is different from y* or *x writes y* are observed in a text then its unlikely that x and y refer to the same entity. At the same time, observations like $dog(x) \wedge animal(y)$ serve as an evidence that x and y might be coreferential.

We extend the ILP-based weighted abduction framework developed by (Inoue and Inui, 2011) and use a feature-based linear function $\phi(x, y)$ to determine a cost of $x = y$. Features are based on various types of knowledge (see Sec. 3.3). In order to compute the weight of each feature, we parametrize the feature function by a n-dimensional real-valued weight vector $\mathbf{w} = \{w_1, w_2, ..., w_n\}$, and propose to tune $\mathbf{w}$ in a supervised manner.

In order to exploit the cost of variable unification assumptions in the reasoning process, we extend the ILP-based objective function for weighted abduction equation (1) as follows:

$$\text{min. } cost(H; \mathbf{w}) = \frac{1}{Z} \sum_{p \in \{p | p \in P, h_p = 1, r_p = 0\}} cost(p) + \frac{1}{|T_p|^2} \sum_{x,y \in T_p} \sum_{i}^{n} w_i \cdot f_{x,y,i}, \qquad (2)$$

where Z is a total cost of observations, and T_p is a set of logical atomic terms that appear in the set of potential elemental hypotheses, and $f_{x,y,i} \in \{0, 1\}$ is a newly introduced ILP variable that denotes the value of i-th feature for x, y. We normalize each term so that the size of observations and the number of logical terms does not affect to the strength of each term.

The features can be designed by a user. The value of each feature depends on the presence of certain propositions in the corresponding candidate hypothesis. For example, the value of feature $f_{x,y,i}$ can depend on the presence of $dog(x)$ and $cat(y)$. This dependence is represented by the following ILP constraint: $-m+1 \leq m \cdot f_{x,y,i} - \sum_{j=1}^{m} h_{p_m} \leq 0$, where h is an ILP variable (see Sec. 2.2) and $p_1, p_2, ..., p_m$ are propositions on which $f_{x,y,i}$ depends (i.e. $f_{x,y,i} = 1 \Leftrightarrow h_{p_1} = 1 \wedge h_{p_2} = 1 \wedge ... \wedge h_{p_m} = 1$).

Weight Learning

In order to train weight vector $\mathbf{w}$, we employ the modified version of the Passive-Aggressive (PA) algorithm (Crammer et al., 2006), which is a supervised large-margin online learning algorithm applicable to a wide range of linear classifiers ranging from binary classifiers to structured predictors. The original PA algorithm requires the complete set of gold standard labels to be present in the training set. In our case, however, the training set is annotated just with the coreference links (unification sets), but not with other assumptions supporting unification. For example, $dog(x) \wedge animal(y)$ will be annotated with $x = y$, but not with $dog(y)$. Therefore we modified the original algorithm for learning weights from a partial gold standard.

Algorithm 1 depicts our learning algorithm. Every time we receive a training instance (O, H_t) from a set $\mathbb{D}$ of training instances, where O is an observation and H_t is a set of gold standard variable unification assumptions for O, we first find the lowest-cost hypothesis $\hat{H}$ given the current weight vector (line 3). If variable unification assumptions made in $\hat{H}$ are inconsistent with H_t (e.g., *dog* and *animal* are unified in $\hat{H}$, but not in H_t), we train the weight vector (line 5–7). In order to train the vector, we find the lowest-cost hypothesis $\overline{H}$ among candidate hypotheses that *are consistent with* H_t (line 5). To get $\overline{H}$, we add ILP constraints for all $x = y$ in $H_t(s_{x,y} = 1)$ and for all $x \neq y$ in $H_t(s_{x,y} = 0)$ to the ILP optimization problem.

The new weight vector $\mathbf{w}$ should satisfy the following conditions: (i) $cost(\overline{H}; \mathbf{w})$ is less than $cost(\hat{H}; \mathbf{w})$ by at least a margin $\Delta(\hat{H}, \overline{H})$, and (ii) the difference between current weight vector $\mathbf{w}'$ and new weight vector $\mathbf{w}$ is minimal. In line 6, we calculate how much $\mathbf{w}$ should be corrected, where C is a parameter of the PA algorithm that is the aggressiveness of weight updates. $\phi(\hat{H})$ and $\phi(\overline{H})$ are the sums of feature vectors for variable unification assumptions in $\hat{H}$ and $\overline{H}$ respectively. $\Delta(\hat{H}, \overline{H})$ is a loss function that measures how different $\hat{H}$ and $\overline{H}$ are. The more different $\hat{H}$ and $\overline{H}$ are the larger an ensured margin is. In our experiments, we use the loss function $\Delta_P(\hat{H}, \overline{H}) = W_O / T_O$, where T_O is the total number of pairs of logical atomic terms in the observation and W_O is the total number of variable unification assumptions for observed logical terms in $\hat{H}$ that disagrees with $\overline{H}$. We implemented this training algorithm in a distributed learning framework (McDonald et al., 2010).

3.3 Features

Each feature we use is defined for pairs of unifiable variables (v_1, v_2). The features are summarized in Table 1.

Incompatible properties If two entities have incompatible properties, they are unlikely to be identical. We use WordNet antonymy (*black – white*) and sibling relation (*cat – dog*) to derive incompatible properties. Moreover, we assume that two proper names not belonging to the same WordNet synset are unlikely to refer to the same entity. Correspondingly, we generate three binary features *A*, *S*, and *P* (see Table 1).

Algorithm 1 Passive-Aggressive algorithm for partial gold standard dataset.

for all $i \in \{1, 2, ..., N\}$ **do**
  for all $(O, H_t) \in \mathbb{D}$ **do**
    $\hat{H} \leftarrow \arg\min_{H \in \mathcal{H}_O} cost(H; \mathbf{w})$
    if $\hat{H} \not\models H_t$ **then**
      $\overline{H} \leftarrow \arg\min_{H \in \mathcal{H}_O} cost(H; \mathbf{w})$ s.t. $H \models H_t$
      $\tau \leftarrow \min(C, \frac{cost(\overline{H}; \mathbf{w}) - cost(\hat{H}; \mathbf{w}) + \Delta(\overline{H}, \hat{H})}{||\phi(\hat{H}) - \phi(\overline{H})||^2})$
      $\mathbf{w} \leftarrow \mathbf{w} + \tau(\phi(\overline{H}) - \phi(\hat{H}))$
    end if
  end for
end for

Feature type	Feature
Incompatible properties	$A(v_1, v_2) = \begin{cases} 1 \text{ if } \exists p_1(.., v_1, ..), p_2(.., v_2, ..): \ p_1, p_2 \text{ are WN antonyms;} \\ 0 \text{ otherwise} \end{cases}$ $S(v_1, v_2) = \begin{cases} 1 \text{ if } \exists p_1(.., v_1, ..), p_2(.., v_2, ..): \ p_1, p_2 \text{ are WN siblings;} \\ 0 \text{ otherwise} \end{cases}$ $P(v_1, v_2) = \begin{cases} 1 \text{ if } \exists p_1(e_1, v_1), p_2(e_2, v_2): \ p_1, p_2 \text{ are proper names,} \\ \text{not in the same WN synset;} \\ 0 \text{ otherwise} \end{cases}$
Conditional unification	$CU(v_1, v_2) = \begin{cases} 1 \text{ if } \exists p_1(v_1, x_1, .., x_n), p_2(v_2, y_1, .., y_n): \\ p_1, p_2 \text{ are frequent predicates} \\ \text{and } \forall i \in \{1, .., n\} : s_{x_i, y_i} = 1; \\ 0 \text{ otherwise} \end{cases}$
Argument inequality	$SA(v_1, v_2) = \begin{cases} 1 \text{ if } \exists p(.., v_1, .., v_2, ..); \\ 0 \text{ otherwise} \end{cases}$ $EA(v_1, v_2) = \begin{cases} 1 \text{ if } \exists p(v_1, .., e_1, ..), p(v_2, .., e_2, ..): \ s_{v_1, v_2} \wedge s_{e_1, e_2} = 0; \\ 0 \text{ otherwise} \end{cases}$
Explicit non-identity	$NI(v_1, v_2) = \begin{cases} 1 \text{ if } \exists p(e, v_1, v_2): \ p \text{ is a non-identity predicate;} \\ 0 \text{ otherwise} \end{cases}$
Functional relations	$FR(v_1, v_2) = \begin{cases} 1 \text{ if } \exists p(e_1, v_1, x_1), p(e_2, v_2, x_2): \\ p \text{ is a functional relation predicate} \\ \text{and } x_1 \neq x_2 \text{ and } v_1 = v_2; \\ 0 \text{ otherwise} \end{cases}$
Modality	$M(v_1, v_2) = \begin{cases} 1 \text{ if } \lvert MCPred(v_1) \cap MCPred(v_2) \rvert = \emptyset; \\ 0 \text{ otherwise} \end{cases}$
Common properties	$CP_1(v_1, v_2) = \lvert CPred(v_1, v_2) \rvert,$ $CP_2(v_1, v_2) = \sum_{p \in CPred(v_1, v_2)} Freq(p)$ $CP_3(v_1, v_2) = \sum_{p \in CPred(v_1, v_2)} WNAbst(p)$
Derivational relation	$DR(v_1, v_2) = \begin{cases} 1 \text{ if } \exists p_1(v_1, ..), p_2(v_2, ..): \\ p_1, p_2 \text{ are derivationally related;} \\ 0 \text{ otherwise} \end{cases}$

Table 1: Summary of the feature set.

Conditional unification If two entities have very frequent common properties, these properties usually do not represent a good evidence for the entities to be identical. For example, given *John goes* and *he goes*, it might be incorrect to assume that *John* and *he* are coreferential just

because they are both going. We want to allow unification of frequent predications (e.g., *go*) only if there is other evidence for their arguments to be unified. In order to capture this idea, we introduce binary feature CU and compute its value as follows: If v_1 and v_2 occur as first arguments of propositions $p_1(v_1, x_1, .., x_n), p_2(v_2, y_1, .., y_n)$, such that p_1, p_2 are frequent predicates, and $\forall i \in \{1, .., n\} : s_{x_i, y_i} = 1$ (where s is an ILP variable, see Sec. 2.2) then $CU(v_1, v_2) = 1$; otherwise $CU(v_1, v_2) = 0$.

Argument inequality We use two argument constraints to generate features. First, we assume that arguments of the same proposition usually cannot refer to the same entity. Reflexive verbs represent an exception (e.g., *John cut himself*), but we assume that these cases are resolved by the *Boxer* semantic parser (see Sec. 3.1) and do not require inference. We create binary feature SA and compute its value as follows: If v_1 and v_2 occur as arguments of the same proposition then $SA(v_1, v_2) = 1$; otherwise $SA(v_1, v_2) = 0$.

One more feature we introduce concerns event variables. For example, given the sentences *John said that Mary was reading* and *John said that he was tired* we do not want to unify both *say* propositions, because in each case something different has been said. Predicates like *say* usually have clauses as their arguments. Unifying clauses just because they are arguments of the same predicate is often incorrect. In our framework, a clause is represented by an event variable, i.e. a variable which is a first argument of the head of the clause. We make the following assumption: If two unifiable propositions $p(v_1, .., e_1, ..), p(v_2, .., e_2, ..)$ have event variables as their arguments, then they are unlikely to be unified if the event arguments have not been already unified. We create binary feature EA and compute its value as follows: if (i) there are two unifiable propositions $p(v_1, .., e_1, ..), p(v_2, .., e_2, ..)$ that have event variables e_1, e_2 as non-first arguments, (ii) $e_1 \neq e_2$, and (iii) $v_1 = v_2$, then $EA(v_1, v_2) = 1$; otherwise $EA(v_1, v_2) = 0$.

Explicit non-identity We manually collected a set of 33 predicates indicating explicit non-identity, e.g., *similar to*, *different from*. Presence of these predicates in a logical form indicates that their second and third arguments are unlikely to refer to the same entity. We create binary feature NI and compute its value as follows: If there is $p(e, v_1, v_2)$ and p is a predicate indicating explicit non-identity then $NI(v_1, v_2) = 1$; otherwise $NI(v_1, v_2) = 0$.

Functional relations A binary relation r is functional if $\forall x, y_1, y_2 : r(x, y_1) \wedge r(x, y_2) \rightarrow y_1 = y_2$. For example, a person can be a son of exactly one man. Lin et al. (2010) automatically learn functional relations from a corpus and assign a confidence score to each extracted relation. We use the set of functional relations generated by Lin et al. (2010) in order to generate feature FR. We extract 1,661 functional relations from the dataset. We create a binary feature FR and compute its value as follows: if (i) there are two predicates $p(e_1, v_1, x_1), p(e_2, v_2, x_2)$, where p indicates a functional relation, (ii) $x_1 \neq x_2$, and (iii) $v_1 = v_2$ then $FR(v_1, v_2) = 1$; otherwise $FR(v_1, v_2) = 0$.

Modality We assume that two predications having different modality are unlikely to refer to the same entity. For example, given *John runs* and *he does not/might run*, *John* and *he* are unlikely to be coreferential. Let $MPred(v)$ be a set of predicates that represent the modality of event v. In our experiments, we consider three modality-denoting predicates produced by the *Boxer* semantic parser (nec, pos, not), and verbal predicates (e.g., *think*) as modality-denoting predicates. We create binary feature M and compute its value as follows: if there are two unifiable verbal propositions $p(v_1, ...), p(v_2, ...)$ and $|MPred(v_1) \cap MPred(v_2)| = \emptyset$ then $M(v_1, v_2) = 1$; otherwise $M(v_1, v_2) = 0$.

Common properties We assume that the more properties two entities share the more likely it is that they are identical. For example, given *John was jogging, while Bill was sleeping. He jogs every day*, *John* and *he* are likely to be coreferential, because they are both arguments of *jog*. Let $CPred(v_1, v_2)$ be a set of pairs of predicates p_1, p_2, such that v_1, v_2 occur at the same argument positions of p_1 and p_2 while p_1 are p_2 equal or they occurs in the same WordNet synset. We generate three types of real-valued features: $CP_1(v_1, v_2) = |CPred(v_1, v_2)|$, $CP_2(v_1, v_2) = \sum_{p \in CPred(v_1, v_2)} Freq(p)$, and $CP_3(v_1, v_2) = \sum_{p \in CPred(v_1, v_2)} WNAbst(p)$, where $Freq(p)$ is a word-frequency of p from the Corpus of Contemporary American English[3], and $WNAbst(p)$ is a level of abstraction of p in the WordNet hierarchy (the number of steps to the root).

Derivational relations We use WordNet derivational relations between nouns and verbs in order to link nominalizations and verbs. For example, given *Sales of cars grew. The growth followed year-to-year increases*, *grew* and *growth* are coreferential. We generate binary feature *DR* to capture these links (see Table 1).

3.4 Knowledge for Inference

The abductive reasoning procedure is based on a knowledge base consisting of a set of axioms. In the experiment described in this paper we employed the following background knowledge.

WordNet The dataset we use for evaluation (see Sec. 4) is annotated with WordNet (Fellbaum, 1998) senses. Given this annotation, we mapped word senses to WordNet synsets. Given WordNet relations defined on synsets, we generate axioms of the following form:

Hyperonymy, instantiation: $synset_1(s_1, x) \rightarrow synset_2(s_2, x)$
Causation, entailment: $synset_1(s_1, e_1) \rightarrow synset_2(s_2, e_2)$
Meronymy, membership: $synset_1(s_1, x_1) \rightarrow synset_2(s_2, x_2) \wedge of(x_1, x_2)$

We extracted 22,815 axioms from WordNet.

FrameNet We generated axioms mapping predicates with their arguments into FrameNet (Ruppenhofer et al., 2010) frames and roles. For example, the following axiom maps the verb *give* to the GIVING frame.

$\text{GIVING}(e_1) \wedge \text{DONOR}(e_1, x_1) \wedge \text{RECIPIENT}(e_1, x_2) \wedge \text{THEME}(e_1, x_3) \rightarrow give(e_1, x_1, x_3) \wedge to(e_2, e_1, x_2)$

Weights of these axioms are based on frequencies of lexeme-frame mappings in the annotated corpora provided by the FrameNet project. Moreover, we used FrameNet frame relations to derive axioms. An example of an axiomatized relation is given below.

$\text{GIVING}(e_1) \wedge \text{DONOR}(e_1, x_1) \wedge \text{RECIPIENT}(e_1, x_2) \wedge \text{THEME}(e_1, x_3) \rightarrow$
$\text{GETTING}(e_2) \wedge \text{SOURCE}(e_2, x_1) \wedge \text{RECIPIENT}(e_1, x_2) \wedge \text{THEME}(e_1, x_3)$

In order to generate the FrameNet axioms, we used the previous work on axiomatizing FrameNet by Ovchinnikova (2012). We generated 12,060 axioms from the dataset. In addition, we used a resource assigning possible lexical fillers disambiguated into WordNet synsets to FrameNet roles (Bryl et al., 2012). For example, the role THEME of the GIVING frame is mapped to synsets `object#n#1` and *thing#n#1*. Given this information, the following axiom is generated.

$thing\#n\#1(s, x) \rightarrow \text{GIVING}(e_1) \wedge \text{THEME}(e_1, x)$

[3] http://www.wordfrequency.info/

Weights of these axioms are based on the scores provided by Bryl et al. (2012). We generated 24,571 axioms from the dataset.

Narrative chains Similar to (Rahman and Ng, 2012), we employ narrative chains learned by Chambers and Jurafsky (2009), which were shown to have impact on resolving complex coreference; see (Rahman and Ng, 2012) for details. Narrative chains are partially ordered sets of events centered around a common protagonist that are likely to happen in a sequence. Knowledge about such sequences can facilitate coreference resolution. For example, given *Max fell, because John pushed him* we know that *Max* and *him* are coreferential, because we know that an object of the pushing event can be a subject of the falling event. For example, we generate the following axioms.

$Script\#1(s, e_1, x_1, u) \rightarrow \text{arrest}(e_1, x_1, x_2, x_3) \wedge police(e_2, x_1)$
$Script\#1(s, e_1, x_1, u) \rightarrow \text{charge}(e_1, x_1, x_2, x_3) \wedge police(e_2, x_1)$

Weights of these axioms are based on the scores provided by (Chambers and Jurafsky, 2009). We extract 1,391,540 axioms from the dataset.

3.5 Disambiguation of Named Entities

In the experiment on coreference resolution, we extended *Boxer*'s output with the information inferred by the *AIDA* tool. The *AIDA* tool (Yosef et al., 2011) is a framework for entity detection and disambiguation. Given a natural language text, it maps mentions of ambiguous names onto canonical entities like people or places, registered in a knowledge base like DBpedia (Bizer et al., 2009) or YAGO (Suchanek et al., 2008). For example, mentions *A. Einstein* and *Einstein* will be both mapped to the YAGO node *Albert_Einstein*. An add-on to our pipeline assigns the same variables to each two named entities disambiguated by *AIDA* into the same YAGO node.

4 Evaluation

We evaluate coreference resolution in our weighted abduction framework using the CoNLL-2011 shared task dataset (Pradhan et al., 2011). The CoNLL-2011 dataset was based on the English portion of the OntoNotes 4.0 data (Hovy et al., 2006). OntoNotes is a corpus of large scale annotation of multiple levels of the shallow semantic structure in text. The OntoNotes coreference annotation captures general anaphoric coreference. Note that OntoNotes captures explicit coreference links only, while our procedure also discovers implicit semantic overlap.

The CoNLL-2011 shared task was to automatically identify mentions of entities and events in text and to link the corefering mentions together to form entity/event chains. In our experiment, we do not identify mentions, but only compute precision and recall of the inferred coreference links given the mentions identified in the gold standard annotation.

In the CoNLL-2011 shared task, four metrics were used for evaluating coreference performance: MUC, B^3, CEAF, and BLANC. The evaluation metrics are described in (Pradhan et al., 2011). Each of the metric tries to address the shortcomings of the earlier metrics. MUC is the oldest metric; it has been criticized for not penalizing overmerging (Recasens and Hovy, 2010). Since one of the goals of this study is to reduce overmerging in our inference-based framework, this metric does not seem to be representative for us. The B^3 and CEAF metrics were also considered to produce counterintuitive results (Luo, 2005; Recasens and Hovy, 2010). BLANC, as the most recent evaluation metric, overcomes the drawbacks of MUC, B^3, and CEAF. The definition formula of BLANC given in (Recasens and Hovy, 2010) is replicated in Table 2, where

rc, wc, rn, wn indicate the number of right coreference links, wrong coreference links, right non-coreference links, and wrong non-coreference links correspondingly.

Score	Coreference	Non-coreference	Metric
P	$P_c = \frac{rc}{rc+wc}$	$P_n = \frac{rn}{rn+wn}$	BLANC-P $= \frac{P_c+P_n}{2}$
R	$R_c = \frac{rc}{rc+wn}$	$R_n = \frac{rn}{rn+wc}$	BLANC-R $= \frac{R_c+R_n}{2}$
F	$F_c = \frac{2P_cR_c}{P_c+R_c}$	$F_n = \frac{2P_nR_n}{P_n+R_n}$	BLANC $= \frac{F_c+F_n}{2}$

Table 2: Definition formula for BLANC.

We rely on BLANC when drawing conclusions, but present values of the other three evaluation metrics as well.

4.1 Results and Discussions

We intend to evaluate whether the introduction of linguistically motivated features (Sec. 3.3) and world knowledge (Sec. 3.4) enables us to outperform the naive inference-based approach implying that predications with the same names refer to the same entities. In order to evaluate the impact of each feature and knowledge component separately, we run ablation tests.

Note that for 145 of 6,894 sentences in the test set, no logical forms were produces by the *Boxer* semantic parser. Moreover, in the run employing WordNet-based inference, inference results could not be produced for 101 of 303 test texts because of the computational complexity of reasoning. In order to keep the comparison fair, we use evaluate all features and knowledge components on the same set of 202 texts, for which inference results were produced in all runs.

Table 3 represents the results of the ablation tests. We test the features listed in Table 1 as well as axioms extracted from WordNet (WN), FrameNet (FN), narrative chains (NC) and knowledge provided by AIDA (AI). All features representing incompatible properties are tested together (IP in Table 3). Similarly, all argument inequality features (AI) and common property features (CP) are tested together.

The first row represents results for the run without employing any features and knowledge resources. In the second run, world knowledge is employed without linguistic features. These two runs correspond to the original weighted abduction approach to unification implying unification of all predications having the same predicate names. We see that adding knowledge results in lower values of BLANC. This happens because of the overmerging problem increased by additional coreference links inferred with the help of the employed knowledge resources.

Then we test linguistic features intended to block incorrect unification (IP, CU, AI, NI, FR, M) one by one. Each of the features improves the BLANC values; conditional unification CU has the most significant impact.[4] The common property feature (CP) and the derivational relations feature (M) introduce additional unifications. Therefore we test them together with the best combination of the unification blocking features (IP+CU+AI+NI+FR+M). Both features have a positive impact as compared to the run employing just the unification blocking features. Now we test each world knowledge component using the best combination of features

[4]It is interesting to note that MUC and B^3 evaluation metrics represent completely the opposite picture, which supports the criticism of these metrics for tolerating overmerging (Recasens and Hovy, 2010).

(IP+CU+AI+NI+FR+M+CP). Each knowledge component except for WordNet has a positive impact in terms of BLANC as compared to the run using the best combination of all features.

Features								Inference				MUC			B^3			CEAFE			BLANC		
IP	*CU*	*AI*	*NI*	*FR*	*M*	*CP*	*DR*	WN	FN	NC	AI	R	P	F	R	P	F	R	P	F	R	P	F
												73.7	69.6	71.6	75.5	39.9	52.2	30.7	36.1	33.2	53.0	51.7	39.1
								✓	✓	✓	✓	72.3	68.6	70.4	74.6	41.6	53.4	32.3	37.1	34.5	52.3	51.3	39.9
✓												71.2	68.8	70.0	73.2	41.8	53.2	32.8	35.9	34.3	53.5	51.9	41.0
	✓											33.4	58.2	42.5	42.5	76.2	54.6	59.6	28.6	38.7	55.7	60.9	56.6
		✓										70.1	68.4	69.3	72.3	41.8	53.0	33.1	35.3	34.2	53.0	51.6	41.0
			✓									70.4	68.5	69.4	72.5	41.6	52.9	32.3	34.8	33.5	52.8	51.5	40.5
				✓								70.4	68.5	69.5	72.7	41.7	53.0	32.5	35.0	33.7	52.9	51.6	40.7
					✓							70.3	68.4	69.3	72.6	41.9	53.1	32.8	35.4	34.1	53.3	51.7	41.0
✓	✓	✓	✓	✓	✓	✓						39.4	62.0	48.2	46.6	74.1	57.2	59.6	30.9	40.8	58.4	61.6	59.4
✓	✓	✓	✓	✓	✓		✓					36.6	61.2	45.8	44.6	75.8	56.1	59.8	29.6	39.6	57.5	61.4	58.6
✓	✓	✓	✓	✓	✓	✓	✓	✓				36.2	60.3	45.2	44.5	75.3	56.0	59.7	29.6	39.6	57.4	61.2	58.5
✓	✓	✓	✓	✓	✓	✓	✓		✓			40.7	63.1	49.5	48.5	73.2	58.4	58.6	30.8	40.4	59.5	61.0	60.1
✓	✓	✓	✓	✓	✓	✓	✓			✓		40.1	63.0	49.0	47.4	74.0	57.8	59.1	30.8	40.5	59.0	61.5	59.9
✓	✓	✓	✓	✓	✓	✓	✓				✓	42.5	64.3	51.1	49.1	72.8	58.7	58.6	31.5	41.0	59.7	61.5	60.4
✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	42.9	64.4	51.5	50.5	73.3	59.9	59.4	32.8	42.3	59.9	60.9	60.3

Table 3: Ablation tests of features and world knowledge.

The results of the ablation tests show significant improvement over the naive approach (by more than 20% F-measure), but can we claim that we solved the overmerging problem? We perform one more experiment in order to get a deeper understanding of the performance of our discourse processing pipeline in coreference resolution.

The best performance in the CoNLL-2011 shared task was achieved by the *Stanford NLP* system (Lee et al., 2011) that is a rule-based resolver encoding traditional linguistic constraints on coreference. We replicate the results of *Stanford NLP* as applied to the CoNLL-2011 dataset; see the first row in Table 4. We use the output of *Stanford NLP* only for those texts, which could be processed by our discourse processing pipeline, therefore the recall/precision values for *Stanford NLP* in Table 4 are lower than the original results published in (Lee et al., 2011).

We aim at checking whether enriching the output of the state-of-the-art coreference resolver with additional links inferred by our system using all features and all world knowledge will improve the performance. The evaluation of the "merged" output is presented in the second row of Table 4 (SNLP+WA). Unfortunately, the precision of SNLP+WA is lower than that of SNLP alone. This happens because adding world knowledge results in new coreference links, while the overmerging problem is not completely solved. SNLP discovers 2277 out of 7557 correct coreference links and 40247 out of 41527 correct non-coreference links. In the merged output, there are more correct coreference links (3065), but less correct non-coreference links (36959). Note that *Stanford NLP* performs noun phrase coreference resolution only, while our system is not restricted to noun phrases and can also discover implicit coreference links.

System	MUC			B^3			CEAFE			BLANC		
	R	P	F	R	P	F	R	P	F	R	P	F
SNLP	42.8	74.4	54.3	50.4	85.2	63.4	66.3	32.6	43.7	63.5	76.2	66.7
SNLP+WA	52.0	70.1	59.7	57.3	72.7	64.1	60.5	37.2	46.1	64.8	64.7	64.7

Table 4: Performance of the *Stanford NLP* system (SNLP) compared to performance of our weighted abduction engine enriched with *Stanford NLP* (SNLP+WA) output.

The main cause of overmerging is related incompatible properties. We anticipated the incom-

patible properties to have a more significant impact on precision than they actually had in the ablation tests. But in the current study, we consider only those properties to be incompatible which are expressed syntactically in the same way, e.g., *Japanese goods* vs. *German goods*. However, the same property can be expressed by a wide variety of syntactic constructions, e.g., *goods from Germany*, *goods produced in Germany*, *Germany produced goods* etc. In order to discover deeper contradictions, we have to work on normalization of the representation of properties, e.g., use *origin*:*Germany*:x instead of $German(e, x)$ and $from(e_1, x, y) \wedge Germany(e_2, y)$. FrameNet attempts to achieve such a normalization by using standardized frame and role names. Unfortunately, the limited coverage of the FrameNet resource (Shen and Lapata, 2007; Cao et al., 2008) does not allow us to solve the problem on a large scale.

Analyzing the results, we also found overmergings not implying any explicit contradictions. For example, in the sentence *He sat near him*, both *he* propositions are unlikely be coreferential, but our framework fails to capture that. Such overmergings might be blocked by explicit modeling of discourse salience. In the future, we plan to use existing discourse salience models (e.g., (Lappin and Leass, 1994)) to create real-valued salience features for weighted unification.

One more issue concerns the quality of the obtained interpretations. Our learning framework assumes that we can obtain optimal solutions, but we also exploit suboptimal solutions by imposing a timeout in this experiment. However, it has been reported that exploiting suboptimal solutions sometimes hurts performance (Finley and Joachims, 2008). In the future, we will address this problem using an approximate learning framework (e.g., (Huang et al., 2012)).

Conclusion and perspectives

In this paper, we investigated the overmerging problem in a general inference-based discourse processing pipeline using the mode of inference called weighted abduction. In our framework, resolving coreference is a by-product of constructing best interpretations of text. Coreference links naturally result from unifications of predications during the inference process. The naive approach to unification involves unifying predications with the same predicate names.

This paper presents the first systematic study of the overmerging problem resulting from naive unification. We proposed several linguistically motivated features for blocking incorrect unifications as well as employed different large-scale world knowledge resources for establishing unification via inference. We extended ILP-based weighted abduction in order to accommodate unification weights and showed how to learn the weights in a supervised manner. All features and almost all knowledge components proved to improve the performance of our system tested on a large state-of-the-art test dataset.

We cannot claim that the problem of overmerging has been solved, because we still discover overmerging of explicit anaphora produced by our system as compared to a state-of-the-art rule-based coreference resolver. However, the proposed framework presents a significant improvement over the naive approach (by over 20% of F-measure). Moreover, it is highly extensible for including more features and knowledge sources.

Acknowledgments

We would like to thank Johan Bos for helping us to set up the *Boxer* system for our experiments. We also thank the *Stanford NLP* team and the *AIDA* team for their cooperation. This work was partially supported by Grant-in-Aid for JSPS Fellows (22-9719) and Grant-in-Aid for Scientific Research (23240018).

References

Bizer, C., Lehmann, J., Kobilarov, G., Auer, S., Becker, C., Cyganiak, R., and Hellmann, S. (2009). Dbpedia - a crystallization point for the web of data. *J. Web Sem.*, 7(3):154–165.

Bos, J. (2008). Wide-Coverage Semantic Analysis with Boxer. In Bos, J. and Delmonte, R., editors, *Proceedings of STEP*, Research in Computational Semantics, pages 277–286. College Publications.

Bos, J. (2011). A survey of computational semantics: Representation, inference and knowledge in wide-coverage text understanding. *Language and Linguistics Compass*, 5(6):336–366.

Bryl, V., Tonelli, S., Giuliano, C., and Serafini, L. (2012). A novel framenet-based resource for the semantic web. In *SAC*, pages 360–365.

Cao, D. D., Croce, D., Pennacchiotti, M., and Basili, R. (2008). Combining word sense and usage for modeling frame semantics. In *Proc. of STEP 2008*, pages 85–101.

Chambers, N. and Jurafsky, D. (2009). Unsupervised Learning of Narrative Schemas and their Participants. In *Proceedings of ACL*, pages 602–610.

Charniak, E. and Goldman, R. P. (1991). A Probabilistic Model of Plan Recognition. In *Proceedings of AAAI*, pages 160–165.

Crammer, K., Dekel, O., Keshet, J., and S. Shalev-Shwartz, Y. S. (2006). Online Passive-Aggressive Algorithms. pages 551–585.

Dellert, J. (2011). Challenges of Model Generation for Natural Language Processing. Master's thesis, University of Tübingen.

Denis, P. and Baldridge, J. (2009). Global Joint Models for Coreference Resolution and Named Entity Classification. In *Procesamiento del Lenguaje Natural 42*, pages 87–96, Barcelona: SEPLN.

Fellbaum, C., editor (1998). *WordNet: an electronic lexical database*. MIT Press.

Fernandes, E., dos Santos, C., and Milidiú, R. (2012). Latent structure perceptron with feature induction for unrestricted coreference resolution. In *CoNLL Shared Task 2012*, pages 41–48.

Finkel, J. R. and Manning, C. D. (2008). Enforcing transitivity in coreference resolution. In *Proceedings of ACL*, pages 45–48.

Finley, T. and Joachims, T. (2008). Training structural svms when exact inference is intractable. In *Proceedings of the 25th international conference on Machine learning*, ICML '08, pages 304–311, New York, NY, USA. ACM.

Hobbs, J. R. (1985). Ontological promiscuity. In *Proceedings of ACL*, pages 61–69, Chicago, Illinois.

Hobbs, J. R., Stickel, M., Martin, P., and Edwards, D. (1993). Interpretation as abduction. *Artificial Intelligence*, 63:69–142.

Hovy, E., Marcus, M., Palmer, M., Ramshaw, L., and Weischedel, R. (2006). Ontonotes: The 90% solution. In *Proceedings of HLT-NAACL 2006*, pages 57–60.

Huang, L., Fayong, S., and Guo, Y. (2012). Structured perceptron with inexact search. In *HLT-NAACL*, pages 142–151.

Inoue, N. and Inui, K. (2011). ILP-Based Reasoning for Weighted Abduction. In *Proceedings of AAAI Workshop on Plan, Activity and Intent Recognition*.

Inoue, N. and Inui, K. (2012). Large-scale Cost-based Abduction in Full-fledged First-order Predicate Logic with Cutting Plane Inference.

Irwin, J., Komachi, M., and Matsumoto, Y. (2011). Narrative schema as world knowledge for coreference resolution. In *Proceedings of the Fifteenth Conference on Computational Natural Language Learning: Shared Task*, pages 86–92, Portland, Oregon, USA. Association for Computational Linguistics.

Kamp, H. and Reyle, U. (1993). *From Discourse to Logic: Introduction to Model-theoretic Semantics of Natural Language, Formal Logic and Discourse Representation Theory*. Studies in Linguistics and Philosophy. Kluwer, Dordrecht.

Lappin, S. and Leass, H. J. (1994). An algorithm for pronominal anaphora resolution. *Computational Linguistics*, 20(4):535–561.

Lee, H., Peirsman, Y., Chang, A., Chambers, N., Surdeanu, M., and Jurafsky, D. (2011). Stanford's multi-pass sieve coreference resolution system at the CoNLL-2011 shared task. In *Proceedings of the Fifteenth Conference on Computational Natural Language Learning: Shared Task*, CONLL Shared Task '11, pages 28–34, Stroudsburg, PA, USA. Association for Computational Linguistics.

Levesque, H. J. (2011). The Winograd Schema Challenge. In *AAAI Spring Symposium: Logical Formalizations of Commonsense Reasoning*.

Lin, T., Mausam, and Etzioni, O. (2010). Identifying Functional Relations in Web Text. In *EMNLP*, pages 1266–1276.

Luo, X. (2005). On coreference resolution performance metrics. In *HLT/EMNLP*, pages 25–32.

McDonald, R., Hall, K., and Mann, G. (2010). Distributed training strategies for the structured perceptron. In *NAACL2010*, pages 456–464.

Ng, V. (2007). Shallow semantics for coreference resolution. In *Proceedings of IJCAI*, pages 1689–1694.

Ng, V. (2010). Supervised noun phrase coreference research: The first fifteen years. In *Proceedings of the ACL*, pages 1396–1411.

Ng, V. and Cardie, C. (2002). Improving machine learning approaches to coreference resolution. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, ACL '02, pages 104–111, Stroudsburg, PA, USA. Association for Computational Linguistics.

Ovchinnikova, E. (2012). *Integration of World Knowledge for Natural Language Understanding*. Atlantis Press, Springer.

Ovchinnikova, E., Montazeri, N., Alexandrov, T., Hobbs, J. R., McCord, M., and Mulkar-Mehta, R. (2011). Abductive Reasoning with a Large Knowledge Base for Discourse Processing. In *Proceedings of IWCS*, pages 225–234, Oxford, UK.

Ponzetto, S. P. and Strube, M. (2006). Exploiting semantic role labeling, wordnet and wikipedia for coreference resolution. In *Proceedings of the main conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics*, HLT-NAACL '06, pages 192–199.

Poon, H. and Domingos, P. (2008). Joint unsupervised coreference resolution with markov logic. In *Proceedings of EMNLP*, pages 650–659.

Pradhan, S., Ramshaw, L., Marcus, M., Palmer, M., Weischedel, R., and Xue, N. (2011). CoNLL-2011 shared task: modeling unrestricted coreference in OntoNotes. In *Proceedings of the Fifteenth Conference on Computational Natural Language Learning: Shared Task*, CONLL Shared Task '11, pages 1–27, Stroudsburg, PA, USA. Association for Computational Linguistics.

Rahman, A. and Ng, V. (2012). Resolving Complex Cases of Definite Pronouns: The Winograd Schema Challenge. In *Proceedings of EMNLP-CoNLL*, pages 777–789.

Recasens, M. and Hovy, E. (2010). BLANC: Implementing the Rand Index for Coreference Evaluation. *Journal of Natural Language Engineering*.

Ruppenhofer, J., Ellsworth, M., Petruck, M., Johnson, C., and Scheffczyk, J. (2010). FrameNet II: Extended Theory and Practice. Technical report, Berkeley, USA.

Shen, D. and Lapata, M. (2007). Using Semantic Roles to Improve Question Answering. In *Proceeding of EMNLP-CoNLL*, pages 12–21.

Song, Y., Jiang, J., Zhao, W. X., Li, S., and Wang, H. (2012). Joint learning for coreference resolution with markov logic. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 1245–1254, Jeju Island, Korea. ACL.

Soon, W. M., Ng, H. T., and Lim, D. C. Y. (2001). A machine learning approach to coreference resolution of noun phrases. *Comput. Linguist.*, 27(4):521–544.

Suchanek, F. M., Kasneci, G., and Weikum, G. (2008). YAGO: A Large Ontology from Wikipedia and WordNet. *J. Web Sem.*, 6(3):203–217.

Yosef, M. A., Hoffart, J., Bordino, I., Spaniol, M., and Weikum, G. (2011). Aida: An online tool for accurate disambiguation of named entities in text and tables. In *Proc. of the 37th Intl. Conference on Very Large Databases (VLDB 2011)*, pages 1450–1453.

N-gram Fragment Sequence Based Unsupervised Domain-Specific Document Readability

Shoaib Jameel Xiaojun Qian Wai Lam
Department of Systems Engineering and Engineering Management,
The Chinese University of Hong Kong.
{msjameel, xjqian, wlam}@se.cuhk.edu.hk

ABSTRACT
Traditional general readability methods tend to underperform in domain-specific document retrieval because they fail to effectively differentiate the reading difficulty of the individual domain-specific terms and the semantic associations between the textual units in a document. On the other hand, recently proposed domain-specific readability methods have relied upon an external knowledge base which may be unavailable in some domains. We develop a novel unsupervised framework for computing domain-specific document readability. Our model does not require an ontology or a knowledge base to capture domain-specific terms in a document. The sequential flow of terms in a document is modeled as a connected sequence of n-gram fragments in the latent concept space. We investigate an automatic sequential n-gram determination scheme that aids in capturing appropriate n-gram fragments which are semantically associated with the document's theme and cohesive with the context. The domain-specific readability cost of a document is computed based on n-gram cohesion and n-gram specificity guided by the latent concepts. The cost can be employed to re-rank the search results generated from an information retrieval (IR) engine. The experimental results demonstrate that our framework achieves significant improvement in ranking documents against the state-of-the-art unsupervised comparative methods.

KEYWORDS: Term Sequence, LSI, IR, Domain-specific Readability, Ranking, Dynamic programming.

Proceedings of COLING 2012: Technical Papers, pages 1309–1326,
COLING 2012, Mumbai, December 2012.

1 Introduction

Readability assessment is an important issue in NLP (Tanaka-Ishii et al., 2010). Traditional general readability methods (Dubay, 2004) have been applied to several problem tasks such as matching books with grade levels (Collins-Thompson and Callan, 2005; Fry, 1969). However, the problem of readability has not been well explored in Information Retrieval (IR) (Kim et al., 2012). Recently researchers have started looking at the problem in IR and several motivations for incorporating readability in an IR system have already been clearly laid out in (Kim et al., 2012; Tan et al., 2012; Nakatani et al., 2009; Yan et al., 2006; Collins-Thompson et al., 2011; Jameel et al., 2011; Zhao and Kan, 2010; Kumaran et al., 2005). What has lacked is a thorough investigation into the problem of reading difficulty in domain-specific IR because traditional unsupervised general readability formulae tend to underperform in domain-specific document retrieval (Yan et al., 2006; Jameel et al., 2011). Domain-specific IR is important because many people are searching for information outside their domain of expertise (Bhavnani, 2002; Yan et al., 2006).

The most affected group who regularly experience difficulties in retrieving documents based on readability are children (Collins-Thompson et al., 2011) and other users who are not domain experts (Yan et al., 2006). Hence, most of them will look for domain-specific documents which they can easily comprehend (Bhavnani, 2002). Domain experts employ complex search strategies such as usage of jargon and complex phrases to successfully retrieve documents based on their reading level (White et al., 2009). Moreover, domain experts know their target destinations such as the ACM Digital Library or Google Scholar using which they can successfully retrieve a document satisfying both relevance and their reading level (White et al., 2009). In contrast, non-domain experts face immense difficulties in formulating a query due to less exposure to the domain-specific terminologies (Vakkari et al., 2003; Paek and Chandrasekar, 2005).

Works which have looked into the problem of domain-specific readability (Yan et al., 2006; Zhao and Kan, 2010) have remained constrained to certain domains only, for instance, the Medical domain because of the required reliance on some knowledge bases to find domain-specific terms in a document. To circumvent this limitation, we propose a novel unsupervised framework for computing domain-specific document readability. The main factor that makes our work superior compared with the existing domain-specific readability methods is that our method does not require an external ontology or lexicon of domain-specific terms. We have previously proposed two terrain models (Jameel et al., 2011, 2012) in Latent Semantic Indexing (LSI) (Berry et al., 1995) to predict the technical difficulty of documents in domain-specific IR using heuristic methods based on conceptual hops between the unigrams and the evaluation is done on one domain only. In this paper, we present an n-gram sequence determination based method which captures appropriate n-gram fragments which are semantically linked with the document automatically while traversing forward following the term sequence in the document. We test the ranking effectiveness of our model on more than one domain. One application of our method is in domain-specific vertical search engines.

Our contributions are as follows: 1) We develop a novel framework to capture suitable n-gram fragments in a domain-specific document by optimizing n-gram fragment sequence connections and taking into account n-gram fragment specificity and cohesion. 2) Our method does not require a domain-specific knowledge base. 3) We conduct extensive experiments on domain-specific document collections in order to show the readability ranking performance.

2 Related Work

Unsupervised heuristic readability methods: Much research has been done in measuring the reading level of text (Qumsiyeh and Ng, 2011). A detailed description about important heuristic readability methods such as Dale-Chall (Dale and Chall, 1948), Automated Readability Index (ARI) (Senter and Smith, 1967), SMOG (McLaughlin, 1969), Coleman-Liau (Coleman and Liau, 1975) etc, can be found in (Dubay, 2004). These methods compute the vocabulary difficulty of a textual discourse. Their readability prediction is based on computing the number of syllables in a term, number of characters etc, which are the surface level features of text. Heuristic readability methods consist of two components linearly combined into a single formula. The components are - syntactic and semantic. These methods have long been in existence and still remain a dominant tool for computing the reading difficulty of traditional documents. In fact, many popular word processing packages use them today. However, readability methods tend to perform poorly on domain-specific texts (Yan et al., 2006) and web pages (Collins-Thompson and Callan, 2005). There are other shortcomings (Bruce et al., 1981) which undermine their importance. In (Nakatani et al., 2009) the authors described an unsupervised method to re-rank the search results of a web search engine in descending order of their comprehensibility using the Japanese Wikipedia but they failed to address the shortcomings in the readability formulae.

Why readability methods underperform on domain-specific documents? Consider a short sentence, "In its simplest form, a star network consists of one central switch, hub or computer, which acts as a conduit to transmit messages." The Flesch reading ease score for this sentence is 62.11, which according to the score is not a difficult sentence. However, the sentence carries a deep technical meaning which requires domain-specific knowledge for proper comprehension. Terms such as "star", "network", and "switch" are domain-specific terms in this example but the readability formula has detected them as easy due to the surface level features.

Domain-Specific Readability Methods: To address the shortcomings inherent in the heuristic readability methods, (Yan et al., 2006) proposed concept based readability ranking method where they have used a domain-specific ontology to capture the domain-specific terms in a document. Their method has a serious drawback in that it requires an ontology for every domain. The authors have only shown the application of their method in one domain. (Kim et al., 2012) described concept readability method in the medical domain. They have used average term and concept familiarity scores from the OAC CHV knowledge base to compute the difficulty of terms and concepts. (Zhao and Kan, 2010) presented domain-specific iterative readability method based on grade levels. Their method is influenced by two popular web link structure based algorithms which are HITS (Kleinberg, 1999) and SALSA (Lempel and Moran, 2001). A limitation of their approach is that they need some seed concepts to initialize their algorithm. This can sometimes be cumbersome as one has to search for a lexicon for every domain. In (Nakatani et al., 2010) the authors used Wikipedia to build a list of some technical terms. In contrast, our proposed framework in this paper does not require an ontology or seed concepts, which can be regarded as a major innovation. The two heuristic terrain models proposed in (Jameel et al., 2011, 2012) computed the technical difficulty of text documents and re-ranked the results obtained from a general purpose IR system. A limitation of the terrain models is that they cannot capture n-grams such as *random access memory* etc. Moreover, they lack a solid theoretical foundation. In this paper, we introduce a principled approach to n-gram fragment determination scheme where we first find the best n-gram sequence in a document automatically. Domain-specific readability cost model is then developed for a document based

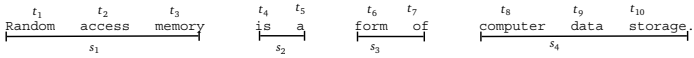


Figure 1: A sentence with four n-gram fragments (s_1, s_2, s_3, s_4). This sample sentence has been taken from one of the Wikipedia documents in our Science test collection. The fragmented sentence has been obtained using Equation 5.

on a new formulation of cohesion and specificity.

Supervised Methods for Readability: Although our proposed framework is completely unsupervised, some supervised methods for computing the reading difficulty of text have been proposed as well (François and Miltsakaki, 2012). Supervised learning approach for readability can be considered as a classification problem. In (Liu et al., 2004), the authors have used Support Vector Machines (SVM) (Vapnik, 1995) for recognizing the reading levels of texts from user queries. They have used syntactic and vocabulary based features to train the classifier. Language modeling has been applied to readability (Collins-Thompson and Callan, 2005) where the authors described a smoothed unigram model for computing the readability of text documents such as web pages. In (Si and Callan, 2001), the authors also used unigram language model to predict readability. Topic familiarity is different from traditional general readability (Kumaran et al., 2005), where the authors studied re-ranking of a search engine result based on familiarity. They also studied the importance of stopwords in their familiarity classifier (FAMCLASS). In (Leroy et al., 2008), classification of health related documents into three levels, namely, Beginner, Intermediate and Advanced is discussed. The authors achieved high classification accuracy using their classifier. In (Schwarm and Ostendorf, 2005; Petersen and Ostendorf, 2009) the authors combined word level features with other textual features. They have used SVM together with several word level features to classify documents based on readability. In (Heilman et al., 2008) the authors introduced a k-Nearest Neighbor classifier based on grammatical features such as sentence length and the patterns of the parse tree. (Bendersky et al., 2011) used several features including readability to improve relevance ranking of the web search results.

Readability is a relative measure (Van Oosten and Hoste, 2011). In order to cater to the results on an individual user basis, recently, methods using query log analysis have been proposed. Search engine query log mining and building individual user profile classifier can also help to solve the problem to some extent as done in (Collins-Thompson et al., 2011; Tan et al., 2012; Kim et al., 2012). But this requires confidential and proprietary query log data with private user session details (Silverstein et al., 1999). Many users might not want their sessions to be recorded or used due to privacy concerns (Jones et al., 2007).

Readability has also been studied in computational linguistics (Leroy and Endicott, 2012). In (Kate et al., 2010) the authors used several linguistic and language model features to build a classifier to predict readability of texts. Language model features were found out to be important to their classifier. (Pitler and Nenkova, 2008) used several textual features in their classifier. Their result shows that word features and average sentence length are strong predictors but the strongest ones are the discourse features. One major limitation of the supervised methods is that one needs a large amount of expensive annotated data (Kanungo and Orr, 2009). Language modeling approaches cannot capture domain-specific concepts in a domain (Zhao and Kan, 2010). In contrast, our method does not need any annotated data.

3 Background and Overview

We tackle the problem of domain-specific readability of text documents. We take as input a document collection of related documents of a domain. We compute the domain-specific readability of each document in the collection. Given a query, a similarity-based IR system retrieves documents. We then re-rank the retrieved top-k results automatically based on readability.

3.1 Overview

We address the problem of domain-specific document readability based on an automatic scheme for finding appropriate n-grams in the Latent Semantic Indexing (LSI) (Berry et al., 1995) latent concept space. As in the previous domain-specific readability approaches, the task lies in an effective capturing of domain-specific terms in the document and their individual specificity scores or scopes (e.g. document scope in (Yan et al., 2006)). In the LSI latent space, n-gram fragments which are central to a document, mainly the domain-specific terms, come close to their document vectors because the semantic fabric of the n-gram fragments inclines best with the technical content of the document (Bellegarda, 2000; Jameel et al., 2011). Common n-grams will not be coherently linked with the document semantics. They can be considered as non-central in that technical storyline (Bellegarda, 2000). N-grams which are semantically similar in meaning will cluster close to each other in the latent space forming a word cloud of semantically related terms (Berry et al., 1995).

We denote the sequence of unigrams in a document d as $(t_1, t_2, \cdots, t_W)$. In Figure 1, an ordered sequence of terms $(t_1, t_2, \cdots, t_{10})$ is shown. Using our proposed methods, the sequence of unigrams can be formed into a sequence comprising of n-gram fragments such as "random access memory" or a connective such as "is a", which are examples of n-gram fragments of order 3 and 2 respectively. An example of a sequence of variable length n-gram fragments is also shown in Figure 1, which is composed of $S = (s_1, s_2, s_3, s_4)$. The sequential flow of terms in a document is modeled as a sequence of n-gram fragments. In this paper, we investigate an automatic sequential n-gram determination scheme that aids in capturing the appropriate n-grams which are semantically associated with the document's theme and cohesive with the context. Cost expended during n-gram formation step can be regarded as a reading difficulty cost of a document. Our proposed n-gram sequence cost optimization linearly combines the effect of both cohesion and specificity, which play a dominant role in determining the domain-specific readability of a document. Some domain-specific readability methods have similar consideration of cohesion and term's domain-specific importance such as (Jameel et al., 2011, 2012; Yan et al., 2006), but they have some serious limitations as mentioned in Section 2.

The first step of our framework is to generate all n-gram fragments (i.e. unigrams, bi-grams, tri-grams etc.) in a document with a predefined maximum value of n, and subsequently we construct a weighted n-gram fragment-document matrix using the product of normalized n-gram frequency and inverse n-gram document frequency (formulae given in (Salton and Buckley, 1988)), where rows are represented by n-gram fragments and columns by documents. Then we perform LSI and obtain a low-dimensional representation of the original vector space. The main computation in LSI is Singular Value Decomposition (SVD) (Golub and Reinsch, 1970). Computing the SVD of a matrix is generally computationally expensive both in space and time (Berry et al., 1995). But with the fast development of better algorithms to compute the SVD, such concerns have been addressed (Wang et al., 2011; Zha et al., 1998).

In (Yan et al., 2006) the authors introduced two components in determining the overall

domain-specific readability of a document which are "document scope" and "document cohesion". The hypothesis is that readability of a document will not only depend on the reading difficulty of the individual terms but also on how the terms in the document are related to one another in the document. Hence a document comprising of many domain-specific terms will be difficult to read and also if the terms are not related to each other (low cohesion) in the same document then the reader will face difficulties in relating different concepts of a domain (Yan et al., 2006). However, the computation of document scope and cohesion in (Yan et al., 2006) is accomplished using an ontology tree which requires an ontology for every domain. Our proposed computation of document scope is different as that in (Yan et al., 2006). We introduce "n-gram specificity" which is able to capture document scope more effectively. As stated earlier, an n-gram fragment which is coherently linked with the technical storyline of a document will be close (Bellegarda, 2000; Jameel et al., 2011, 2012) to the document in the LSI latent space and thus will be more *specific* to that technical storyline. We use the notion of closeness of an n-gram vector to the document vector in the LSI latent space to compute the n-gram specificity.

Psychologists have studied various aspects which lead a reader to comprehend particular piece of discourse (Graesser et al., 1997). An important aspect in text comprehension is cohesion between texts. Cohesion is the property of text in which the units are semantically related to each other and describe one theme. Cohesion is important because the interpretation of one element of text depends on that of another. Texts frequently exhibit varying degrees of cohesion in different sections and hence the start of the text will not be cohesive with the end of the same text (Halliday and Hasan, 1976). Our work is inspired by this observation and we maintain the term order in the document in order to compute cohesion between the n-gram fragments in sequence. Text cohesion has been discussed in (Yan et al., 2006; Ferstl and von Cramon, 2001; Morris and Hirst, 2006; Moe, 1979; Graesser et al., 2004) which establish relation between cohesion and comprehension. Accurate comprehension of technical texts requires accurate identification of the technical meaning of the terms and connections between the terms with the surrounding parts of the text (Freebody and Anderson, 1983). Describing too many difficult non-cohesive terms in sequence will make the reading path of the reader troublesome and thus will affect discourse comprehension (McNamara et al., 1996).

One may argue why we have used a conceptual model instead of the original high-dimensional vector space? An obvious bottleneck in the original vector space is the curse of dimensionality as one has to deal with the space which will be enormous. The high dimensionality would lead to huge computational cost. Moreover, obtaining the n-gram fragment and document semantic relationships directly from the vector space is not possible (Berry et al., 1995) unless additional techniques such as LSI (Deerwester et al., 1990) or the method described in (Yamamoto and Church, 2001) is applied. LSI also handles issues related to data sparsity.

4 Sequential N-gram Connection Model (SNCM)

We present our model that establishes sequential n-gram connections in the document and eventually a cost is expended in order to make such a connection of n-grams in sequence. The aggregated cost expended in the document can be regarded as a document's domain-specific readability cost. If the textual units are not cohesive, then the reader faces cognitive difficulties in comprehension. In addition, if the individual textual units are difficult, then the reader expends cognitive load in figuring out the inherent meaning of the textual unit while reading the text (Yan et al., 2006). This phenomenon can be captured in a cost computation

model described below.

Let s be an n-gram fragment. Let d be the document where this n-gram fragment occurs. Let this fragment be represented as a vector in the LSI latent space as $\vec{s}$ and the document vector as $\vec{d}$. We compute the n-gram specificity, $\vartheta(\vec{s},\vec{d})$, using the following formula:

$$\vartheta(\vec{s},\vec{d}) = \text{cosine\_sim}(\vec{s},\vec{d}) \tag{1}$$

where cosine_sim$(\vec{s},\vec{d})$ is the cosine similarity (formula given in (Bellegarda, 2000)) between the n-gram vector $\vec{s}$ and the document vector $\vec{d}$ in the latent space. An n-gram fragment will obtain a high cosine similarity if it is close to the document vector in the LSI latent space and a low cosine similarity value if it is not close. Therefore, what we expect is that domain-specific n-grams will obtain a high cosine similarity (Jameel et al., 2012) value compared with common/general n-gram fragments. In (Park et al., 2002), they named a domain-specific term extraction scheme as *Degree of Domain-specificity*. However, their method deals with a completely different problem task.

Suppose $T = (t_1, t_2, \cdots, t_W)$ is the term sequence and $S = (s_1, s_2, \cdots, s_K)$ is one particular n-gram fragmented sequence of T, W is the total number of terms in the document d, K is the number of n-grams in S. We denote n-gram cohesion, $\eta(\vec{s_i}, \vec{s_{i+1}})$, between the two n-grams s_i and s_{i+1} at positions i and $i+1$ in sequence whose vectors are represented as $\vec{s_i}$ and $\vec{s_{i+1}}$ respectively for a particular document as:

$$\eta(\vec{s_i}, \vec{s_{i+1}}) = \text{cosine\_sim}(\vec{s_i}, \vec{s_{i+1}}) \tag{2}$$

When the cosine similarity between the two consecutive n-grams is high, the n-gram fragments are cohesive. The reason is that in the latent concept space n-gram fragments which appear under similar storylines and similar semantic meaning will cluster close to each other (Berry et al., 1995; Jameel et al., 2011, 2012). Hence the closer they are, the more semantically related they tend to become. A reader will be able to semantically relate those n-gram fragments (which are cohesive) easily (Halliday and Hasan, 1976) and will comprehend a piece of textual discourse well. A document tends to be semantically readable if the constituent terms are simple (i.e. low specificity values) with reference to the document vector in the LSI latent space. Therefore, we hypothesize that specificity values will be directly proportional to the document's overall domain-specific reading difficulty. In fact, in order to compute document generality and readability, (Yan et al., 2006, 2011) have made a similar hypothesis and have evaluated their hypothesis through experiments. We also hypothesize that cohesion is inversely related to the document domain-specific readability. Again, this assumption is in line with the assumptions made in (Yan et al., 2006, 2011).

4.1 Our First Model (SNCM1)

Our framework works towards determining a least cost n-gram connected sequence in the document where at each forward transition sequential n-gram cohesion is minimized. The sequence of terms consisting of variable n-gram fragments is considered while traversing forward. For a particular document, suppose the term sequence T and its n-gram fragmented sequence S are defined in a similar manner as above. The cost of the n-gram fragment sequence S, $C_1^{(d)}(S)$, can be written as:

$$C_1^{(d)}(S) = \sum_{k=1}^{K} \left(\frac{1}{\eta(\vec{s_{k-1}}, \vec{s_k}) + 1} \right) \tag{3}$$

1 is added in the denominator to handle the cases where the n-gram vectors are orthogonal to each other. Our goal is to minimize this cost, $C_1^{(d)}(S)$. The rationale for such minimization

formulation is to fit the most cohesive n-gram fragment in sequence which matches with the sequential storyline of the document. We achieve this using the following optimization scheme given in Equation 4.

$$\min_{S} C_1^{(d)}(S) \tag{4}$$

This scheme ensures that an n-gram will be the least cost match at that position if it cohesively fits in that sequential discourse. We now describe a dynamic programming method to find the optimal cost. We define $C_1^{(d)}(T_i)$ as the optimal cost from the beginning until the term t_i in the document. Since the accumulated cost is the sum of the local costs, it can be decomposed in the same way as its predecessors and the local cost accumulated with the n-gram itself. To obtain the optimal path cost, we have to select the predecessor with the minimum total cost. Another issue is that we need to set a maximum bound for the number of terms in an n-gram. In principle, this bound could be set to any number m. Let $\vec{S_X}$ be a unigram composed of t_i, $\vec{S_Y}$ be a bigram composed of (t_{i-1}, t_i) and $\vec{S_Z}$ be an m-gram composed of $(t_{i-m+1}, \cdots, t_i)$. $\vec{S_{X-1}}$, $\vec{S_{Y-1}}$ and $\vec{S_{Z-1}}$ represent the particular n-gram (where n may be from 1 to m) in the optimal sequential path that appears just before $\vec{S_X}$, $\vec{S_Y}$ and $\vec{S_Z}$ respectively. The optimal cost for all the terms from t_1 until position t_i, (denoted as $C_1^{(d)}(T_i)$) can be written as:

$$\begin{aligned} C_1^{(d)}(T_i) = \text{minimum}\Big(& C_1^{(d)}(T_{i-1}) + \frac{1}{\eta(\vec{S_{X-1}}, \vec{S_X}) + 1}, \\ & C_1^{(d)}(T_{i-2}) + \frac{1}{\eta(\vec{S_{Y-1}}, \vec{S_Y}) + 1}, \\ & \cdots, \\ & \cdots, \\ & C_1^{(d)}(T_{i-m}) + \frac{1}{\eta(\vec{S_{Z-1}}, \vec{S_Z}) + 1}\Big) \end{aligned} \tag{5}$$

The final reading difficulty of a document will not only depend on cohesion but also specificity. Therefore, to compute the final readability cost of a text document, we linearly combine specificity values of the n-grams formed during sequential linear n-gram determination scheme using Equation 5. The overall document domain-specific readability cost, $E_1^{(d)}$, is given in Equation 6 where α ($0 \leq \alpha \leq 1$) is a parameter controlling the relative contribution of cohesion and specificity. A higher cost indicates that the document is difficult to read and a low cost is indicative of the ease in reading the document. We shall use the cost values to re-rank the search results obtained from a general purpose IR system.

$$E_1^{(d)} = \frac{\alpha C_1^{(d)}(T_W) + (1-\alpha)\sum_{i=1}^{K} \vartheta(\vec{s_i}, \vec{d})}{W} \tag{6}$$

where W is the total number of terms in the document and it removes the document length bias. Note that W in the denominator is more suitable than K because the reading difficulty of a document is not dependent on the number of n-gram fragments formed.

4.2 An Extended Model: SNCM2

We now modify our previous model in Equation 3 further and extend to the case where we combine the effect of both cohesion and specificity. In this model, we linearly combine the effect of specificity along with cohesion during n-gram fragment determination phase. We design the cost, $C_2^{(d)}(S)$, of an n-gram fragment sequence formation S as:

$$C_2^{(d)}(S) = \sum_{k=1}^{K} \left(\beta\, \vartheta(\vec{s_k}, \vec{d}) + (1-\beta) \frac{1}{\eta(\vec{s_{k-1}}, \vec{s_k}) + 1} \right) \tag{7}$$

β $(0 \leq \beta \leq 1)$ is a parameter controlling the relative weights of the two components. Our goal is to minimize the total cost $C_2^{(d)}(S)$ as follows:

$$\min_{S} C_2^{(d)}(S) \tag{8}$$

We can apply similar dynamic programming methodology. Let the optimal cost for all the terms from t_1 until position t_i be $C_2^{(d)}(T_i)$.

$$\begin{aligned} C_2^{(d)}(T_i) = \text{minimum} \\ \Bigg(& C_2^{(d)}(T_{i-1}) + \beta\, \vartheta(\vec{S_X}, \vec{d}) + (1-\beta) \frac{1}{\eta(\vec{S_{X-1}}, \vec{S_X} + 1)}, \\ & C_2^{(d)}(T_{i-2}) + \beta\, \vartheta(\vec{S_Y}, \vec{d}) + (1-\beta) \frac{1}{\eta(\vec{S_{Y-1}}, \vec{S_Y} + 1)}, \\ & \cdots, \\ & \cdots, \\ & C_2^{(d)}(T_{i-m}) + \beta\, \vartheta(\vec{S_Z}, \vec{d}) + (1-\beta) \frac{1}{\eta(\vec{S_{Z-1}}, \vec{S_Z} + 1)} \Bigg) \end{aligned} \tag{9}$$

The overall document domain-specific readability cost $E_2^{(d)}$ is given in Equation 10. We rank the documents based on an optimal cost obtained at the end of the n-gram sequence formation.

$$E_2^{(d)} = \frac{C_2^{(d)}(T_W)}{W} \tag{10}$$

Intuitive Justification: Specificity helps us in finding an n-gram fragment which matches with the technical storyline of the entire document. Cohesion on the other hand helps in finding the best linked n-gram fragments in the sequential discourse based on the context. In Equation 9 our objective is to find a least cost n-gram fragment that is a best match in the n-gram sequence which considers both components, namely, cohesion and specificity simultaneously in the document. The intuition behind adopting this strategy is to find n-gram fragments in a document which are semantically linked with the document's thematic structure and are cohesive with the context. However, in Equation 5 we are minimizing only cohesion between the n-gram fragments in sequence. This strategy may help us find cohesive n-grams with the context but the n-grams may not be thematically linked with the technical storyline of the document because of the absence of specificity during n-gram sequence determination phase. A closer look at the two variants of our model paints some interesting pictures. When $\beta = 0$ and $\alpha = 1$ the two variants (`SNCM1` and `SNCM2`) behave equivalently. Note that the cost expended to fit a specific n-gram fragment will be more in comparison to an n-gram fragment which is common/general. The longer the contextual history m, the better n-gram fragment prediction will be. However, longer contextual histories shall bring about additional computational burden especially for large datasets as ours.

5 Empirical Evaluation

5.0.1 Experimental Setup

Testbed Data: Current IR evaluation test sets such as TREC, and CLEF cannot be used in our experiments due to lack of readability annotations. Currently, they only contain relevance

judgments. So we chose two popular domains 1) Psychology, and 2) Science and subsequently we crawled a large number of web pages in these domains. We enlist some of the important resources from where we crawled the majority of the web pages as enlisting every crawled resource would be too long. Psychology web pages were crawled from: 1) Wikipedia, 2) Psychology.com, 3) Psychology Today 4) Simple English Wikipedia. Science web pages were crawled from: 1) ScienceDaily, 2) ScienceForKids, 4) Simple English Wikipedia 5) Wikipedia, 6) About.com and some more related web resources. We also included Computer Science documents in the Science domain. The reason for choosing these online resources was mainly due to their popularity and high quality content. By crawling web pages from different resources available online we are able to collect domain-specific documents which match diverse genres and audience. In all, our test collection includes 170,000 documents in Psychology with 154,512 n-grams in the vocabulary, and 300,000 documents in Science consisting of 490,770 n-grams in the vocabulary. No term stemming was performed as we wish to keep the original words. In fact, traditional unsupervised readability methods do not consider stemmed terms. We prepared two sets of document collection, one with stopwords[1] kept and another with stopwords removed. The objective is to study the role of stopwords in domain-specific readability (Refer Section 2). Note that we conduct experiments in each domain separately.

We used Zettair[2] to conduct document retrieval and obtained a ranked list using Okapi BM25 (Robertson et al., 1996) ranking function. BM25 retrieves documents based on relevance and the retrieved list contained a mix of easy readable and difficult to read documents. We then selected top-k documents retrieved from the ranked list where $k = 10$ for evaluation purpose. Selecting a higher value of k would lead to a huge cognitive load on the human subjects (which we describe later in the text) during annotation. Therefore, we keep this number as low as possible. Also, a previous study has found that users generally look at the first page of the search results containing the top ten documents (Silverstein et al., 1999).

Domain-specific information needs: Topics are queries posed to an IR system. We strictly followed topic development guidelines laid out in "INEX 2009 Topic Development Guidelines"[3]. We had asked two undergraduate students possessing beginner level knowledge in both Science and Psychology to generate domain-specific topics which represent real information needs. They generated 110 topics in Psychology and 150 topics in Science. We enlist some sample information needs in two domains here: Science: 1) "x-ray machine", 2) "acid and alkali" 3) "why the sky is blue", Psychology: 1) "depression", 2) "bad dreams", 3) "school of Psychology".

Annotations and metrics: To obtain the ground truth of domain-specific readability of the documents for evaluation purpose, two human annotators who were undergraduate students having varied background were invited. They had basic knowledge about Science and Psychology. They were asked to rate the documents based on relative domain-specific readability judgment of the documents. For the judgment, the selection was among "very low domain-specific readability" (i.e. difficult to read), "reasonably low domain-specific readability", "average domain-specific readability", "reasonably high domain-specific readability" and "very high domain-specific readability". These options were further translated to integer gains ranging from 4 to 0. A simple readable document obtained a score of 4 whereas the most difficult obtained a score of 0. In the beginning we acquainted them with the main aim of the study

[1] http://jmlr.csail.mit.edu/papers/volume5/lewis04a/a11-smart-stop-list/english.stop
[2] http://www.seg.rmit.edu.au/zettair/
[3] http://www.inex.otago.ac.nz/tracks/adhoc/gtd.asp

and showed them some sample documents from our test collection so that they could get an idea about the relative difficulty levels of the documents in the collection. Overall, the annotators annotated 983 documents in Psychology and 1442 documents in Science. In order to ascertain whether the manual annotation that we collected was feasible and reproducible, we assessed the inter-annotator agreement by computing the Cohen's Kappa coefficient. We found that there was an acceptable agreement between the annotators (approximately 0.8) in both Psychology and Science domains.

The evaluation metric is NDCG (same formula as in (Cai et al., 2011)) which is widely used for IR ranking effectiveness measurement. We computed the NDCG@i for each annotator and aggregated the final NDCG by taking the average. NDCG is well suited for our task because it is defined by an explicit position discount factor and it can leverage the judgments in terms of multiple ordered categories. NDCG@i scores will directly correlate with the readability annotation of the documents given by humans. Such scores can measure the quality of difficulty ranking of documents based on readability judgments provided by humans. If NDCG is high, it means the ranking function correlates better with the human judgments.

Result re-ranking scheme: We automatically re-rank the search results obtained from an IR system from *simple* to *difficult* readable documents using our proposed model as well as comparative methods. The reason is that domain experts normally employ complex search strategies to successfully retrieve documents based on their reading level (refer Section 2). They can find their material of interest easily but non-experts face difficulty in locating their content as they have to sift through the ranked list carefully to locate a document which can match their domain-specific reading level. In addition, previous related approaches have also followed similar re-ranking scheme (i.e. re-ranking from simple to advanced without integrating the readability scores in the initial retrieval score) such as (Yan et al., 2006; Nakatani et al., 2009; Kumaran et al., 2005) and in (Yan et al., 2011), the authors re-rank the top-k documents obtained from a baseline IR system based on decreasing specificity.

Comparative methods: We chose popular unsupervised general readability methods as our comparative models. They are ARI: Automated Readability Index, Coleman-Liau (denoted as C-L in the tables in our results), Flesch Reading Ease formula, Fog, LIX and SMOG. More details about these readability methods can be found in (Dubay, 2004). Our model does not have a syntactic component. Hence, it would be more appropriate to compare with the semantic components of the general readability methods (similar scheme also adopted in (Yan et al., 2006; Collins-Thompson and Callan, 2005). More details about the semantic components can be found in (Yan et al., 2006; Collins-Thompson and Callan, 2005; Dubay, 2004). For each readability formula; it computes a readability score for every document. Then the documents are re-ranked in descending order of the readability score. In addition, we also chose some recent unsupervised comparative methods described in (Collins-Thompson and Callan, 2005) such as Mean Log Frequency (denoted as `MLF`) and %UNK. We also compare our method with `CHM` described in Jameel et al., (Jameel et al., 2011). We denote their method as `CHM`.

In addition, we compare our method by manually collecting an extensive list of domain-specific concepts from online resources. The collection process was indeed cumbersome and time consuming but it will help us evaluate our method by mimicking the working principle of the recently proposed domain-specific readability methods which rely on some external knowledge-bases. Overall we collected about 900 domain-specific concepts in Science and about 600 in Psychology. In this method, we count how many times domain-specific terms occur in the

(a) Psychology

	NDCG@3	NDCG@5	NDCG@7	NDCG@10
ARI	0.515	0.548	0.582	0.618
C-L	0.525	0.553	0.584	0.612
Flesch	0.449	0.490	0.537	0.579
Fog	0.513	0.547	0.577	0.612
LIX	0.516	0.550	0.584	0.619
SMOG	0.517	0.550	0.579	0.616
CHM	0.465	0.456	0.473	0.482
Counts	0.551	0.575	0.603	0.649
MLF	0.530	0.554	0.581	0.631
%UNK	0.558	0.585	0.611	0.653
SNCM1	0.537	0.571	0.602	0.651
SNCM2	**0.581***	**0.607***	**0.635***	**0.680***

(b) Science

	NDCG@3	NDCG@5	NDCG@7	NDCG@10
ARI	0.524	0.547	0.562	0.564
C-L	0.541	0.551	0.572	0.576
Flesch	0.554	0.560	0.566	0.574
Fog	0.593	0.508	0.538	0.640
LIX	0.541	0.562	0.583	0.585
SMOG	0.584	0.538	0.500	0.523
CHM	0.400	0.406	0.407	0.412
Counts	0.595	0.563	0.564	0.627
MLF	0.557	0.584	0.611	0.657
%UNK	0.562	0.590	0.619	0.660
SNCM1	**0.617***	**0.645***	**0.672***	**0.713***
SNCM2	0.602*	0.625*	0.650*	0.702*

Table 1: Comparison of SNCM variants when $\alpha = \beta = 0.5$ against the comparative methods in both domains. * denotes statistically significant results for all comparisons according to paired t-test ($p < 0.05$).

Doc1	Specificity	Syl	Doc2	Specificity	Syl
earth science	0.74	2	**cancer**	0.71	2
earth	0.78	1	in-spite	0.12	2
any	0.09	1	**lung cancer**	0.71	3
mapped	0.51	2	management	0.18	4

Table 2: N-gram specificity values obtained from two separate documents in our collection using Equation 1. We compare the specificity values with the number of syllables in the n-gram (denoted as Syl).

document with respect to the list contained in the lexicon and then divide by the number of words in the document to remove document length bias. We name this comparative method as Counts.

We had set the value for m in Equations 5 and 9 to 3 in our experiments. This could help capture tri-gram fragments which we believe is a suitable number for large datasets. We used MATLAB to compute SVD of the matrix using the "svds" function. The number of latent concepts in LSI were 200 as previous studies have found that 150-200 dimensions give optimal performance (Dumais, 1995). Our term weighting scheme was the product of normalized n-gram count and inverse n-gram document frequency (formulae given in (Salton and Buckley, 1988)). Our main models are SNCM1 and SNCM2 with stopwords kept intact. Our objective is to study the performance of our proposed variants by keeping the entire documents sequence intact without removal of any of the features as term order plays an important role in our model. Moreover, general traditional readability methods also work on original texts. We had set the value of $\alpha = \beta = 0.5$ in our experiments, which means that both the components have equal weights in determining the final readability ranking order.

5.0.2 Results and Analysis

To enlighten the reader more about the specificity values, we show some specificity values obtained from our experimental dataset in Table 2. The technical storyline of $Doc1$ revolves around *Earth Science* and $Doc2$ deals with *Lung Cancer*. Domain-specific terms which we indicate in bold text have obtained higher specificity values compared to common n-gram fragments. We can also observe that the domain-specific terms appear simple in terms of the number of syllables (denoted as Syl). Such n-grams will appear common to any readability formula which relies on the number of syllables for text readability prediction. Note that for a readability formula, if the number of syllables is more, the word is difficult.

(a) Psychology

Method Name	Queries Improved		Average Improvement	
	SNCM1	SNCM2	SNCM1	SNCM2
ARI	53	59	17.56%	18.06%
C-L	61	61	22.84%	22.86%
Flesch	65	65	25.66%	25.66%
Fog	68	65	20.02%	17.12%
LIX	60	62	22.05%	24.03%
SMOG	58	60	23%	23.08%
CHM	86	88	36%	38%
Counts	29	40	1.02%	12.05%
MLF	49	60	2.01%	20.76%
%UNK	3	32	0	9.34%

(b) Science

Method Name	Queries Improved		Average Improvement	
	SNCM1	SNCM2	SNCM1	SNCM2
ARI	95	95	22.34%	22.01%
C-L	90	91	20.12%	20.36%
Flesch	92	92	21.56%	21.50%
Fog	80	80	17.90%	17.90%
LIX	90	90	20.19%	20.13%
SMOG	92	92	25.56%	26%
CHM	121	119	32%	29.99%
Counts	82	79	19.76%	17.55%
MLF	83	75	21.45%	19.23%
%UNK	77	69	17.55%	16.53%

Table 3: Performance comparison based on queries for SNCM1 and SNCM2.

NDCG@i	α=0	α=0.2	α=0.5	α=0.8	α=1
@3	0.506	0.524	0.537	0.566	0.573
@5	0.545	0.586	0.571	0.583	0.599
@7	0.579	0.605	0.602	0.630	0.627
@10	0.631	0.623	0.651	0.547	0.672

Table 4: Varying α for SNCM1 in Psychology with stopwords.

NDCG@i	α=0	α=0.2	α=0.5	α=0.8	α=1
@3	0.496	0.499	0.498	0.537	0.567
@5	0.534	0.525	0.545	0.577	0.587
@7	0.570	0.571	0.574	0.598	0.611
@10	0.624	0.631	0.666	0.640	0.658

Table 5: Varying α for SNCM1 in Psychology without stopwords.

NDCG@i	α=0	α=0.2	α=0.5	α=0.8	α=1
@3	0.603	0.604	0.617	0.545	0.599
@5	0.631	0.633	0.645	0.630	0.622
@7	0.657	0.646	0.672	0.639	0.647
@10	0.697	0.698	0.713	0.710	0.700

Table 6: Varying α for SNCM1 in Science with stopwords.

The main result of the document ranking performance is given in Table 1. In the Psychology domain, SNCM2 has performed better than other comparative methods whereas in the Science domain we notice that SNCM1 has fared better than all other models. We performed a paired t-test between the SNCM variants and the comparative methods. We have obtained statistically significant improvement with ($p < 0.05$) for all comparisons. Counts did not perform well in both domains. An obvious reason is that this method demands a longer list of technical terms, which is extremely time consuming to obtain. Readability methods have failed to give optimal ranking performance. It is because they fail to capture the inherent semantics of text which our method can effectively capture. CHM performed rather poorly in both the domains. One reason could be due to the weak model and incorporation of non-linearity using a heuristic approach. %UNK has shown some good performance. One reason is due to the use of an elaborate list of words (over 3000). In Table 3 we present results based on the improvement we have obtained on query basis. Results show that our models have obtained tangible improvement against the comparative methods.

Some interesting conclusions can be derived from the results in Tables 4,5,6,7,8,9,10 and 11. These results highlight the role of the two components, namely, cohesion and specificity in influencing the overall ranking of the results. Tables 4,5,8 and 9 show the effect of varying α and β in the Psychology domain. Our discussion will mainly focus on the results when $\alpha = 0$, $\alpha = 1$ and $\beta = 0$, $\beta = 1$ because these values portray the contribution that the two components, namely, cohesion and specificity individually make in the overall ranking of the search results. We notice in the Psychology domain that ranking of the search results is significantly dominated by cohesion (note the values close to $\alpha = 1$ and $\beta = 0$). This observation can be reasoned out from the usage of terms across the documents in the collection. We noticed in the Psychology corpus that the documents are more general than the Science documents. Science documents contain relatively more domain-specific terms than Psychology documents. Thus the contribution of specificity will be more uniform across documents in Psychology than in Science. Hence usage of terminologies will be almost at the same level.

We obtained some interesting results in the Science domain as well. In Tables 6 and 7, we note

that the gap in the results when $\alpha = 0$ and $\alpha = 1$ is not very wide. This means that both the components have approximately equal role in affecting the final ranking of the search results. However an interesting conclusion is that cohesion has a slightly more dominant effect than specificity but the importance of specificity cannot be completely disregarded (we conclude this when $\alpha = \beta = 0.5$). Even for SNCM2 in Tables 10 and 11 the observations remain the same where we note that both components have almost equal role in affecting the overall ranking of the results.

NDCG@i	α=0	α=0.2	α=0.5	α=0.8	α=1
@3	0.607	0.609	0.598	0.590	0.590
@5	0.633	0.633	0.620	0.606	0.617
@7	0.658	0.659	0.650	0.637	0.642
@10	0.699	0.701	0.698	0.691	0.693

Table 7: Varying α for SNCM1 in Science without stopwords.

NDCG@i	β=0	β=0.2	β=0.5	β=0.8	β=1
@3	0.573	0.576	0.581	0.573	0.509
@5	0.599	0.602	0.607	0.598	0.548
@7	0.627	0.630	0.635	0.630	0.582
@10	0.672	0.675	0.680	0.675	0.634

Table 8: Varying β for SNCM2 in Psychology with stopwords.

NDCG@i	β=0	β=0.2	β=0.5	β=0.8	β=1
@3	0.567	0.565	0.573	0.574	0.503
@5	0.587	0.585	0.591	0.592	0.542
@7	0.611	0.610	0.615	0.615	0.578
@10	0.658	0.656	0.660	0.661	0.630

Table 9: Varying β for SNCM2 in Psychology without stopwords.

NDCG@i	β=0	β=0.2	β=0.5	β=0.8	β=1
@3	0.599	0.602	0.602	0.603	0.618
@5	0.622	0.625	0.625	0.628	0.646
@7	0.647	0.649	0.650	0.651	0.670
@10	0.700	0.702	0.702	0.704	0.719

Table 10: Varying β for SNCM2 in Science with stopwords.

NDCG@i	β=0	β=0.2	β=0.5	β=0.8	β=1
@3	0.590	0.590	0.594	0.587	0.620
@5	0.617	0.617	0.620	0.615	0.645
@7	0.642	0.642	0.644	0.641	0.670
@10	0.693	0.695	0.697	0.694	0.717

Table 11: Varying β for SNCM2 in Science without stopwords.

We can now infer that cohesion has a more deep seated role compared to specificity in the two domains but specificity cannot be completely disregarded. We also studied the behavior of SNCM variants along with the role of stopwords in the two domains. From the results we note the stopwords have an important role to play in influencing ranking. This stands consistent with the prior findings discussed in Section 2.

6 Conclusions and Future Work

We have presented our SNCM models where we form a fragmented n-gram sequence in a document. We find a least cost path in the n-gram sequence. The cost reflects the domain-specific readability of a text document. We have shown that general readability methods and other state-of-the-art unsupervised methods are not effective to determine the readability of a text document. Experiments in two domains show the superiority of our proposed models. Our proposed approach is more scalable than recently proposed domain-specific readability methods because we do not use any external domain-specific ontology to capture domain-specific terms.

In the future, we would study how the hyperlink structure of the web can aid in determining the reading difficulty of text documents. The hypothesis is that a general web page would link with other general web pages (Akamatsu et al., 2011) as well. We would also explore other features which could help improve readability ranking performance such as a web page layout and content such as fonts, title fields, line and paragraph breaks, etc.

7 Acknowledgements

The work described in this paper is substantially supported by grants from the Research Grant Council of the Hong Kong Special Administrative Region, China (Project Code: CUHK413510) and the Direct Grant of the Faculty of Engineering, CUHK (Project Codes: 2050476 and 2050522). This work is also affiliated with the CUHK MoE-Microsoft Key Laboratory of Human-centric Computing and Interface Technologies.

References

Akamatsu, K., Pattanasri, N., Jatowt, A., and Tanaka, K. (2011). Measuring comprehensibility of web pages based on link analysis. In *Proc. of WI-IAT*, pages 40–46.

Bellegarda, J. (2000). Large vocabulary speech recognition with multispan statistical language models. *IEEE Transactions on Speech and Audio Processing*, 8(1):76 –84.

Bendersky, M., Croft, W. B., and Diao, Y. (2011). Quality-biased ranking of web documents. In *Proc. of WSDM*, pages 95–104.

Berry, M. W., Dumais, S. T., and O'Brien, G. W. (1995). Using linear algebra for intelligent information retrieval. *SIAM Rev.*, 37(4):573–595.

Bhavnani, S. K. (2002). Domain-specific search strategies for the effective retrieval of healthcare and shopping information. In *Human Factors in Computing Systems*, pages 610–611.

Bruce, B., Rubin, A., and Starr, K. S. (1981). Why readability formulas fail. *IEEE Transactions on Professional Communication*, pages 50–52.

Cai, P., Gao, W., Zhou, A., and Wong, K.-F. (2011). Relevant knowledge helps in choosing right teacher: active query selection for ranking adaptation. In *Proc. of SIGIR*, pages 115–124.

Coleman, M. and Liau, T. L. (1975). A computer readability formula designed for machine scoring. *Journal of Applied Psychology*, 60(2):283–284.

Collins-Thompson, K., Bennett, P. N., White, R. W., de la Chica, S., and Sontag, D. (2011). Personalizing web search results by reading level. In *Proc. of CIKM*, pages 403–412.

Collins-Thompson, K. and Callan, J. (2005). Predicting reading difficulty with statistical language models. *J. Am. Soc. Inf. Sci. Technol.*, 56(13):1448–1462.

Dale, E. and Chall, J. S. (1948). A formula for predicting readability: Instructions. *Educational Research Bulletin*, 27(2):pp. 37–54.

Deerwester, S., Dumais, S. T., Landauer, T. K., Furnas, G. W., and Harshman, R. (1990). Indexing by latent semantic analysis. *Journal of the Society for Information Science*, 41(6):391–407.

Dubay, W. H. (2004). The principles of readability. *Costa Mesa, CA: Impact Information*.

Dumais, S. T. (1995). Latent semantic indexing (LSI): TREC-3 report. In *Overview of the Third Text REtrieval Conference*, pages 219–230.

Ferstl, E. C. and von Cramon, D. (2001). The role of coherence and cohesion in text comprehension: an event-related fmri study. *Cognitive Brain Research*, 11(3):325 – 340.

François, T. and Miltsakaki, E. (2012). Do NLP and machine learning improve traditional readability formulas? In *Proc. of the First Workshop on Predicting and Improving Text Readability for target reader populations*, pages 49–57, Montréal, Canada.

Freebody, P. and Anderson, R. C. (1983). Effects of vocabulary difficulty, text cohesion, and schema availability on reading comprehension. *Reading Research Quarterly*, 18(3):pp. 277–294.

Fry, E. B. (1969). The readability graph validated at primary levels. *The Reading Teacher*, 22(6):pp. 534–538.

Golub, G. and Reinsch, C. (1970). Singular value decomposition and least squares solutions. *Numerische Mathematik*, 14:403–420.

Graesser, A., McNamara, D., Louwerse, M., and Cai, Z. (2004). Coh-metrix: Analysis of text on cohesion and language. *Behavior Research Methods*, 36:193–202.

Graesser, A. C., Millis, K. K., and Zwaan, R. A. (1997). Discourse comprehension. *Annual Review of Psychology*, 48(1):163–189.

Halliday, M. A. K. and Hasan, R. (1976). *Cohesion in English (English Language)*. Longman Pub Group.

Heilman, M., Collins-Thompson, K., and Eskenazi, M. (2008). An analysis of statistical models and features for reading difficulty prediction. In *Proc. of EANL*, pages 71–79.

Jameel, S., Lam, W., Au Yeung, C.-m., and Chyan, S. (2011). An unsupervised ranking method based on a technical difficulty terrain. In *Proc. of CIKM*, pages 1989–1992.

Jameel, S., Lam, W., Qian, X., and Au Yeung, C.-m. (2012). An unsupervised technical difficulty ranking model based on conceptual terrain in the latent space. In *Proc. of JCDL*, pages 351–352.

Jones, R., Kumar, R., Pang, B., and Tomkins, A. (2007). "I know what you did last summer": query logs and user privacy. In *Proc. of CIKM*, pages 909–914.

Kanungo, T. and Orr, D. (2009). Predicting the readability of short web summaries. In *Proc. of WSDM*, pages 202–211.

Kate, R. J., Luo, X., Patwardhan, S., Franz, M., Florian, R., Mooney, R. J., Roukos, S., and Welty, C. (2010). Learning to predict readability using diverse linguistic features. In *Proc. of COLING*, pages 546–554.

Kim, J. Y., Collins-Thompson, K., Bennett, P. N., and Dumais, S. T. (2012). Characterizing web content, user interests, and search behavior by reading level and topic. In *Proc. of WSDM*, pages 213–222.

Kleinberg, J. M. (1999). Authoritative sources in a hyperlinked environment. *J. ACM*, 46(5):604–632.

Kumaran, G., Jones, R., and Madani, O. (2005). Biasing web search results for topic familiarity. In *Proc. of CIKM*, pages 271–272.

Lempel, R. and Moran, S. (2001). SALSA: The Stochastic approach for Link-structure Analysis. *ACM Trans. Inf. Syst.*, 19(2):131–160.

Leroy, G. and Endicott, J. E. (2012). Combining NLP with evidence-based methods to find text metrics related to perceived and actual text difficulty. In *Proc. of IHI*, pages 749–754.

Leroy, G., Miller, T., Rosemblat, G., and Browne, A. (2008). A balanced approach to health information evaluation: A vocabulary-based naive bayes classifier and readability formulas. *J. Am. Soc. Inf. Sci. Technol.*, 59(9):1409–1419.

Liu, X., Croft, W. B., Oh, P., and Hart, D. (2004). Automatic recognition of reading levels from user queries. In *Proc. of SIGIR*, pages 548–549.

McLaughlin, G. H. (1969). SMOG grading: A new readability formula. *Journal Of Reading*, 12(8):639–646.

McNamara, D. S., Kintsch, E., Songer, N. B., and Kintsch, W. (1996). Are good texts always better? interactions of text coherence, background knowledge, and levels of understanding in learning from text. *Cognition and Instruction*, 14(1):pp. 1–43.

Moe, A. J. (1979). Cohesion, coherence, and the comprehension of text. *Journal of Reading*, 23(1):pp. 16–20.

Morris, J. and Hirst, G. (2006). The subjectivity of lexical cohesion in text. In Shanahan, J., Qu, Y., and Wiebe, J., editors, *Computing Attitude and Affect in Text: Theory and Applications*, volume 20 of *The Information Retrieval Series*, pages 41–47. Springer Netherlands.

Nakatani, M., Jatowt, A., and Tanaka, K. (2009). Easiest-first search: Towards comprehension-based web search. In *Proc. of CIKM*, pages 2057–2060.

Nakatani, M., Jatowt, A., and Tanaka, K. (2010). Adaptive ranking of search results by considering user's comprehension. In *Proc. of the 4th International Conference on Uniquitous Information Management and Communication*, pages 27:1–27:10.

Paek, T. and Chandrasekar, R. (2005). Windows as a second language: an overview of the jargon project. In *Proc. of the First International Conference on Augmented Cognition*.

Park, Y., Byrd, R. J., and Boguraev, B. K. (2002). Automatic glossary extraction: Beyond terminology identification. In *Proc. of COLING*, pages 1–7.

Petersen, S. E. and Ostendorf, M. (2009). A machine learning approach to reading level assessment. *Comput. Speech Lang.*, 23(1):89–106.

Pitler, E. and Nenkova, A. (2008). Revisiting readability: a unified framework for predicting text quality. In *Proc. of EMNLP*, pages 186–195.

Qumsiyeh, R. and Ng, Y.-K. (2011). ReadAid: A robust and fully-automated readability assessment tool. In *Proc. of the 23rd IEEE International Conference on Tools with Artificial Intelligence*, pages 539–546.

Robertson, S., Walker, S., Jones, S., Hancock-Beaulieu, M., and Gatford, M. (1996). Okapi at trec-3. pages 109–126.

Salton, G. and Buckley, C. (1988). Term-weighting approaches in automatic text retrieval. *Inf. Process. Manage.*, 24(5):513–523.

Schwarm, S. E. and Ostendorf, M. (2005). Reading level assessment using Support Vector Machines and statistical language models. In *Proc. of ACL*, pages 523–530.

Senter, R. and Smith, E. (1967). Automated readability index. *Cincinnati University Ohio*.

Si, L. and Callan, J. (2001). A statistical model for scientific readability. In *Proc. of CIKM*, pages 574–576.

Silverstein, C., Marais, H., Henzinger, M., and Moricz, M. (1999). Analysis of a very large web search engine query log. *SIGIR Forum*, 33(1):6–12.

Tan, C., Gabrilovich, E., and Pang, B. (2012). To each his own: personalized content selection based on text comprehensibility. In *Proc. of WSDM*, pages 233–242.

Tanaka-Ishii, K., Tezuka, S., and Terada, H. (2010). Sorting texts by readability. *Comput. Linguist.*, 36(2):203–227.

Van Oosten, P. and Hoste, V. (2011). Readability annotation: Replacing the expert by the crowd. In *Proc. of the 6th Workshop on Innovative Use of NLP for Building Educational Applications*, pages 120–129.

Vakkari, P., Pennanen, M., and Serola, S. (2003). Changes of search terms and tactics while writing a research proposal a longitudinal case study. *Inf. Process. Manage.*, 39(3):445–463.

Vapnik, V. N. (1995). *The nature of statistical learning theory*. Springer-Verlag New York, Inc.

Wang, Q., Xu, J., Li, H., and Craswell, N. (2011). Regularized latent semantic indexing. In *Proc. of SIGIR*, pages 685–694.

White, R. W., Dumais, S. T., and Teevan, J. (2009). Characterizing the influence of domain expertise on web search behavior. In *Proc. of WSDM*, pages 132–141.

Yamamoto, M. and Church, K. W. (2001). Using suffix arrays to compute term frequency and document frequency for all substrings in a corpus. *Comput. Linguist.*, 27(1):1–30.

Yan, X., Lau, R. Y., Song, D., Li, X., and Ma, J. (2011). Toward a semantic granularity model for domain-specific information retrieval. *ACM Trans. Inf. Syst.*, 29(3):15:1–15:46.

Yan, X., Song, D., and Li, X. (2006). Concept-based document readability in domain specific information retrieval. In *Proc. of CIKM*, pages 540–549.

Zha, H., Marques, O., and Simon, H. (1998). Large-scale SVD and subspace-based methods for information retrieval. In Ferreira, A., Rolim, J., Simon, H., and Teng, S.-H., editors, *Solving Irregularly Structured Problems in Parallel*, volume 1457 of *Lecture Notes in Computer Science*, pages 29–42. Springer Berlin / Heidelberg.

Zhao, J. and Kan, M.-Y. (2010). Domain-specific iterative readability computation. In *Proc. of JCDL*, pages 205–214.

Using Knowledge and Constraints To Find the Best Antecedent

Prateek Jindal *Dan Roth*
Dept. of Computer Science, UIUC
201 N. Goodwin Ave., Urbana, IL - 61801
{jindal2, danr}@illinois.edu

Abstract

Coreference resolution is the problem of clustering mentions into entities and is very critical for natural language understanding. *This paper studies the problem of coreference resolution in the context of the newly emerging domain of Electronic Health Records (EHRs).* The commonly used "best-link" model for coreference resolution considers only the scores from a pairwise classifier in selecting the best antecedent. In this paper, we extend this model to include several constraints derived from surface-form of the mentions and the context in which they appear. Another major contribution of this paper is to show the use of domain-specific knowledge sources, mention parsing and clinical descriptors in deriving features which contribute to improved coreference resolution performance. We present experiments on 4 different clinical datasets illustrating that our approach outperforms a strong baseline and a state-of-the-art system by a wide margin.

Keywords: Natural Language Processing, Information Extraction, Coreference Resolution, Electronic Health Records, Knowledge Based Systems.

Proceedings of COLING 2012: Technical Papers, pages 1327–1342,
COLING 2012, Mumbai, December 2012.

1 Introduction

The HITECH (Health Information Technology for Economic and Clinical Health) Act, part of the 2009 economic stimulus package (American Recovery and Reinvestment Act) passed by the US Congress, aims at inducing more physicians to adopt *Electronic Health Records (EHRs)*. An EHR is an evolving concept defined as a systematic collection of electronic health information about individual patient. Ability to automatically extract information from EHRs lies at the heart of several applications.

This paper addresses the task of coreference resolution for EHRs. *Coreference resolution is the task of finding referring expressions in a text that refer to the same entity, i.e., finding expressions that corefer*. The set of coreferring expressions is called as a coreference chain. Consider the following text sampled from one of the EHRs in the corpus used by us:

> *This 63-year-old man had **[malignant fibrous histiocytoma of duodenum]**, discovered in 02/95. Other than **[a mass in the duodenum]**, the patient was also diagnosed with anemia. A **[leiomyosarcoma]** was resected after embolization of the splenic artery. However, **[it]** could not be completely excised; moreover **[the tumor]** metastasized to the liver as was discovered on follow up scan in 06/95.*

In the above text, all the phrases which are shown in brackets refer to the same entity and hence form a coreference chain. It is clear that identifying such coreference chains requires a lot of medical knowledge. For example, we need to know that "mass" can refer to a "malignant histiocytoma".

Most of the work on coreference resolution has focussed on the news text. Several different architectures have been proposed for coreference resolution. Recently, entity-based models for coreference resolution have been proposed. Such approaches try to directly model the entities in the text and usually involve some kind of global inference and tend to be quite complex. However, most of the best results on coreference resolution were achieved with simpler architectures which use a pairwise classifier between mentions and a decoding strategy like "closest-first" or "best-link" to first find the best antecedent for every mention. This step is then followed up by an inference procedure in which coreference chains are formed (Chang et al., 2011; Pradhan et al., 2011).

In this paper, we extend the "best-link" model to include several constraints derived from surface-form of the mentions and the context in which they appear. Another contribution of this paper is to show the use of domain-specific knowledge sources (like UMLS[1], MetaMap), mention parsing and clinical descriptors (obtained from medical ontologies) in deriving the features which are helpful for coreference resolution. In clinical Information Extraction (IE), researchers often map clinical text to UMLS concepts (Zheng et al., 2012; Rink et al., 2012). But such mapping alone doesn't allow an IE system to exploit the useful information contained in the parent trees of the concepts. Clinical descriptors designed by us overcome this limitation. We use two medical ontologies, MeSH[2] and SNOMED CT[3] to design our descriptors.

We conducted experiments on four different clinical datasets. Our results show that knowledge sources help in improving the recall and constraints help to increase the precision of the system. Knowledge and constraints used by us helped us to achieve significant performance improvements over a strong baseline derived from existing state-of-the-art approaches.

[1] http://www.nlm.nih.gov/research/umls/
[2] http://www.nlm.nih.gov/mesh/meshhome.html
[3] http://www.ihtsdo.org/snomed-ct/

To summarize, the key contributions of our paper are as follows:

- This paper studies coreference resolution on the new and important domain of EHRs.
- This paper presents different knowledge sources which would be useful for Information Extraction in medical and clinical domains. We use medical ontologies (MeSH, SNOMED CT) to get clinical descriptors which encode useful information contained in the parent trees of the concepts.
- We propose a rich local model to find the best antecedent.
- We use *mention parsing* to obtain a semantic representation of the mentions. Similar technique can also be used for other domains.
- Our system outperforms a strong baseline on four different clinical datasets.

2 Task Description

Coreference resolution aims at clustering together textual mentions within a single document based on underlying referent entities. For our experiments, we used the datasets provided by i2b2 team as part of coreference challenge. *We use the same problem definition as was specified in the i2b2 coreference challenge.* Mentions have already been identified and classified into 4 types : test (TEST), treatment (TRE), problem (PROB) and pronoun (PRON). Coreference relation can exist only within the mentions of same type. However, PRON mentions can corefer with any other mention. Given the entity mentions along with the types, the aim is to build coreference chains for the first 3 types: TEST, TRE and PROB. Since PRON mentions can corefer with the mentions of other types, there are no separate pronoun (PRON) chains.

3 Coreference Model

In this paper, we view coreference resolution as a graph problem: Given a set of mentions and their context as nodes, generate a set of edges such that any two mentions that belong in the same equivalence class are connected by some path in the graph. We construct this entity-mention graph by finding out the best antecedent of each given mention (anaphor) such that the antecedent belongs to the same equivalence class as the anaphor. The "Best-Link" strategy (Ng and Cardie, 2002; Bengtson and Roth, 2008; Chang et al., 2011) for selecting the antecedent of a mention chooses as the antecedent that candidate which gets the maximum score according to a pairwise coreference function pc. We extend the "Best-Link" strategy by including several constraints in its objective function as shown below.

3.1 Decision Model: Constrained Best-Link

Given a document d and a pairwise coreference scoring function pc that maps an ordered pair of mentions to a value indicating the probability that they are coreferential, we generate a coreference graph G_d according to the following decision model:

For each mention m_i in document d, let B_{m_i} be the set of mentions appearing before m_i in d. Thus, $B_{m_i} = \{m_1, m_2, ..., m_{i-1}\}$. Let a be the highest scoring antecedent. Then, we have:

$$
\begin{aligned}
a &= \underset{m_j \in B_{m_i}}{\arg\max} \quad score_i(m_j) \\
&= \underset{m_j \in B_{m_i}}{\arg\max} \quad k_1 \cdot pc(m_j, m_i) - d(m_j, m_i) + \sum_{l=1}^{L} C_l(m_j, m_i)
\end{aligned} \tag{1}
$$

In the above equation, $d(m_j, m_i)$ refers to the normalized distance between m_j and m_i which takes values between 0 and 1. In equation (1), C_l refers to l^{th} constraint and is defined as follows (for all values of l):

$$C_l(m_j, m_i) = \begin{cases} 0 & \text{if } l^{th} \text{ constraint is satisfied} \\ -p_l & \text{otherwise} \end{cases} \quad (2)$$

If $score_i(a)$ is greater than a threshold δ, then we add the edge (a, m_i) to the coreference graph G_d. Threshold parameter δ is chosen to be $\frac{k_1}{2}$. Value of $pc(m_j, m_i)$ lies between 0 and 1. The value of k_1 is chosen to be sufficiently greater than 1 so that the pairwise classifier is given preference over the distance term in choosing the best antecedent. But if the pc values of any two candidates are almost similar, then the antecedent which is closer to the anaphor gets the higher score because of the distance term in Equation (1). Thus, our decision model combines the advantages of both "best-link" and "closest-first" models which are generally used for coreference resolution. Setting $k_1 = \infty$ and $L = 0$ reduces our model to the standard "best-link" decision model.

p_l is the penalty associated with the l^{th} constraint. Thus, different constraints can have different penalties. Higher the penalty associated with the constraint, the stronger it is enforced. If $0 < p_l < \frac{k_1}{2}$, then the constraint is soft because violation of such constraint by a mention pair doesn't necessarily rule it out. But if $p_l > \frac{k_1}{2}$, then the constraint becomes hard.

The resulting graph produced by the decoding technique mentioned above contains connected components, each representing one equivalence class, with all the mentions in the component referring to the same entity. Equivalence classes are determined by taking the transitive closure of all the links.

3.2 Pairwise Coreference Function

We train 4 classifiers, one each for TEST, TRE, PROB and PRON classes. Each of these classifiers takes as input an ordered pair of mentions (a, m) such that a precedes m in the document, and produces as output a value that is interpreted as the conditional probability that a and m belong in the same equivalence class. For any mention-pair (a, m), the classifier is chosen based on the type of mention m.

For each mention m we select from m's equivalence class the closest preceding mention a and present the pair (a, m) as a positive training example to the classifier which corresponds to the type of mention m. For each m, we generate negative examples (a, m) for all mentions a that precede m and are not in the same equivalence class.

We learn the pairwise classifiers using LIBSVM package (Chang and Lin, 2011).

4 Baseline

In this section, we describe the baseline system used by us. We designed the baseline system based on the existing state-of-the-art coreference systems which use pairwise models (Bengtson and Roth 2008; Haghighi and Klein 2009). Baseline system uses the coreference model as described in the previous section. However, there are no constraints in the baseline system. The features used for training the pairwise classifier have been described below. All the features used by us take only two values: 1 (if the feature is active) or 0 (if the feature is not active).

4.1 Lexical Features

Lexical features indicate whether two strings share some property. These features are listed below:

- Both the mentions have identical surface forms (i.e. $extent_{m_i} == extent_{m_j}$).
- Surface form of one of the mentions is a proper substring of that of another.
- Both the mentions share the same head word.

4.2 Syntactic Features

We check for the presence of several syntactic constructs among the mentions and generate the following features which tell whether or not the given mention pair satisfies the constructs:

- *Apposition*: Two noun phrases (NPs) are appositive when they are placed side-by-side with one element serving to define or modify the other e.g. *In a recent examination, the patient was diagnosed with **[medulloblastoma]**, **[a malignant brain tumor]**.*
- *Predicate Nominative*: The predicate nominative is the noun following a linking verb that restates or stands for the subject e.g. ***[Coronary Arteriosclerosis]** is a **[heart disease]** which happens when the coronary arteries become narrowed.*
- *Relative Pronoun*: It is a pronoun that modifies the head of the antecedent NP e.g. *After discussion , an **[abdominal CT scan]** was obtained **[which]** revealed diffuse metastatic lesions of the ribs .*

4.3 Semantic Features

Some of the coreferential mention pairs have similar but not identical heads. To find out whether any two words are similar or not requires semantic knowledge. Wordnet has been extensively used as a source of semantic knowledge for general English text. We generate the following two features from Wordnet:

- *Wordnet-head-match*: We get the synsets of heads of both the mentions and see whether the heads share any common synset. For example, the words *hemorrhage* and *bleeding* share the same synset which refers to the *flow of blood from a ruptured blood vessel*.
- *Wordnet-head-hypernyms-match*: Some closely related words (like *epistaxis* and *hemorrhage*) do not share any common synset. However, if we consider the parents (or hypernyms) of the synsets of such words in the Wordnet hierarchy, we can see that the two words are similar. We take only the immediate hypernyms of the synsets. Inclusion of hypernyms which are more than 1 level above the synsets of the words leads to over-generalization. For example, we may get that *nausea* and *anemia* are coreferent which is actually not true.

4.4 Distance-Based Features

We used the following distance-based features:

- *Adjacent-Mentions*: This feature is active if there is no intervening mention between the given mentions which has the same type as the mentions under consideration.
- *Distant-Mentions*: This feature is active if the two mentions are separated by more than 2 sentences.

5 Using Domain-Specific Knowledge

One of the major limitations of the baseline system is that it lacks domain-specific knowledge. In medical terminology, same concept can be represented in several different ways. For example, “headache”, “cranial pain” and “cephalgia” all refer to the same concept. Similarly, “Atrial Fibrillation”, “AF” and “AFib” also refer to the same concept. The baseline system is not sufficient to

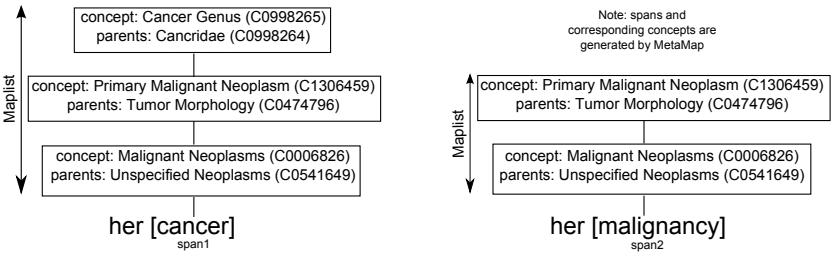


Figure 1: This figure shows the UMLS mappings for two mentions "her cancer" and "her malignancy". The terms "cancer" and "malignancy" have at least one common concept. Our matching procedure based on UMLS correctly predicts the given mentions to be coreferent.

address the ambiguity and variability that exists in medical terminology. To improve the performance of coreference resolution, we extended the baseline system by incorporating domain-specific knowledge into it.

5.1 UMLS and MetaMap

The UMLS (UMLS, 2012), or Unified Medical Language System, is a set of files and software that brings together many health and biomedical vocabularies. MetaMap (Aronson and Lang, 2010) is a configurable program which maps biomedical text to the UMLS Metathesaurus. We use the mapping provided by MetaMap to represent the mentions in a standard way which allows for effective matching of the mentions. We find the parents of concepts using the Web Service provided by UTS (UTS, 2012) (UMLS Technology Services).

Matching Mentions Using UMLS: We would refer to the surface forms of the two mentions by $s1$ and $s2$. First, we remove the stopwords from the given strings and then process the resulting strings using MetaMap and thus, get the mappings of the mentions to UMLS concepts. Next, we check whether any two spans (given by MetaMap) of $s1$ and $s2$ are equivalent. Two spans are considered equivalent if they share the same UMLS concepts (or parents of UMLS concepts). Whenever we find two equivalent spans, we remove them from $s1$ and $s2$. Finally, we check whether the resulting strings $s1$ and $s2$ match trivially. Two strings match trivially if they are identical or one of them is a substring of the other.

Consider Figure 1 for an example. This figure shows the UMLS mappings for two mentions "her cancer" and "her malignancy". "her" is considered as a stopword and is first removed from both the strings. Since the two spans "cancer" and "malignancy" share same UMLS concepts, they are equivalent. So, we remove "cancer" and "malignancy" from the two strings. The resulting strings are both empty and are considered to be matching.

Features Derived: Based on the matching procedure described above, we derive the following two features:

- *UMLS-Match*: In this feature, we do not consider the parents of the concepts during the matching
- *UMLS-Match-Parents*: In this feature, parents of the concepts are also considered during the matching

5.2 Mention Parsing

We parsed the mentions to extract the components like Modifiers, Body Parts and Anatomical Terms of location (ATs). We did not require exact match for these extracted components. We just specified that these components should not be incompatible with each other. The remaining portions of the surface forms of mentions were canonicalized and matching procedure described in Section 5.1 was used to determine whether they matched. Figure 2 shows an example where the structures obtained by parsing the two mentions are matching to one another.

Canonicalization referred to above involves the following two steps:

- *Expanding the abbreviations*: Clinical narratives use a lot of abbreviations. A few examples are: mri (magnetic resonance imaging), copd (chronic obstructive pulmonary disease) etc. Abbreviations were expanded to their full forms as a normalization step. We collected abbreviations from several sources like training data, Wikipedia[4], Medilexicon[5] etc. For ambiguous abbreviations, we considered all possible expansions. If a match was found using any of the expansions, then coreference pair was considered valid.
- *Converting Hyponyms to Hypernyms*: During preprocessing, we converted some of the common hyponyms to the corresponding hypernyms. Examples of such conversions are: chemotherapy → therapy, hemicolectomy → colectomy. Such conversions were found to be very helpful because it is a common practice in clinical documents to refer to some of the problems and treatments introduced earlier in the document with their more general names later on. These hyponym-hypernym pairs were collected from the unannotated training data in an unsupervised setting.

Features Derived: Following feature was derived from mention parsing:

- *Mention-Parsing*: This feature is true for the mention pairs which match according to the mention parsing procedure described above.

5.3 Clinical Descriptors

MeSH[6] (Medical Subject Headings) is the National Library of Medicine's controlled vocabulary thesaurus. It consists of sets of terms in a hierarchical structure that permits searching at various levels of specificity. We obtain MeSH descriptor for a concept in the following way:

1. First of all, we get all the paths from the concept to the root of MeSH hierarchy. In general, there can be more than 1 paths.
2. Then we construct one list which consists of the top 4 parents of all the paths obtained in Step 1.
3. The list obtained in step 2 is pruned where more preference is given to those parents which appear more frequently.
4. The final list obtained in step 3 is the MeSH descriptor of the concept.

Similar procedure is used to obtain the SNOMED CT[7] descriptor of a concept. SNOMED CT (SNOMED Clinical Terms) is yet another medical ontology which consists of the most comprehensive, multilingual clinical healthcare terminology in the world. SNOMED CT is owned,

[4] http://en.wikipedia.org/wiki/List_of_medical_abbreviations
[5] http://www.medilexicon.com/medicalabbreviations.php
[6] http://www.nlm.nih.gov/mesh/meshhome.html
[7] http://www.ihtsdo.org/snomed-ct/

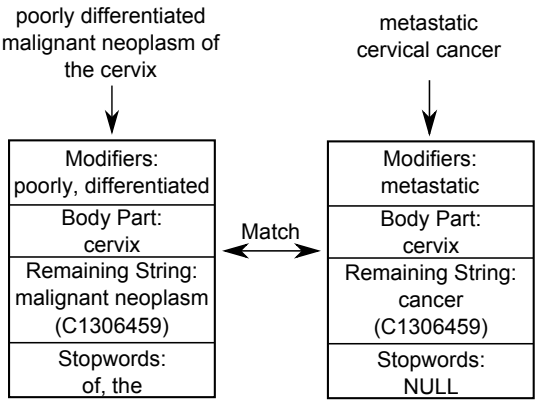


Figure 2: This figure shows the structures obtained by the mention parsing of two mentions shown on the top of the figure. Since the two structures match one another, we predict the two mentions to be coreferent.

maintained and distributed by the International Health Terminology Standard Development Organisation (IHTSDO). Figure 3 and Figure 4 show two different paths in MeSH and SNOMED CT parent trees for the same concept "Myocardial Infarction". We found that, in general, SNOMED CT gives much more paths (from concept to root of hierarchy) than MeSH. Some concepts in SNOMED CT have more than 300 possible paths to the root of hierarchy.

Features Derived: Based on the clinical descriptors described above, we derive the following two features:

- *MeSH-Match*: This feature is active if the Mesh descriptors of two concepts are the same
- *SNOMEDCT-Match*: This feature is active if the SNOMED CT descriptors of two concepts are the same

6 Description of Constraints

Constraints are used to model domain knowledge and they refer to those conditions which, if not satisfied, strongly indicate that the given mention pair is not coreferential. Features, on the other hand, can be more vague and don't necessarily provide such a strong clue. Constraints are applied only during the inference phase and not the learning phase. So, constraints can be added or removed without having to retrain the classifiers. Even if a particular constraint is not seen very often in the training data, it can still be very useful at the test time if the testing data contains cases where the constraint is applicable. This is a clear advantage of modeling constraints separately from the features. We divide the constraints in two categories depending on whether the constraint is derived from the surface form of the mentions or from the context in which the mentions occur. Constraints used by us are described in the following subsections. These constraints were obtained by the manual examination of small portion of training data.

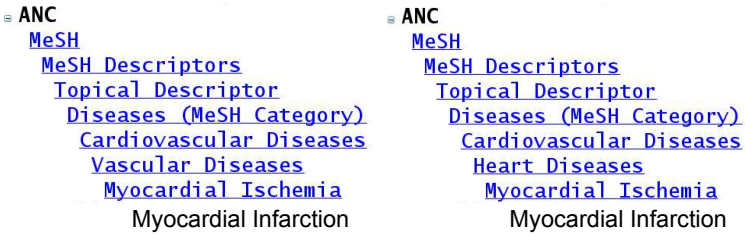


Figure 3: Figure showing two different paths in MeSH parent tree for the concept "Myocardial Infarction"

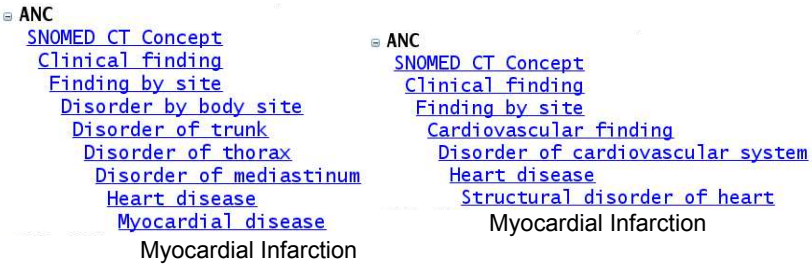


Figure 4: Figure showing two different paths in SNOMED CT parent tree for the concept "Myocardial Infarction"

6.1 Surface Form Constraints

Following surface form constraints were used by us:

- *Length Constraint*: Surface form of both the mentions must be at least 2 characters long.
- *Modifier Constraint*: Mentions should not have incompatible modifiers e.g. "small/large"
- *Body Parts Constraint*: If body parts (like chest, arm, head) are specified, they should not be incompatible.
- *Anatomical Terms Constraint*: If anatomical terms[8] (like proximal, anterior, dorsal) are specified, they should not be incompatible.
- *Popular Head Constraint*: Certain head words like "disease" occur very commonly in the dataset. Mentions which have same popular head are considered coreferential only if the classifier predicts the mentions to be coreferential even after removing the heads from the mentions.
- *Number Constraint*: Two mentions must agree in number.
- *Temporal Constraint*: If only one of the mentions contains the word "follow-up" or "repeat", then the mention pair is not considered coreferential because the two mentions refer to tests or treatments which have been done at different times.

[8] http://en.wikipedia.org/wiki/Anatomical_terms_of_location

6.2 Contextual Constraints

Following contextual constraints were used by us:

- *Family History*: If the left context of any mention (in a window of size 4) contains the phrase "family history", then the mention pair is not considered coreferential because one of the mentions refers to some family member of the patient and not the patient himself. Window size was determined using cross-validation on the training set.
- *Negation Constraint*: None of the mentions should be present in a negated form.
- *PRN Constraint*: Problem mentions which have "p.r.n." as the prefix can't participate in coreference relation because such mentions refer to hypothetical problems and not the real problems. For example, "p.r.n. headache" means "if the headache arises ...".
- *TEST Constraint*: We observed from the documents in the training data that the TEST mentions which appear under the heading "LABORATORY DATA" generally don't participate in coreference.

Other than the above mentioned constraints, following additional constraint was used to disallow coreference chains beginning with pronouns.

- In Equation (1), if m_j is a pronoun, then there must exist some mention m_k with $k < j$ such that m_k is a valid antecedent of m_j.

7 Experimental Setup

Datasets: For our experiments, we used the coreference datasets made available by i2b2 team as part of 2011 i2b2 challenge. The datasets consist of EHRs from three different organizations: Partners HealthCare (Part), Beth Israel Deaconess Medical Center (Beth) and University of Pittsburgh (Pit). The data from University of Pittsburgh is divided into 2 parts, namely Discharge and Progress records. All records have been fully de-identified and manually annotated for coreference. This gave us a total of 4 datasets. We would refer to these datasets as *Part*, *Beth*, *PitD* and *PitP* in the following discussion.

The total number of documents in the training set of *Part*, *Beth*, *PitD* and *PitP* are 136, 115, 119 and 122 respectively. Test set of *Part*, *Beth*, *PitD* and *PitP* contains 94, 79, 77 and 72 documents respectively. For more information about the datasets, please refer to Uzuner et al. (Uzuner et al., 2012). We used B-cubed (Bagga and Baldwin, 1998), MUC (Vilain et al., 1995) and CEAF (Luo, 2005) as the evaluation metrics in our experiments.

Choice of Parameters: We use cross-validation on the training data to determine the system parameters. In Equation (1), we set $k_1 = 100$. With this choice of k_1, distance term becomes significant only if the scores given by pairwise classifier for different mention pairs differ by less than 0.01. Since all our constraints are important to be enforced, we chose $p_l = 100$ in Equation (2) for all values of l. This choice of penalty parameters makes all the constraints hard.

8 Results

Table 1 compares the performance of four systems (1) Baseline (B), (2) Baseline + Knowledge (BK), (3) Baseline + Knowledge + Constraints (BKC) and (4) Baseline + Constraints (BC). We compare the performance of these systems for *Test*, *Treatment* and *Problem* categories on 4 different datasets, namely, *Part*, *Beth*, *PitD* and *PitP*. Table 1 reports precision (P), recall (R) and F1 scores for MUC evaluation metric. For B-cubed and CEAF Evaluation metrics, we only show the F1 scores because of space limitation. Please note that there are no separate scores for PRON category because there

are no separate PRON chains. PRON mentions are included within the TEST, TRE and PROB chains. Results shown in Table 1 are quite interesting and are explained below.

It is interesting to note that adding knowledge to the system always leads to higher recall values. On the other hand, addition of constraints always leads to higher precision values. Next, we note that different metrics behave differently in evaluating the performance of the systems. B-cubed metric gives higher F1 scores that CEAF metric which in turn gives higher F1 scores than MUC metric. This is because of the presence of large number of singletons in the corpora. B-cubed metric highly awards the correct prediction of singletons. MUC, on the other hand, is totally insensitive to singletons. CEAF is intermediate between B-cubed and MUC as far as singletons are concerned.

Next, we note the following major points about each category of mentions. For statistical significance tests, Bootstrap Resampling Test (Koehn, 2004) was used at $p = 0.05$.

1. ***Test***: For *Test* mentions, the best configuration is Baseline+Constraints (BC). For MUC metric, both BKC and BC performed the best for 2 corpora each. However, for B-cubed and CEAF evaluation metrics, BC performed the best for all the corpora. Hence, overall, we can say that BC is the best configuration for *Test* mentions. This is because of the fact that coreference for *Test* mentions (like "his ct scan", "a mammogram" etc.) can generally be easily predicted simply by looking at the surface forms. Also, many of the *Test* coreference chains are quite short with only 2-3 mentions which occur close to one another. So, knowledge is not so helpful for *Test* mentions.
2. ***Treatment***: For *Treatment* mentions, the best configuration is Baseline+Knowledge (BK). This is clearly evident from MUC metric. Only for *Beth* corpus, BKC performed better than BK but the difference is not statistically significant (67.8 vs 67.9). For B-cubed and CEAF evaluation metrics, the maximum F1 scores for *Treatment* category are quite close to Baseline scores and hence, the results are not statistically significant. Thus, B-cubed and CEAF metrics do not help much in predicting which system is better for *Treatment* mentions.
3. ***Problem***: For *Problem* mentions, the best system is Baseline+Knowledge+Constraints (BKC). This is clearly evident from B-cubed and CEAF Evaluation metrics. For MUC evaluation metric, BK performed better than BKC for 2 corpora. However, the difference in such cases is not statistically significant (69.1 vs 69.3 and 58.3 vs 58.4). Thus, we see that both, knowledge and constraints, benefit *Problem* mentions. This is due to the fact that *Problem* mentions, in general are quite long and complicated. *Problem* mentions generally occur with modifiers and have variegated surface forms. For example, "the patient's low potassium level" is coreferrent with "postoperatively hypokalemia".

Finally, in Table 2, we show the comparison of our system with a state-of-the-art system, Ware et al. (Ware et al., 2012), which used same test settings as ours. The numbers reported in Table 2 refer to the unweighted average of Bcubed, MUC and CEAF F1 scores computed across all the 4 corpora. We chose unweighted average for comparison because it was the official metric of i2b2 2011 shared task on coreference. We see from this table that our system consistently outperformed Ware et al.'s system for all categories of mentions.

9 Error Analysis

Table 3 shows the number of pairwise errors produced by our system on a portion of the test dataset. Rows indicate types of antecedent; columns are mention types. Each cell shows the number of precision/recall errors for that configuration. The total number of gold links is 2,252. We see that our system makes more precision errors than the recall errors. This is also confirmed by the results

	MUC Evaluation											
	B			BK			BKC			BC		
	P	**R**	**F1**	**P**	**R**	**F1**	**P**	**R**	**F1**	**P**	**R**	**F1**
						Test						
Part	30.4	84.8	44.8	29.3	88.8	44.0	32.4	85.8	47.0	33.8	83.8	**48.2**
Beth	13.2	70.2	22.2	13.9	77.8	23.6	16.3	71.7	**26.5**	16.0	67.5	25.9
PitD	29.4	82.7	43.4	28.9	86.3	43.3	30.4	81.0	44.2	30.9	79.2	**44.5**
PitP	25.1	79.3	38.1	25.7	86.0	39.5	28.6	80.2	**42.2**	27.4	74.4	40.1
						Treatment						
Part	58.1	81.4	67.8	57.7	86.0	**69.0**	58.1	83.1	68.4	58.1	79.4	67.1
Beth	57.9	79.2	66.9	57.4	82.7	67.8	58.5	81.0	**67.9**	58.5	78.0	66.9
PitD	51.0	72.1	59.7	51.7	77.4	**62.0**	52.4	74.7	61.6	51.3	70.9	59.5
PitP	55.8	72.2	63.0	55.4	76.5	**64.2**	55.9	74.7	64.0	56.0	71.2	62.7
						Problem						
Part	54.6	72.8	62.4	55.0	80.8	65.5	57.8	77.4	**66.2**	56.6	70.8	62.9
Beth	60.0	70.1	64.6	60.1	81.8	**69.3**	62.2	77.7	69.1	61.1	67.7	64.3
PitD	54.0	75.9	63.1	55.3	85.3	67.1	57.3	81.8	**67.4**	55.5	73.9	63.4
PitP	45.5	73.1	56.1	46.0	80.1	**58.4**	46.5	78.2	58.3	45.8	71.7	55.9

(a) MUC Evaluation

	B-Cubed Evaluation				CEAF Evaluation			
	B	BK	BKC	BC	B	BK	BKC	BC
	F1	**F1**	**F1**	**F1**	**F1**	**F1**	**F1**	**F1**
				Test				
Part	93.0	92.5	93.4	**93.8**	84.5	82.7	85.9	**87.4**
Beth	89.8	89.4	90.7	**90.9**	66.9	65.1	73.2	**74.4**
PitD	88.4	87.7	88.9	**89.3**	76.7	75.0	78.7	**79.8**
PitP	92.9	92.6	**93.2**	**93.2**	83.3	82.4	85.4	**85.6**
				Treatment				
Part	**92.8**	**92.8**	**92.8**	**92.8**	85.9	85.2	85.7	**86.0**
Beth	**92.5**	92.4	92.4	**92.5**	81.8	81.2	81.9	**82.1**
PitD	91.2	**91.5**	91.4	91.2	84.8	84.7	**84.9**	**84.9**
PitP	91.6	91.6	**91.7**	91.6	84.3	84.0	84.3	**84.4**
				Problem				
Part	92.3	92.3	**92.9**	92.6	84.9	84.8	**86.6**	86.1
Beth	92.2	92.3	**92.7**	92.3	83.7	83.8	**85.2**	84.3
PitD	90.5	90.6	**91.1**	90.8	83.1	83.6	**85.6**	84.5
PitP	93.7	93.6	93.8	**93.9**	85.3	85.1	**85.6**	**85.6**

(b) B-cubed and CEAF Evaluation

Table 1: This table compares the performance of four systems: (1) Baseline (B), (2) Baseline + Knowledge (BK), (3) Baseline + Knowledge + Constraints (BKC) and (4) Baseline + Constraints (BC). Part (a) of table reports Precision, Recall and F1 scores for MUC evaluation metric for TEST, TRE and PROB categories on 4 different datasets. Part (b) shows the F1 scores for B-cubed and CEAF evaluation metrics. For detailed discussion of the results, please refer to Section 8.

	Avg of B^3, MUC, CEAF F1		
	Test	**Treatment**	**Problem**
Ware et al.	68.4	79.4	80.8
This Paper	**69.1**	**80.7**	**81.6**

Table 2: This table shows the comparison of system presented in this paper with a state-of-the-art system, Ware et al. The numbers refer to the unweighted average of Bcubed, MUC and CEAF F1 scores computed across all the 4 corpora.

	TEST	TRE	PROB	PRON
TEST	92/60	-	-	22/12
TRE	-	187/186	-	57/14
PROB	-	-	320/293	70/21
PRON	32/10	47/13	97/37	20/7
Total	124/70	234/199	417/330	169/54

Table 3: This table shows the number of pairwise errors produced by our system on a portion of the test dataset. Rows indicate types of antecedent; columns are mention types. Each cell shows the number of precision/recall errors for that configuration. The total number of gold links is 2,252.

in Table 1. Error analysis of our system reveals that its precision can be improved by analyzing the context of the mentions more deeply. For example, it would be helpful to know the time (if mentioned) at which a particular test was conducted. It would be also beneficial to know whether a particular problem is mentioned in relation to the patient or one of his/her family members. On inspection, we found that our system made recall errors only on very difficult mention pairs. Predicting coreference relation among such mention pairs requires a lot of reasoning.

10 Related Work

For news text, several different architectures have been proposed for coreference resolution. Systems have been developed which allow for entity-level features or features over sets of noun phrases (Culotta et al., 2007). Such methods generally involve some kind of global inference which is difficult to implement and may also be intractable. Research (Finkel and Manning, 2008; Haghighi and Klein, 2007; Poon and Domingos, 2008) has also been carried out to explore how to reconcile pairwise decisions to form coherent clusters.

However, pairwise models with rich knowledge base have been shown to be very successful in both supervised and unsupervised setups (Bengtson and Roth, 2008; Haghighi and Klein, 2009). An important step in such models is to find the antecedent for each mention. For selecting the antecedent, "best-link" decoding strategy has been shown to give better results than "closest-first". In this paper, we extended the "best-link" strategy used by researchers by incorporating other factors like distance between mentions, several constraints etc. during the inference step.

There has been an increasing interest in knowledge-rich coreference resolution (Uryupina et al., 2011; Rahman and Ng, 2011; Bryl et al., 2010; Ng, 2010; Ponzetto and Strube, 2006; Bean and Riloff, 2004). Wikipedia is one of the most common knowledge resources that have been used by researchers. However, Wikipedia is not very good for clinical text because it doesn't have sufficient coverage of medical terms and also lacks precision. *In this paper, we used domain-specific knowledge sources like UMLS, MeSH and SNOMED CT to improve coreference resolution in clinical*

domain.

One of the earliest works in coreference resolution in clinical domain is that of Zheng et al. (Zheng et al., 2011). In this work, authors review recent advances in general purpose coreference resolution to lay the foundation for methodologies in the clinical domain. Later, Zheng et al. (Zheng et al., 2012) describe a simple pairwise classification technique for coreference resolution in clinical domain and got an overall B-cubed score of 0.69 and MUC score of 0.35. Bodnari et al. (Bodnari et al., 2012) and Jindal et al. (Jindal and Roth, 2012) also use a pairwise classification technique for clinical coreference resolution and use UMLS to get some of their semantic features. However, they don't use the concepts' parents information available in UMLS. Uzuner et al. (Uzuner et al., 2012) give a brief overview of several systems which participated in 2012 i2b2 coreference challenge. Most of the systems submitted in the challenge were rule-based. Rink et al. (Rink et al., 2012) used a multi-pass sieve architecture which is similar to the one developed by Raghunathan et al. (Raghunathan et al., 2010). Xu et al. (Xu et al., 2012) developed an effective strategy for pronoun resolution where they first determined the type of the pronoun and then chose the closest preceding concept of the same type as the antecedent. All these works assumed mentions' boundaries (along with their types) to be given just like ours.

Conclusion

Electronic Health Records are becoming increasingly important and their automatic analysis lies at the heart of several applications. This paper presented a system for coreference resolution for EHRs. In this paper, we proposed a rich model for selecting the best antecedent which involves inference using pairwise classifier scores and several constraints derived from surface-form of the mentions and the context in which they appear. We also showed the importance of domain-specific knowledge sources and clinical descriptors for achieving good performance in coreference resolution. While the knowledge sources used by us helped to improve the recall, constraints were helpful to increase the precision of system. Our experimental results show that different mention types benefit to different extent from knowledge and constraints. Our system consistently outperformed a strong baseline and a state-of-the-art system on four different datasets.

Acknowledgments

The authors would like to thank Ozlem Uzuner, Andreea Bodnari and Brett South for organizing the 2011 i2b2 challenge and providing the data used in these experiments. We would also like to thank the anonymous reviewers for their valuable suggestions. This research was supported by Grant HHS 90TR0003/01. Its contents are solely the responsibility of the authors and do not necessarily represent the official views of the HHS or the US government.

References

Aronson, A. and Lang, F. (2010). An overview of metamap: historical perspective and recent advances. *Journal of the American Medical Informatics Association*, 17(3):229.

Bagga, A. and Baldwin, B. (1998). Algorithms for scoring coreference chains. In *In LREC Workshop on Linguistics Coreference*, pages 563–566. Citeseer.

Bean, D. and Riloff, E. (2004). Unsupervised learning of contextual role knowledge for coreference resolution. In *Proc. of HLT/NAACL*, pages 297–304.

Bengtson, E. and Roth, D. (2008). Understanding the value of features for coreference resolution. In *Proceedings of EMNLP*, pages 294–303. ACL.

Bodnari, A., Szolovits, P., and Uzuner, Ö. (2012). Mcores: a system for noun phrase coreference resolution for clinical records. *J Am Med Info Assoc*, 19(5):906–912.

Bryl, V., Giuliano, C., Serafini, L., and Tymoshenko, K. (2010). Using background knowledge to support coreference resolution. In *Proceedings of ECAI 2010, August*.

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

Chang, K., Samdani, R., Rozovskaya, A., Rizzolo, N., Sammons, M., and Roth, D. (2011). Inference protocols for coreference resolution. In *CoNLL Shared Task*, pages 40–44. ACL.

Culotta, A., Wick, M., Hall, R., and McCallum, A. (2007). First-order probabilistic models for coreference resolution. In *Proceedings of NAACL HLT*, pages 81–88.

Finkel, J. and Manning, C. (2008). Enforcing transitivity in coreference resolution. In *Proceedings of 46th ACL-HLT Short Papers*, pages 45–48. ACL.

Haghighi, A. and Klein, D. (2007). Unsupervised coreference resolution in a nonparametric bayesian model. In *Annual meeting-Association for Computational Linguistics*, page 848.

Haghighi, A. and Klein, D. (2009). Simple coreference resolution with rich syntactic and semantic features. In *Proceedings of EMNLP Volume 3*, pages 1152–1161. ACL.

Jindal, P. and Roth, D. (2012). Using domain knowledge and domain-inspired discourse model for coreference resolution for clinical narratives. *Journal of the American Medical Informatics Association*.

Koehn, P. (2004). Statistical significance tests for machine translation evaluation. In *Proceedings of EMNLP*, volume 4, pages 388–395.

Luo, X. (2005). On coreference resolution performance metrics. In *Proceedings of HLT and EMNLP*, pages 25–32. ACL.

Ng, V. (2010). Supervised noun phrase coreference research: The first fifteen years. In *Proceedings of 48th ACL*, pages 1396–1411. ACL.

Ng, V. and Cardie, C. (2002). Improving machine learning approaches to coreference resolution. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, pages 104–111. Association for Computational Linguistics.

Ponzetto, S. and Strube, M. (2006). Exploiting semantic role labeling, wordnet and wikipedia for coreference resolution. In *Proceedings of the NAACL*, pages 192–199. ACL.

Poon, H. and Domingos, P. (2008). Joint unsupervised coreference resolution with markov logic. In *Proceedings of EMNLP*, pages 650–659. ACL.

Pradhan, S., Ramshaw, L., Marcus, M., Palmer, M., Weischedel, R., and Xue, N. (2011). Conll-2011 shared task: Modeling unrestricted coreference in ontonotes. *CoNLL 2011*, page 1.

Raghunathan, K., Lee, H., Rangarajan, S., Chambers, N., Surdeanu, M., Jurafsky, D., and Manning, C. (2010). A multi-pass sieve for coreference resolution. In *Proceedings of EMNLP*, pages 492–501. ACL.

Rahman, A. and Ng, V. (2011). Coreference resolution with world knowledge. In *Proceedings of the 49th ACL-HLT Volume 1*, pages 814–824. ACL.

Rink, B., Roberts, K., and Harabagiu, S. (2012). A supervised framework for resolving coreference in clinical records. *Journal of the American Medical Informatics Association*, 19(5):875–882.

UMLS (2012). http://www.nlm.nih.gov/research/umls/ (accessed aug 25, 2012).

Uryupina, O., Poesio, M., Giuliano, C., and Tymoshenko, K. (2011). Disambiguation and filtering methods in using web knowledge for coreference resolution. In *Proceedings of the 24th International Florida Artificial Intelligence Research Society Conference*.

UTS (2012). Umls technology services. https://uts.nlm.nih.gov/home.html (accessed aug 25, 2012).

Uzuner, O., Bodnari, A., Shen, S., Forbush, T., Pestian, J., and South, B. (2012). Evaluating the state of the art in coreference resolution for electronic medical records. *Journal of the American Medical Informatics Association*.

Vilain, M., Burger, J., Aberdeen, J., Connolly, D., and Hirschman, L. (1995). A model-theoretic coreference scoring scheme. In *Proceedings of the 6th MUC*, pages 45–52. ACL.

Ware, H., Mullett, C., Jagannathan, V., and El-Rawas, O. (2012). Machine learning-based coreference resolution of concepts in clinical documents. *J Am Med Info Assoc*, 19(5):883–887.

Xu, Y., Liu, J., Wu, J., Wang, Y., Tu, Z., Sun, J., Tsujii, J., Eric, I., and Chang, C. (2012). A classification approach to coreference in discharge summaries: 2011 i2b2 challenge. *Journal of the American Medical Informatics Association*, 19(5):897–905.

Zheng, J., Chapman, W., Crowley, R., and Savova, G. (2011). Coreference resolution: A review of general methodologies and applications in the clinical domain. *JBI*.

Zheng, J., Chapman, W., Miller, T., Lin, C., Crowley, R., and Savova, G. (2012). A system for coreference resolution for the clinical narrative. *J Am Med Info Assoc*.

Towards a Generic and Flexible Citation Classifier Based on a Faceted Classification Scheme

Charles Jochim Hinrich Schütze
Institute for Natural Language Processing,
University of Stuttgart
charles.jochim@ims.uni-stuttgart.de

ABSTRACT
Citations are a valuable resource for characterizing scientific publications that has already been used in applications such as summarization and information retrieval. These applications could be even better served by expanding citation information. We aim to achieve this by extracting and classifying citation information from the text, so that subsequent applications may make use of it. We make three contributions to the advancement of fine-grained citation classification. First, our work uses a standard classification scheme for citations that was developed independently of automatic classification and therefore is not bound to any particular citation application. Second, to address the lack of available annotated corpora and reproducible results for citation classification, we are making available a manually-annotated corpus as a benchmark for further citation classification research. Third, we introduce new features designed for citation classification and compare them experimentally with previously proposed citation features, showing that these new features improve classification accuracy.

KEYWORDS: citation classification, feature extraction.

Proceedings of COLING 2012: Technical Papers, pages 1343–1358,
COLING 2012, Mumbai, December 2012.

1 Introduction

Citations are a valuable resource for characterizing scientific publications and their links to each other. They have been exploited for a number of natural language processing (NLP) and information retrieval (IR) applications, including summarization (Qazvinian and Radev, 2008; Qazvinian et al., 2010)[CJPF][1], improved indexing and retrieval (Ritchie et al., 2006)[CJPF]; and building integrated research databases (Nanba et al., 2004)[CJPF]. Bibliometric measures that quantify the impact of publications (e.g., Moed, 2005)[CJPF] are also based on citations.

Most of this work does not differentiate between uses of citations, e.g., whether a citation is more or less important to the paper or whether the paper's authors support or refute the claims made in the cited work. However, recently a number of research groups have attempted to classify citations with respect to dimensions like importance and relation to cited work (Teufel et al., 2006b; Dong and Schäfer, 2011; Sugiyama et al., 2010; Abu-Jbara and Radev, 2012)[CEPF]. By adding such fine-grained information to individual citations, the various applications of citation analysis can be better served; e.g., citations that are foundational to a paper may constitute better summary sentences for the cited paper.

Thus, there are clear potential benefits to fine-grained citation analysis; and a number of case studies have been published that demonstrate this potential (Nanba et al., 2004; Teufel et al., 2006b)[CEPF]. However, fine-grained citation analysis is currently not widely used in applications that access and analyze the scientific literature. In this paper, we identify a number of potential reasons for this state of affairs and propose solutions.

The first problem with current fine-grained citation analysis is that prior work has tended to develop custom classification schemes for a particular application. This means that the development cycle for a citation classifier must be started from scratch for each new application. In contrast to this prior work, we base our work on a standard classification scheme for citations from information science, the classification scheme of Moravcsik and Murugesan (1975)[CERF] (henceforth MM). We believe it is important to use an annotation scheme that is not bound to automatic citation classification for one particular task such as IR or bibliographic measures. Instead, it should be expressive enough to handle citations across many tasks. The MM scheme comprises four different dimensions or *facets*, which allows us to annotate the quality of the cited work along with its relation to the citing work. This gives the classification flexibility, so that it can be used in different application scenarios; e.g., some facets of the citation are more relevant for IR in digital libraries, while others are more useful in automatic summarization.

The second reason that fine-grained citation analysis has not seen widespread adoption is that it remains a challenge to accurately and automatically classify citations according to a predefined classification scheme (Teufel et al., 2006b)[CEPF]. We address this problem by introducing several novel features designed specifically for use in citation classification. Some of these new features are needed to support the more flexible and generic MM facet classification scheme. In particular, we extract novel features that capture the relationship between the citing paper and the cited paper. Identifying this relationship helps in understanding what motivated an author to reference the cited work. We also investigate how different features perform across the four facets, and how other variables, like the size of the context from which we extract features, affect the classification. We go on to compare different feature sets used for citation

[1]The citation annotation, described later in Sections 2 and 3, has likewise been applied to the citations in this paper. The following abbreviations apply: **C**=*conceptual*; **O**=*operational*; **E**=*evolutionary*; **J**=*juxtapositional*; **R**=*organic*; **P**=*perfunctory*; **F**=*confirmative*; **N**=*negational*.

classification. In particular we compare different lexical, syntactic, and positional features. To our knowledge this is the most extensive investigation of the comparative utility of features for citation analysis to date.

The final barrier to widespread adoption of fine-grained citation analysis is the fact that progress in the field has been hampered by the lack of a standard annotated corpus. Although all of the previous work we cover has used corpora of NLP articles for citation analysis experiments, none has tried reusing an existing corpus or annotation scheme. This makes accurately comparing results impossible, which in turn makes it difficult to gauge the advancement of the state of the art. Authors have focused on developing new annotation schemes, but no work has gone into building resources that allow the research community to evaluate and compare different citation classification methods.

As we will show below, results are also difficult or impossible to reproduce because existing citation approaches have not been described in sufficient detail and resources created or used for the approach have not been published. To address the lack of reproducible experiments in citation classification, we are making available, in conjunction with this paper, the manually-annotated corpus and feature vectors that produce the results reported here.[2] We hope that this corpus can provide a benchmark for further advances in citation classification.

The rest of the paper is structured as follows. Sections 2 and 3 cover the details of our annotation scheme and corpus, followed by a detailed description of the features used for classification in Section 4. Section 5 presents the different classification experiments we conduct with a discussion of the results in Section 6. Section 7 discusses related work. Finally, we close with a summary and an outline of future work.

2 Annotation scheme

In selecting our fine-grained classification scheme, we focused on two criteria. The first criterion is that we should consult the field of research that has the most expertise and the longest research record in developing classification schemes for citations. This field is information science. We have chosen the scheme proposed by Moravcsik and Murugesan (MM) because it adequately represents scientific literature for a broad range of citation classification scenarios. Furthermore, it is a well-established annotation scheme that is widely cited and used inside and outside of the information science community.

The second criterion for selecting the scheme was that it should be flexible and adaptable for different citation use cases. The MM scheme achieves this in that it is composed of four *independent* or *orthogonal* facets. For each facet, it assigns a label from a set of two labels. The scheme can be summarized with the four questions they posed: (i) Is the reference conceptual or operational? (ii) Is the reference organic or perfunctory? (iii) Is the reference evolutionary or juxtapositional? (iv) Is the reference confirmative or negational?

The *conceptual* vs *operational* facet – **CONC-OP** – asks: "Is this an idea or a tool?," where examples of tools are MRI in brain imaging and part-of-speech (POS) taggers in NLP. The *organic* vs *perfunctory* facet – **ORG-PERF** – distinguishes those citations that form the underpinnings of the citing work from more cursory citations. The *evolutionary* vs *juxtapositional* facet – **EVOL-JUX** – highlights the relationship between the citing and cited papers. If the citing paper builds on the cited work, it is EVOL while it is JUX if it presents an alternative to the cited

[2] `http://www.ims.uni-stuttgart.de/~jochimcs/citation-classification`.

work. Finally, **CONF-NEG**, the *confirmative* vs *negational* facet, captures the completeness and correctness of the cited work. A NEG citation usually is not derogatory, it may simply say that the cited work is weaker than the citing work or is otherwise missing some critical point. These distinctions are covered in more detail in the annotation guidelines.[2]

These four facets can be thought of as orthogonal dimensions along which citations can vary. This is the basis for flexible and adaptable citation analysis; e.g., a facet that is not relevant for a particular application can simply be omitted. If interactions between two facets are important for another application, they are made available by the citation classifier without complicating the model or its training.

Although there are now four facets to annotate for each citation instead of a single label, the annotation task is not more difficult. Making a binary decision is easier than trying to pick a label from ten possibilities with subtle differences between some of them. Yet, with the combination of different facets we still can achieve a finer-grained label.

It is also important to note that this classification has no undefined class. Several previous annotation schemes have a default label, *neutral* or *other*, that is assigned to a citation when no other classes can be. In the work we have seen that uses such annotation schemes, more than half of the citation instances are assigned this undefined label. In these cases, summarization or IR systems that want to make use of citation information obtain no useful information from the citation classifier for more than half of citations.

3 Corpus

Our corpus, like corpora from some previous studies (Athar, 2011; Dong and Schäfer, 2011)[CEPF], is taken from NLP literature. Specifically, we have taken the 2004 ACL proceedings from the ACL Anthology Reference Corpus (ARC) (Bird et al., 2008)[OEPF]. NLP literature was chosen because our annotators (NLP students) are more familiar with this data and can make more informed decisions when annotating the citations.

Some statistics on the number of documents and citations in the corpus can be found in Table 1. Each citation in the corpus has been independently annotated by at least two of six annotators. Gold labels are chosen by a simple majority vote and in the case of ties the votes of more experienced annotators are weighted higher. The annotators were given guidelines to help ensure consistent annotation. We built a browser-based annotation tool that displays the full text of the paper, so that the annotators can look at the wider context of the citation when necessary. In many cases the context necessary for annotation is only one sentence, but it will often span sentences or fill a paragraph.

section	docs	citations
main ACL	57	1668
student	6	101
poster/demo	21	239
total	84	2008

CONC	OP	EVOL	JUX
1792	216	1804	204
ORG	**PERF**	**CONF**	**NEG**
203	1805	1836	172

Table 1: ACL 2004 corpus (left) and summary of annotated citations (right).

As mentioned in Section 2, no facets were left undefined. This reduces the classification to only two classes and avoids a neutral class. For our purposes it is reasonable to avoid having a neutral class; e.g., a citation that is not explicitly CONF should still be implicitly considered CONF because including the citation is still an endorsement of the cited work.

Fleiss's κ values of the annotation are .42 (CONC-OP), .45 (EVOL-JUX), .18 (ORG-PERF) and .41 (CONF-NEG). These numbers indicate that the difficulty of the annotation task varies for the different facets, with ORG-PERF being most difficult.[3] Due to the highly skewed distribution, κ suffers from *prevalence* (Eugenio and Glass, 2004)[CEPF]; yet three of the facets still have *moderate* agreement (according to Landis and Koch (1977)[CEPF]), and ORG-PERF has *slight* agreement. We feel that the observed agreement[4] is high enough that we can rely on the gold labels for evaluation.

We are releasing the corpus along with this paper.[2] To the best of our knowledge this corpus is the first to be annotated by individuals other than the study's authors. It is important to have independent annotators to limit any bias in the gold-standard annotation. One consequence of this is that our inter-annotator agreement scores are lower than those previously published as the previous annotation came from the developers of the respective annotation schemes and from the authors reporting on the classification experiments using them.

4 Description of features

Our goal is to accurately classify citations according to MM, the annotation scheme described in Section 2. We make the assumption that the necessary clues for correctly labeling citations, both manually and automatically, can be found in the context of the citation, i.e., the running text surrounding the citation. If we are able to extract the right clues from the citation context we can accurately label the citation's use.

Because there is not yet a standard corpus for the task of automatic citation classification, the results from previous work are difficult to compare. Previous studies have used different corpora, different annotation schemes, different feature sets, and different classifiers. In an effort to borrow from – and eventually compare ourselves to – previous work, we investigate some of the features used previously and introduce our own. The reader may want to refer to the overview of features in Table 2 as we describe the features in what follows.

Lexical features. Much of the earlier work on automatic citation classification (Dong and Schäfer, 2011; Nanba and Okumura, 1999; Teufel et al., 2006b)[CEPF] relied on cue words and phrases ($\texttt{cues}_k$). These were often implemented as follows. For a class (e.g., Dong and Schäfer's *idea* class), a list of of cues (e.g., the word "following") are defined that indicate that class. Finally, a Boolean feature (e.g., $\texttt{cues}_{idea}$) is set to true if any word from the list is in the citing context. This results in k Boolean features where k is often the number of classification labels (although it can be greater, see Dong and Schäfer (2011)[CEPF]).

Different length n-grams were later used by Athar (2011)[CJPF] with results indicating that combined unigram, bigram, and trigram features (`1+2+3-gram`) performed better than unigrams (`1-gram`) and unigrams plus bigrams (`1+2-gram`).

We use only unigrams because they perform at least as well as using unigrams, bigrams and trigrams in our experiments, without introducing a much larger, sparsely-populated feature set. Unigrams should also be quite robust and perform reasonably well across the four facets.

Word-level linguistic features. Part-of-speech (POS) tags of the words in the citation sentence were used as features by Athar (2011)[CJPF] (`POS` and `1-gram+POS`). Select linguistic features

[3] MM's definition was "is the reference truly needed for the understanding of the referring paper," so the annotation hinges on the understanding of the individual annotator, resulting in higher disagreement.

[4] Agreements were: .86 (CONC-OP), .88 (EVOL-JUX), .72 (ORG-PERF), .91 (CONF-NEG)

feature class	name	source	type or *example value*	description
lexical feats.	cues$_k$	NO99, TST06, DS11	Boolean	k Boolean features: one for each group of cue words/phrases
	1-gram	Ath11, own	*hard*	unigrams
	1+2-gram	Ath11	*hard language*	unigrams & bigrams
	1+2+3-gram	Ath11	*hard language like*	unigrams, bigrams, & trigrams
word-level linguistic feats.	POS	Ath11	*NN, JJ, IN*	POS tags
	1-gram+POS	Ath11	*quality+NN, new+JJ*	POS tag-word conjunctions
	tense	TST06	*present, past*	verbal tense
	voice	TST06	*active, passive*	verbal voice
	modal	TST06	*can, may*	modal verb (if any)
	has-modal	own	Boolean	sentence has modal verb
	root	own	*have, present*	dependency root node
	main-verb	own	*present, use*	main verb
	has-1stPRP	own	Boolean	first person POS
	has-3rdPRP	own	Boolean	third person POS
	comp/sup	own	*more, better*	comparative/superlative POS
	but	own	Boolean	has "but"
	has-cf	own	Boolean	has "cf."
ling. structure feats.	dep-rel	Ath11	*pobj:to:information*	Stanford typed dependencies (de Marneffe et al., 2006)
	POS-pattern$_k$	DS11	Boolean	k Boolean features: one for each POS tag pattern
	is-constituent	own	Boolean	citation is a constituent
	self-comp	own	Boolean	author linked to comparative
	other-comp	own	Boolean	citation linked to comparative
	other-contrast	own	Boolean	citation is in contrastive clause
	self-good	own	Boolean	author linked to positive sentiment
location feats.	section	DS11	*Introduction, Method*	1 of 6 possible section headings
	paper-loc	TST06	unknown	citation position in paper
	paragraph-loc	TST06	unknown	citation position in paragraph
	section-loc	TST06	unknown	citation position in section
	sentence-loc	own	*beginning, middle, end*	location in the first quarter, middle half (25%-75%), and last quarter
frequency feats.	popularity	DS11	Integer	citations in the same sentence
	density	DS11	Integer	citations in the same context (sentence and its neighbors)
	avgDensity	DS11	Real	average density of neighboring sentences
sent. feats.	scilex	Ath11	unknown	scientific polarity lexicon
	cpol	Ath11	unknown	general polarity lexicon
	positive-words	own	*best, advantage*	general positive lexicon
	negative-words	own	*problem, against*	general negative lexicon
other feats.	self-cite	TST06	Boolean	citation to own work
	has-resource	own	Boolean	resource entity found with NER
	has-tool	own	Boolean	tool entity found with NER

Table 2: Feature list (grouped by feature class). NO99=Nanba and Okumura (1999); TST06=Teufel et al. (2006b); Ath11=Athar (2011); DS11=Dong and Schäfer (2011). "unknown" = exact definition of the feature (e.g., Boolean or Real) is unknown. Examples of possible feature values are given in italics where appropriate.

related only to the main verb were shown to be effective by Teufel et al. (2006b)[CEPF]; e.g., tense (`tense`), voice (`voice`), and modality (`modal`).

We also include modality in our feature set (`has-modal`) along with separate features for the main verb (`main-verb`) and the root (`root`) as determined by the MATE dependency parser (Bohnet, 2010)[OEPF]! We do not include POS as features per se, but some features are triggered by the occurrence of selected POS: 1st and 3rd person pronouns (`has-1stPRP`, `has-3rdPRP`); and comparatives and superlatives (`comp/sup`). Comparatives and superlatives can help distinguish CONF from NEG. Pronouns on the other hand may be useful in classifying EVOL-JUX, e.g., first person pronouns are used when clarifying the differences between proposed and cited approaches. We add two other features for the contrastive conjunction "but" (`but`) and the abbreviation "cf." (`has-cf`). In our analysis of citations we looked at the role of contrastive conjunctions in citation sentences and found these simple features to be useful.

Linguistic structure features. Dependency relations (`dep-rel`) were used as features and showed a marked improvement over the baseline by Athar (2011)[CEPN]! Dong and Schäfer (2011)[CEPF] used seven regular expression patterns of POS tags ($\texttt{POS-pattern}_k$) to capture syntactic information (e.g., ".*(VHP|VHZ) VV.*"); then $k = 7$ Boolean features marked the presence (or absence) of these patterns.

We add other new features related to the linguistic structure of the citation sentence. For `is-constituent`, the citation is labeled as a constituent if the authors appear outside of the parentheses with only the date in parentheses, e.g., "Gusfield (1997) showed that . . . ", or if the citation acts as a placeholder for the cited work following a preposition, e.g., ". . . following the experiments in (Kaplan et al., 2004)". These cases are distinguished from citations like: ". . . are two popular examples of kernel methods (Fukunaga, 1990; Cortes and Vapnik, 1995)". We are relying here on a certain style of writing and citation format, like that found in ACL proceedings. We expect this feature to help for ORG-PERF as organic citations are more likely to show up as constituents in citation sentences.

The personal pronoun and comparative features mentioned above (`has-1stPRP`, `has-3rdPRP`, and `comp/sup`) are useful features, but we would like to extract a more specific feature that links them. We want features that indicate that the citing work is better than the cited work. To obtain these features we parse the sentence and extract relations from the parse tree. For the author/comparative relation, we first find the comparative in the sentence and traverse the tree to find the subject of the phrase that contains that comparative. If the subject refers to the author of the paper (e.g., with a first person pronoun), we set the `self-comp` feature to true.

We also found that JUX citations are often set apart using contrastive conjunctions, e.g., *while* or *despite*. We again traverse the parse tree to extract the relationship between contrastive conjunctions and the citation (`other-contrast`), where the citation or cited authors show up in the dependent clause governed by the contrastive conjunction. The feature is set to true if the citation is found among the descendants of the contrastive conjunction.

Location features. The section of the paper in which the citation is located (`section`) was used as a feature by Dong and Schäfer (2011)[CEPF]! Teufel et al. (2006b)[CEPF] also included location features at different granularities: within the paper (`paper-loc`), within the paragraph (`paragraph-loc`), and within the section (`section-loc`).

We include a different location feature approximating where the citation is found in the sentence (`sentence-loc`): beginning, middle, or end. This feature is motivated by the fact that citations

at the end of the sentence are predominantly PERF.

Frequency features. Dong and Schäfer (2011)[CEPF] used the number of citations in a single sentence (`popularity`) and in the citation sentence plus its neighboring sentences (`density`) as features. They also included a third feature for the average density of neighboring sentences (`avgDensity`).

Sentiment features. Athar (2011)[CEPF] included two different polarity lexicons. One is hand-crafted and specific to the scientific domain (`scilex`). The other is the large general purpose polarity lexicon from Wilson et al. (2005)[OEPF] (`cpol`). He also tried features (`neg`) that account for negation. This was done by appending "_neg" to the end of the 15 lexical items that follow any negation term.

We were not able to obtain the scientific polarity lexicon, but use the polarity lexicon from Wilson et al. (2005)[OEPF] to extract sentiment features. Our polarity features are represented as a bag of words (BOW) where the citation context words present in the polarity lexicon are added to the BOW features `positive-words` or `negative-words` according to their polarity. Although CONF-NEG is not strictly a matter of sentiment, we still apply this feature hoping for improvements on this facet.

Self-reference feature. Teufel et al. (2006b)[CEPF] used a feature, `self-cite`, that indicates if one of the citing authors also (co-)authored the cited work. This feature is unique in that it is the only feature based on the reference and not the individual citation and therefore not taken from the context in which it is found.

NER features. Using lexical features alone, there are a number of words that help indicate OP (*operational*) citations in NLP, e.g., "parser", "tagger", "corpus". We decide to take this a step further and train a named-entity recognition (NER) system to identify NLP named entities. We identify two types of NLP named entities: corpora and tools. First, we create a gazetteer of NLP tools and corpora from an online list of these resources.[5] Next, we tag a portion of our corpus using the gazetteer list to label any occurrence of the words in the list and then manually check those labeled instances to be sure they are correctly labeled. In this way we can expediently create training data, with an emphasis on precision over recall. Finally, we train the SuperSenseTagger (Ciaramita and Altun, 2006)[OEPF] on this annotated portion, and tag the remaining part of the corpus. NER is not central to our task, so we did no direct evaluation of it; we looked only to see if it might lead to improvements in our classification. We include two features, `has-resource` and `has-tool`, for the two types of entities.

The NER features we extract are related only to the NLP domain. However, this approach for acquiring named entities is not domain dependent and can be used to develop a reasonably efficient NER system using lists of tools or resources from any domain.

5 Experiments

In this section we will outline our classification experiments and then discuss the results in Section 6. We use the term *feature set* to describe a collection of features used by us or in previous studies; we use the term *feature class* to describe a collection of similar features as they are organized in Section 4 and in Table 2.

Setup. All our experiments were conducted on the corpus described in Section 3. We trained the Stanford MaxEnt classifier (Manning and Klein, 2003)[OEPF] for each of the four facets

[5] `http://nlp.stanford.edu/links/statnlp.html`

in a 5-fold cross validation setup with default settings except that we set the regularization parameter $\sigma = 10$ based on previous experiments.

Feature set comparison. In our first set of experiments we test our own feature set and the feature sets described in previous studies. Each of these feature sets is a subset of the features described in Section 4 and is identified below by some of its more distinguishing features; e.g., **NgramDep** refers to the feature set that mainly uses n-grams and dependencies.

CueVerbLoc. This feature set is intended to mimic (Teufel et al., 2006b)[CEPF] to the extent this is possible. It includes cue phrase features ($\texttt{cues}_k$), the verbal features `tense`, `voice`, and `modal` as well as `paper-loc`. The cue phrases used in (Teufel et al., 2006b)[CJPF] are not available so we applied automatic feature selection using mutual information (MI) (Manning et al., 2008)[CEPF] to select the most informative unigrams, bigrams, and trigrams for each class label. We borrow from the manual feature selection in (Teufel et al., 2006b)[CEPF] by assigning cue phrases to each of the labels (8 in our case – Teufel et al. used 12) and limiting the number of cue words to 75 per label. Some examples for OP cues are *wordnet*, and *parser*.

NgramDep. This feature set corresponds to (Athar, 2011)[CEPF]. It includes lexical features: unigrams, bigrams, and trigrams (`1+2+3-grams`) and the `dep-rel` features. Athar (2011)[CJPF] tested other features, but we have only reimplemented those that improved results.

CueFreqPOS. This feature set is based on (Dong and Schäfer, 2011)[CEPF]. It includes a list of cue words ($\texttt{cues}_k$), then the frequency features `popularity`, `density`, `avgDensity`, and the syntactic feature $\texttt{POS-pattern}_k$.

PREV. This feature set combines all features previously used for citation classification into one feature set (i.e., CueVerbLoc + NgramDep + CueFreqPOS).

OWN. The feature set OWN includes all the features we have introduced in our work – those marked "own" in Table 2. Some features were designed to help one facet or another, but we use them all together here for all facets.

We note here that by reimplementing features from previous work we claim only to extract the same or similar information as the original authors. Due to sometimes major differences in the corpus, annotation scheme, and classifier used, we are not able to reproduce the same conditions that led to previous results. We are instead more interested in the types of features that seem to perform best on our dataset with our annotation scheme.

Citation context size. The tests just described are run with a fixed context size of one sentence. It is not clear how much context is best for feature extraction, so in another set of experiments we fix the feature set and test the features extracted from different sized context windows. In previous work, different sized context windows were used by different studies, e.g., Athar (2011)[CEPF] used only the sentence containing the citation while Dong and Schäfer (2011)[CEPF] used up to three sentences. Kaplan et al. (2009)[CEPF] and Abu-Jbara and Radev (2012)[CEPF] have illustrated the difficulties in delineating the exact boundary for each individual citation context, while Athar and Teufel (2012)[CEPF] tried different fixed context sizes for citation classification. We follow this general idea and test context lengths of 1, 2, and 3 sentences.

Feature class comparison. In addition to comparing our own feature set with those from previous work, we also want to investigate what feature classes assist most in the classification. We perform this analysis by examining the impact of the seven feature classes described in Section 4. More specifically, we compare the results of their individual performance using *only features in the feature class* (Table 4, top), and their ablation from the entire feature set using *all*

features except those in the feature class (Table 4, center). Finally, we extend the ablation study, *successively removing all feature classes* in order of importance (i.e., by their contribution to F_1 score) (Table 4, bottom).

6 Results and Discussion

Feature set results. The results for the different feature sets when using one sentence of context are found in Table 3. All of the F_1 results presented in this paper are macro-averaged F_1. We have included two baseline experiments. We use a majority baseline (BL) that labels each citation with the label occurring most often in the corpus, e.g., for CONC-OP, all citations are labeled CONC. We also include results for unigram, bigram, and trigram features (Ngram), which is the baseline used by Athar (2011)[CJPN]. The results in Table 3 show that our feature combination outperforms both baselines and all reimplemented feature sets for all four facets. With two exceptions (Ngram for EVOL-JUX and PREV for ORG-PERF), these results are significant.[6]

The greatest improvement over the baseline is with the OWN features for CONC-OP. Several of the other feature sets also do better on CONC-OP than BL, but OWN is still significantly better than PREV, the combination of all other feature sets. Simple BOW features along with our new features (e.g., `has-resource` and `has-tool`) increase F_1 by 7 points over PREV. As an example, in a sentence citing "The Penn TreeBank (Marcus et al., 1993)," the citation is incorrectly classified using PREV. The NER tool recognizes Penn TreeBank as a corpus, which results in the OWN feature `has-resource` to be set to true and a correct classification of the citation as OP.

EVOL-JUX proves to be more difficult than CONC-OP with either no or very small improvements over BL for all feature sets except for Ngram and OWN. The BOW features from our OWN feature set are responsible for most of the improvement of F_1 from 47.3 to 52.9. BOW features contribute to the improvement with OWN features for all four facets.

OWN features improve F_1 by 10.7 (from 47.3 to 58.0) over the BL for ORG-PERF, and are also better by 3.2 (54.8 vs 58.0) than PREV. Some features that contribute to the better results are `root` and `main-verb` with values such as "describe" and "present"; these appear to be useful in identifying ORG citations. In this facet, the feature set CueFreqPOS sees its most significant improvement over BL. This is due in a large part to the frequency features that are not found in other feature sets.

Finally, CONF-NEG is the most difficult facet. All feature sets except our own performed only as well as or even worse than BL. OWN features improve F_1 by 3.3 (from 47.8 to 51.1), which is due in part to the location feature that finds citations in the middle of sentences to be CONF, while NEG citations are more likely to come at the beginning.

To get an idea of a possible upper bound for this task, we include a *human classifier* ("Human" in Table 3): we take the annotation from the most experienced annotator and consider it as classification output. CONC-OP is the "easiest" facet for the human classifier to label, similar to automatic classification. However, the most difficult facet for automatic classification, CONF-NEG, appears to be straightforward for the human classifier. This is consistent with the high observed agreement for CONF-NEG (.91, footnote 4).

Context size results. For OWN, we tested three different context sizes c: $c \in \{1, 2, 3\}$ sentences. We found that $c = 1$ is best for CONC-OP (significant) and ORG-PERF; and $c > 1$ is better for

[6] $p < .05$. All significance tests in this paper use the approximate randomization test (Noreen, 1989)[CEPF].

	CONC-OP	EVOL-JUX	ORG-PERF	CONF-NEG
baseline (BL)	*47.2	*47.3	*47.3	*47.8
Ngram	*53.2	50.7	*51.3	*47.8
CueFreqPOS	*48.4	*49.4	*54.1	*47.7
NgramDep	*53.3	*47.3	*50.5	*47.8
CueVerbLoc	*51.1	*47.3	*47.3	*47.8
PREV	*61.2	*48.5	54.8	*47.5
OWN	68.2	52.9	58.0	51.1
Human	94.7	91.1	91.7	93.5

Table 3: F_1 for different feature sets. Marked with $*$: significantly worse than OWN ($p < .05$). Underlined: best performing feature set per facet.

CONF-NEG (significant) and EVOL-JUX. These results suggest that context size is an important factor, but one that does not have a uniform effect on the four facets. The online appendix describes these experiments in more detail.[2]

Feature class results. In the discussion of the feature class results we will refer to the line numbers in Table 4. The table presents F_1 results using only a single feature class (lines 1–7); F_1 using all features ("All") and F_1 using all features except the listed feature class (lines 8–14); and finally, extended ablation results where a feature class is successively removed from "All" (seven classes) until one feature class remains (lines 15–21). Our goal is to get a better idea of which feature classes are informative for a given facet.

CONC-OP. LEXICAL features appear to be the most important for this facet. Alone they do well against the baseline (61.6 vs 47.2, line 1) and when removed from the entire feature set F_1 drops more than for any other feature class (from 64.5 to 58.2, line 8). Both of these Δ's are significant. The feature class NER has the second highest F_1 (54.1, line 7) when used alone, which makes sense as it was designed for this facet. Removing NER features hurts F_1 (down to 64.0, line 14), but not significantly. Using only WORD-LEVEL or STRUCTURE features also leads to significant improvement: increases of 4.8 (line 2) and 4.3 (line 3). After that, SENTIMENT features improve F_1 but not significantly (line 6), while the LOCATION and FREQUENCY features show no difference from the BL (lines 4–5). The ablation results show that after the significant contributions of the LEXICAL features, the removal of other feature classes does not affect the results much: Removing STRUCTURE, LOCATION, and SENTIMENT features actually increases F_1 (lines 10, 11, 13), and the ablation of WORD-LEVEL, FREQUENCY, and NER features shows no significant change (lines 9, 12, 14).

EVOL-JUX. For this facet, three of the seven feature classes, LOCATION, FREQUENCY, and NER, lead to no change from the baseline when run alone (lines 4, 5, 7). Another three feature classes, LEXICAL, WORD-LEVEL, and SENTIMENT, significantly improve over BL (lines 1, 2, 6). Conversely, the FREQUENCY features, with no improvement alone, help improve results of the entire feature set; when those features are removed, F_1 drops by 2.2 (from 53.4 to 51.2, line 12). Also, the SENTIMENT features, which do well against the baseline (line 6), hurt F_1 when added to the full feature set (decrease by -0.7, line 13).

ORG-PERF. Individually, the feature classes LEXICAL, WORD-LEVEL, and STRUCTURE all had significant improvements (lines 1–3). The other four classes do not help for this facet (lines 4–7). However, in the ablation results, omitting these feature classes also *increases* F_1 (lines 8–10). Only removing LOCATION significantly decreases F_1 (line 11). This result indicates that several of the feature classes are correlated for classifying this facet. They contain useful information for the task (as indicated by good performance when used individually), but mutual correlation has

the effect of bad generalization when all of them are used together. The results show that this type of analysis (which has not been performed before for citation classification) is important to understand how features impact performance and what steps are needed to achieve better performance.

		CONC-OP		EVOL-JUX		ORG-PERF		CONF-NEG	
	BL	47.2		47.3		47.3		47.8	
		F_1	Δ BL	F_1	Δ BL	F_1	Δ BL	F_1	Δ BL
1	LEXICAL	†61.6	†14.4	†52.7	†5.4	†56.1	†8.8	47.7	0.0
2	WORD-LEVEL	†52.0	†4.8	†52.4	†5.0	†51.6	†4.2	49.7	2.0
3	STRUCTURE	†51.5	†4.3	48.8	1.5	†52.0	†4.7	47.8	0.0
4	LOCATION	47.2	0.0	47.3	0.0	47.3	0.0	47.8	0.0
5	FREQUENCY	47.2	0.0	47.3	0.0	47.3	0.0	47.8	0.0
6	SENTIMENT	48.0	0.9	†52.7	†5.3	47.2	-0.1	†49.9	†2.1
7	NER	†54.1	†7.0	47.3	0.0	47.3	0.0	47.8	0.0

		CONC-OP		EVOL-JUX		ORG-PERF		CONF-NEG	
	All	64.5		53.4		59.2		48.9	
		F_1	Δ All	F_1	Δ All	F_1	Δ All	F_1	Δ All
8	LEXICAL	*58.2	*6.2	53.3	0.1	60.2	-1.0	49.5	-0.6
9	WORD-LEVEL	64.0	0.4	53.6	-0.3	59.3	-0.1	48.9	0.1
10	STRUCTURE	66.7	-2.2	53.1	0.3	59.5	-0.3	*48.8	*0.1
11	LOCATION	65.0	-0.5	53.2	0.1	*55.8	*3.4	49.6	-0.6
12	FREQUENCY	64.4	0.1	51.2	2.2	58.3	0.9	49.8	-0.9
13	SENTIMENT	65.0	-0.5	54.1	-0.7	59.2	0.0	49.0	0.0
14	NER	64.0	0.4	53.7	-0.3	58.8	0.4	49.4	-0.5

	CONC-OP		EVOL-JUX		ORG-PERF		CONF-NEG	
	All	64.5	All	53.4	All	59.2	All	48.9
15	LEXICAL	*58.2	FREQUENCY	51.2	LOCATION	*55.8	STRUCTURE	*48.8
16	NER	*53.6	STRUCTURE	*50.3	LEXICAL	*55.4	WORD-LEVEL	48.2
17	STRUCTURE	*50.3	LEXICAL	50.6	STRUCTURE	*53.0	LOCATION	47.4
18	WORD-LEVEL	*47.9	NER	50.7	WORD-LEVEL	*48.6	NER	47.4
19	SENTIMENT	*47.2	LOCATION	52.7	SENTIMENT	*47.3	SENTIMENT	47.4
20	FREQUENCY	*47.2	SENTIMENT	52.4	FREQUENCY	*47.3	FREQUENCY	47.7
21	LOCATION	*47.2	WORD-LEVEL	*47.3	NER	*47.3	LEXICAL	47.8

Table 4: **Top.** Results when a single feature class is used. **Middle.** Ablation results: F_1 and decrease in F_1 when each feature class is ablated; i.e., each result shown is a classification result using six feature classes. **Bottom.** Extended ablation results: Left columns indicate the feature class removed. Marked with †: significantly better than BL ($p < .05$); marked with $*$: significantly lower than All ($p < .05$). Underlined: best performing feature class per facet (largest Δ).

CONF-NEG. Only SENTIMENT (line 6) and WORD-LEVEL (line 2) improve over BL and the remaining five feature classes do only as well as BL. Removing four of the seven feature classes actually seems to improve F_1 (lines 8, 11, 12, 14), with F_1 only increasing by adding WORD-LEVEL or STRUCTURE features (lines 9–10).[7] In fact, it seems that including the feature classes LEXICAL, STRUCTURE, LOCATION, FREQUENCY, and NER might only be detrimental for this facet, as F_1 using only SENTIMENT features is 49.9 (line 6) compared to using all features at 48.9 ("All").

To further analyze the relative importance of a feature class for a facet we extend the ablation results by successively removing that feature class whose removal results in the lowest F_1, among the possible ablations, until all have been removed (Table 4, lines 15–21). E.g., in CONC-OP we start with all features ($F_1 = 64.5$) and calculate F_1 after removing each of the feature classes individually. In this case, removing LEXICAL leads to the largest drop in F_1, from

[7] Lines 9–10 have different F_1 (48.9 vs 48.8) but the same $\Delta = 0.1$ due to rounding.

64.5 to 58.2 (line 15). In the next iteration, we again compare F_1 after removing each of the six remaining feature classes. Removing NER features results in the lowest F_1 (now 53.6, line 16), and we proceed by removing one of the five remaining feature classes, etc. These results support what was discussed for the top and middle portions of Table 4, but present it as a list of the feature classes in descending order of importance. This table helps us to compare different facets; we can easily see that LEXICAL and NER features are important for CONC-OP, while LOCATION features are not. Compare this to CONF-NEG where LEXICAL and NER features are not important and WORD-LEVEL is higher in the list. Note also, that F_1 does not always decrease (e.g., removing LEXICAL for EVOL-JUX). Some combinations of subsets of features will perform better than the previous superset. In this case, we see that after having removed STRUCTURE, removing any other feature class can only improve results.

The results in this section give us some valuable insight into how to design features for citation classification. First, we consider the first three feature classes, LEXICAL, WORD-LEVEL, and STRUCTURE. All three contain quite general text classification features, and consequently are quite robust and informative across the four facets of citations that we consider. WORD-LEVEL seems to be the most robust across all four facets, while LEXICAL has the largest Δ BL values for three of the four facets (i.e., CONC-OP, EVOL-JUX, and ORG-PERF). The last four feature classes – LOCATION, FREQUENCY, SENTIMENT, NER – represent different citation features which seem to impact certain citation facets. NER was designed particularly for CONC-OP and does in fact contribute most to that facet; LOCATION helps only ORG-PERF (i.e., the position of the citation indicates its importance) where it contributes significantly to a combination of features; similarly FREQUENCY contributes significantly to a combination of features for EVOL-JUX; and finally, SENTIMENT is important for EVOL-JUX and CONF-NEG, as expected. There is no single feature class that is the most important for all facets, which lends credence to the claim that these facets capture different properties of citations. We conclude that our multi-faceted scheme benefits from a diverse feature set and that although general, easily-extractable features help classification more consistently, the extraction of more specific features is important for improvements on certain classification tasks.

7 Related Work

Information scientists started labeling and studying citations long before automatic text classification became a reality. Garfield (1964)[CEPF] originally introduced 15 different motivations for why an author might cite a paper; Weinstock (1971)[CEPF] then revisited this classification as he explored the emergence of citation indexes. Several studies following Weinstock also aimed to characterize the *function* of citations (as opposed to the motivation). One example is the MM scheme we adopt here. Chubin and Moitra (1975)[CEPF] attempted to simplify and flatten MM using six categories. Spiegel-Rösing (1977)[CEPF] produces another classification scheme with 13 categories that she uses to evaluate one journal's scholarly contributions. Further comparison of previous citation studies can be found in (Liu, 1993)[CEPF] and more recently in (Bornmann and Daniel, 2008)[CEPF]. We note that several other studies (Cano, 1989; McCain and Turner, 1989)[CEPF] have also reused or refined MM in some way, which reinforces our choice. As stated earlier in Section 2, we feel that the multi-faceted composition of MM provides us with a more flexible annotation scheme and a powerful one that can easily represent the quality of a citation as well as its relation to the citing author.

These early annotation schemes were manually applied to a limited amount of scientific literature and did not consider automatic application on large amounts of text. One early

application of automatic citation classification (Nanba and Okumura, 1999)[CEPF] used an annotation scheme with only three classes (Basis, Compare, Other) that are reportedly based on the 15 classes from Weinstock (1971)[CEPF]. Teufel et al. (2006a)[CEPF] introduce a much more complete annotation scheme with 12 classes designed for IR. They thoroughly motivate and analyze their annotation scheme and report inter-annotator agreement of κ=.72. More recently, sentiment analysis has been applied to citations. Athar (2011)[CEPF] classifies citations as *positive*, *negative*, and *objective*, and finds marked improvement in classification using dependency relation features. Athar and Teufel (2012)[CEPF] extend this work and consider context windows of different widths. For each of these three studies the largest class is the one with the least informative label: Nanba and Okumura's *Other* is 52% of citations; Teufel et al.'s *Neutral* is 63%; and Athar's *objective* is 86%. This means that an application receives little information about a majority of citations. In contrast, our annotation scheme does not have a neutral label and always assigns a multi-faceted label that will contain some useful information as no facet can be left undefined.

Dong and Schäfer (2011)[CEPF] conducted a classification study using their own classification scheme with four labels relating to the function of the MM *organic/perfunctory* facet. In addition to adding new syntactic features (`POS-pattern`$_k$, see above), they tested ensemble-style self-training to overcome the problem of limited annotated data. Their paper also included a new dataset with annotated citing sentences. It is important to use previously-tested, publicly-available data, however, their dataset does not contain the full corpus from which they extracted features. Due to this restriction we cannot extract many of the features that they use (e.g., features in the LOCATION and FREQUENCY classes). The annotation in their dataset is also attached to the sentence and not individual citations. This makes it impossible to classify individual citations and prevents us from using the citation-specific features that we have developed (OWN features in STRUCTURE class, e.g., `is-constituent`). We have conducted experiments on the Dong and Schäfer dataset and include those experiments in the online appendix.[2] We believe that annotating and classifying citing sentences (as opposed to citations) is not specific enough for tasks like IR and bibliometrics. Thus, it is essential that we have a citation-annotated corpus for accurate classification.

As we have argued above, the motivation for our work is to provide a generic classification scheme that is established and accepted in information science in the hope that it can be used for a wide range of applications.

Conclusion

In this paper, we address the task of citation classification for applications that access and analyze the scientific literature. Our work uses MM, a standard classification scheme for citations that was developed independently of automatic classification and therefore is not bound to any particular citation application. We introduce new features designed for citation classification and show that they improve performance as measured by F_1. To address the lack of available annotated corpora and reproducible results for citation classification, we are publishing, along with this paper, a manually-annotated corpus as a benchmark for further citation classification research. In future work, we want to further extend the feature set to improve classification and show the benefits of our system for applications like bibliometrics.

Acknowledgments. We thank DFG for funding this work (SPP 1335 *Scalable Visual Analytics*) and Christian Scheible, Wiltrud Kessler, Alex Fraser and the anonymous reviewers for their contributions to the paper.

References

Abu-Jbara, A. and Radev, D. R. (2012). Reference scope identification in citing sentences. In *Proceedings of HLT-NAACL*, pages 80–90.

Athar, A. (2011). Sentiment analysis of citations using sentence structure-based features. In *Proceedings of the ACL Student Session*, pages 81–87.

Athar, A. and Teufel, S. (2012). Context-enhanced citation sentiment detection. In *Proceedings of HLT-NAACL*, pages 597–601.

Bird, S., Dale, R., Dorr, B., Gibson, B., Joseph, M., Kan, M.-Y., Lee, D., Powley, B., Radev, D., and Tan, Y. F. (2008). The ACL anthology reference corpus: A reference dataset for bibliographic research in computational linguistics. In *Proceedings of LREC*, pages 1755–1759.

Bohnet, B. (2010). Top accuracy and fast dependency parsing is not a contradiction. In *Proceedings of COLING*, pages 89–97.

Bornmann, L. and Daniel, H.-D. (2008). What do citation counts measure? A review of studies on citing behavior. *Journal of Documentation*, 64(1):45–80.

Cano, V. (1989). Citation behavior: Classification, utility, and location. *Journal of the American Society for Information Science*, 40:284–290.

Chubin, D. E. and Moitra, S. D. (1975). Content analysis of references: Adjunct or alternative to citation counting? *Social Studies of Science*, 5:423–441.

Ciaramita, M. and Altun, Y. (2006). Broad-coverage sense disambiguation and information extraction with a supersense sequence tagger. In *Proceedings of EMNLP*, pages 594–602.

de Marneffe, M.-C., MacCartney, B., and Manning, C. D. (2006). Generating typed dependency parses from phrase structure parses. In *Proceedings of LREC*, pages 449–454.

Dong, C. and Schäfer, U. (2011). Ensemble-style self-training on citation classification. In *Proceedings of IJCNLP*, pages 623–631.

Eugenio, B. D. and Glass, M. (2004). The kappa statistic: a second look. *Computational Linguistics*, 30(1):95–101.

Garfield, E. (1964). Can citation indexing be automated? In *Statistical Association Methods for Mechanized Documentation, Symposium Proceedings*, pages 189–192.

Kaplan, D., Iida, R., and Tokunaga, T. (2009). Automatic extraction of citation contexts for research paper summarization: A coreference-chain based approach. In *Proceedings of 2009 Workshop on Text and Citation Analysis for Scholarly Digital Libraries*, pages 88–95.

Landis, J. R. and Koch, G. G. (1977). The measurement of observer agreement for categorical data. *Biometrics*, 33(1):159–174.

Liu, M. (1993). Progress in documentation the complexities of citation practice: A review of citation studies. *Journal of Documentation*, 49(4):370–408.

Manning, C. D. and Klein, D. (2003). Optimization, maxent models, and conditional estimation without magic. In *Proceedings of HLT-NAACL*, pages 8–8.

Manning, C. D., Raghavan, P., and Schütze, H. (2008). *Introduction to Information Retrieval*. Cambridge University Press, New York.

McCain, K. W. and Turner, K. (1989). Citation content analysis and aging patterns of journal articles in molecular genetics. *Scientometrics*, 17(1–2):127–163.

Moed, H. F. (2005). *Citation Analysis in Research Evaluation*. Springer.

Moravcsik, M. J. and Murugesan, P. (1975). Some results on the function and quality of citations. *Social Studies of Science*, 5:86–92.

Nanba, H., Abekawa, T., Okumura, M., and Saito, S. (2004). Bilingual PRESRI - integration of multiple research paper databases. In *Proceedings of RIAO*, pages 195–211.

Nanba, H. and Okumura, M. (1999). Towards multi-paper summarization using reference information. In *Proceedings of IJCAI*, pages 926–931.

Noreen, E. (1989). *Computer-intensive methods for testing hypotheses: An introduction*. Wiley.

Qazvinian, V. and Radev, D. R. (2008). Scientific paper summarization using citation summary networks. In *Proceedings of COLING*, pages 689–696.

Qazvinian, V., Radev, D. R., and Özgür, A. (2010). Citation summarization through keyphrase extraction. In *Proceedings of COLING*, pages 895–903.

Ritchie, A., Teufel, S., and Robertson, S. (2006). How to find better index terms through citations. In *Proceedings of Workshop on How Can Computational Linguistics Improve Information Retrieval?*, pages 25–32.

Spiegel-Rösing, I. (1977). Science studies: Bibliometric and content analysis. *Social Studies of Science*, 7:97–113.

Sugiyama, K., Kumar, T., Kan, M.-Y., and Tripathi, R. C. (2010). Identifying citing sentences in research papers using supervised learning. In *Proceedings of International Conference on Information Retrieval and Knowledge Management*, pages 67–72.

Teufel, S., Siddharthan, A., and Tidhar, D. (2006a). An annotation scheme for citation function. In *Proceedings of SIGdial Workshop on Discourse and Dialogue*, pages 80–87.

Teufel, S., Siddharthan, A., and Tidhar, D. (2006b). Automatic classification of citation function. In *Proceedings of EMNLP*, pages 103–110.

Weinstock, M. (1971). *Encyclopedia of Library and Information Science*, volume 5, chapter Citation indexes. Dekker, New York, NY.

Wilson, T., Wiebe, J., and Hoffmann, P. (2005). Recognizing contextual polarity in phrase-level sentiment analysis. In *Proceedings of HLT/EMNLP*, pages 347–354.

Semantics-Based Machine Translation with Hyperedge Replacement Grammars

Bevan JONES[1*] *Jacob ANDREAS*[2,3*] *Daniel BAUER*[3*]
Karl Moritz HERMANN[4*] *Kevin KNIGHT*[5]
(1) University of Edinburgh and Macquarie University
(2) University of Cambridge (3) Columbia University
(4) University of Oxford (5) Information Sciences Institute

ABSTRACT

We present an approach to semantics-based statistical machine translation that uses synchronous hyperedge replacement grammars to translate into and from graph-shaped intermediate meaning representations, to our knowledge the first work in NLP to make use of synchronous context free graph grammars. We present algorithms for each step of the semantics-based translation pipeline, including a novel graph-to-word alignment algorithm and two algorithms for synchronous grammar rule extraction. We investigate the influence of syntactic annotations on semantics-based translation by presenting two alternative rule extraction algorithms, one that requires only semantic annotations and another that additionally relies on syntactic annotations, and explore the effect of syntax and language bias in meaning representation structures by running experiments with two different meaning representations, one biased toward an English syntax-like structure and another that is language neutral. While preliminary work, these experiments show promise for semantically-informed machine translation.

TITLE AND ABSTRACT IN GERMAN

Semantikbasierte Maschinelle Übersetzung mit Hyperkantenersetzungsgrammatiken

Wir beschreiben einen Ansatz zur semantikbasierten statistischen maschinellen Übersetzung, der synchrone Hyperkantenersetzungsgrammatiken benutzt um in und aus graphgeformten Zwischenrepräsentationen zu übersetzen. Unseres Wissens ist dies die erste Arbeit in der natürlichen Sprachverarbeitung die synchrone kontextfreie Graphgrammatiken verwendet. Wir beschreiben Algorithmen für jeden Schritt der semantikbasierten Übersetzungskette, inklusive einem neuen Graph-zu-Wort Alinierungsalgorithmus und automatische Regelextraktionsalgorithmen für synchrone Grammatiken. Wir untersuchen den Effekt der syntaktischen Annotation auf semantikbasierte Übersetzung, indem wir zwei verschiedene Regelextraktionsalgorithmen vorstellen, einen, der lediglich semantische Annotationen erfordert und einen, der zusätzlich syntaktische Informationen verwendet. Wir untersuchen ausserdem den Einfluss von semantischen Repräsentationen die auf bestimmte Syntax und Sprache ausgerichted sind indem wir mit zwei verschiedenen Repräsentationen experimentieren: mit einer englischausgerichteten syntaxartigen Struktur und mit einer sprachneutralen Struktur. Unsere Arbeit zeigt dass semantikbasierte maschinelle Übersetzung vielversprechend ist.

[*] The authors contributed equally to this work and are listed in randomized order.

Proceedings of COLING 2012: Technical Papers, pages 1359–1376,
COLING 2012, Mumbai, December 2012.

1 Introduction

In this paper, we introduce a model for semantic machine translation using a graph-structured meaning representation. While it has been claimed since the inception of machine translation that a semantic model is necessary to achieve human-like translation (Weaver, 1955; Bar-Hillel, 1960), most recent work in MT has instead focused on phrase-based approaches. Statistical phrase-based systems rely on large volumes of parallel training data to learn translation probabilities across two languages; while, given sufficient data, phrase-based systems can cope with some of the ambiguity problems identified by early MT researchers, they are limited by the underlying assumption that surface phrases can be translated without reference to syntax or meaning. Such systems often struggle to generate correct translations that involve non-local phenomena such as argument reorderings across languages, deep embeddings, empty categories and anaphora.

With the increasing availability of syntactically-annotated data in many languages, it has become possible to more directly integrate syntax into data-driven approaches. Such syntax-based SMT systems can automatically extract larger rules, and learn syntactic reorderings for translation (Yamada and Knight, 2001; Venugopal and Zollmann, 2006; Galley et al., 2004; Chiang, 2007; Zollmann et al., 2008; DeNero et al., 2009; Shen et al., 2010; Genzel, 2010).

However, many problems remain unsolved. For illustration of a specific phenomenon difficult to capture without an intermediate meaning representation, consider the following translation example using a state-of-the-art German→English SMT system [1]:

Source	**System output**	**Reference**
Anna fehlt ihrem Kater	**Anna is missing her cat*	Anna's cat is missing her

SMT systems are frequently unable to preserve basic meaning structures (e.g. "who does what to whom") across languages when confronted with verbs that realize their arguments differently. A system using an intermediate meaning representation need not suffer from this problem. Instead of learning many bilingual translation rules over all possible realizations of this pattern, it can rely on monolingual realizations to preserve meaning in translation.

Due to the recent emergence of large, multilingual, semantically annotated resources such as OntoNotes (Hovy et al., 2006), we believe the time is ripe for data-driven, semantics-based machine translation. In this paper we present a pilot statistical, semantic machine translation system which treats MT as a two-step process of analysis into meaning in the source language, and decoding from meaning in the target language.

Our system assumes that meaning representations are directed acyclic graphs; beyond that, it is completely agnostic with respect to the details of the formalism, including the inventory of node and edge labels used. Figure 1 illustrates a pipeline via one possible graph as semantic pivot. The proposed framework is flexible enough to handle numerous existing meaning representations, including the programming language syntax of the GeoQuery corpus (Wong and Mooney, 2006) (used for the experiments in this paper), the PropBank-style structures (Palmer et al., 2005) used for the CoNLL shared task on recognizing semantic dependencies (Hajič et al., 2009), and the Elementary Dependency Structures of the LOGON corpus (Oepen and Lønning, 2006).

[1] Google Translate, 08/31/2012

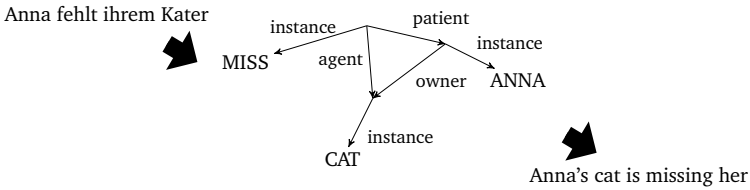


Figure 1: A string to meaning graph to string translation pipeline.

Experimental results demonstrate that our system is capable of learning semantic abstractions, and more specifically, to both analyse text into these abstractions and decode them back into text in multiple languages.

The need to manipulate graph structures adds an additional level of complexity to the standard MT task. While the problems of parsing and rule-extraction are well-studied for strings and trees, there has been considerably less work within the NLP community on the equivalent algorithms for graphs. In this paper, we use hyperedge replacement grammars (HRGs) (Drewes et al., 1997) for the basic machinery of graph manipulation; in particular, we use a synchronous HRG (SHRG) to relate graph and string derivations.

We provide the following contributions:

1. Introduction of string $\Leftrightarrow$ graph transduction with HRGs to NLP
2. Efficient algorithms for
   - string–graph alignment
   - inference of graph grammars from aligned graph/string pairs
3. Empirical results from a working machine translation system, and analysis of that system's performance on the subproblems of semantic parsing and generation.

We proceed as follows: Section 2 explains the SHRG formalism and shows how it is used to derive graph-structured meaning representations. Section 3 introduces two algorithms for learning SHRG rules automatically from semantically-annotated corpora. Section 4 describes the details of our machine translation system, and explains how a SHRG is used to transform a natural language sentence into a meaning representation and vice-versa. Section 6 discusses related work and Section 7 summarizes the main results of the paper.

2 Synchronous Hyperedge Replacement Grammars

Hyperedge replacement grammars (Drewes et al., 1997) are an intuitive generalization of context free grammars (CFGs) from strings to hypergraphs. Where in CFGs strings are built up by successive rewriting of nonterminal *tokens*, in hyperedge replacement grammars (HRGs), nonterminals are *hyperedges*, and rewriting steps replace these nonterminal hyperedges with subgraphs rather than strings.

A hypergraph is a generalization of an graph in which edges may link an arbitrary number of nodes. Formally, a hypergraph over a set of edge labels C is a tuple $H = \langle V, E, l, X \rangle$, where V is a finite set of nodes, E is a finite set of edges, where each edge is a subset of V, $l : E \to C$ is a labeling function. $|e| \in \mathbb{N}$ denotes the *type* of a hyperedge $e \in E$ (the number of nodes connected by the edge). For the directed hypergraphs we are concerned with, each edge contains a distinguished source node and one or more target nodes.

A HRG over a set of labels C is a rewriting system $G = \langle N, T, P, S \rangle$, where N and $T \subset C$ are the finite sets of nonterminal and terminal labels ($T \cap N = \emptyset$), and $S \in N$ is the start symbol. P is a finite set of productions of the form $A \rightarrow R$, where $A \in N$ and R is a hypergraph over C, with a set of distinguished *external nodes*, X_R.

To describe the rewriting mechanism, let $H[e/R]$ be the hypergraph obtained by replacing the edge $e = (v_1 \cdots v_n)$ with the hypergraph R. The external nodes of R "fuse" to the nodes of e, $(v_1 \cdots v_n)$, so that R connects to $H[e/R]$ at the same nodes that e does to H. Note that $H[e/R]$ is undefined if $|e| \neq |X_R|$. Given some hypergraph H with an edge e, if there is a production $p : l_H(e) \rightarrow R \in G_P$ and $|X_R| = |e|$, we write $H \Rightarrow_p H[e/R]$ to indicate that p can derive $H[e/R]$ from H in a single step. We write $H \Rightarrow_G^* R$ to mean that R is derivable from H by G in some finite number of rewriting steps. The grammars we use in this paper do not contain *terminal* hyperedges, thus the yield of each complete derivation is a graph (but note that intermediate steps in the derivation may contain hyperedges).

A *Synchronous Hyperedge Replacement Grammar* (SHRG) is a HRG whose productions have pairs of right hand sides. Productions have the form $(A \rightarrow \langle R, Q \rangle, \sim)$, where $A \in N$ and R and Q are hypergraphs over $N \cup T$. $\sim$ is a bijection linking nonterminal mentions in R and Q. We call the R side of a rule the source and the Q side the target. Isolating each side produces a *projection* HRG of the SHRG. In general the target representation can be any hypergraph, or even a string since string can be represented as monadic (non-branching) graphs. Because we are interested in translation between MRs and natural language we focus on graph-string SHRGs. The target projection of such a SHRG is a context free string grammar. To ensure that source and target projection allow the same derivations, we constrain the relation $\sim$ such that every linked pair of nonterminals has the same label in R and Q.

Figure 2 shows an example SHRG with start symbol $\overset{\text{ROOT}}{\text{S}}$. External nodes are shaded black.

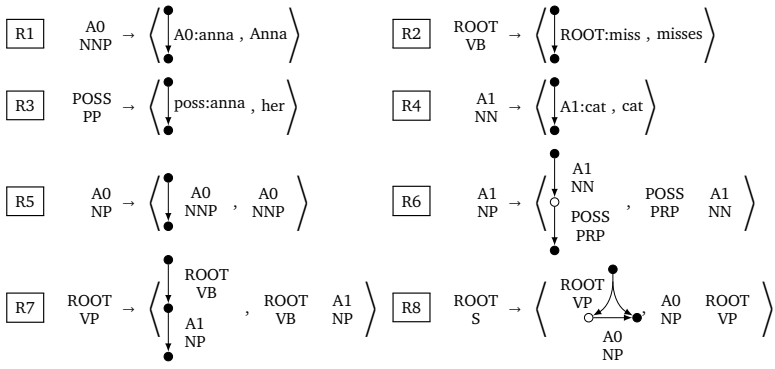


Figure 2: A graph-string SHRG automatically extracted from the meaning representation graph in figure 3a using the SynSem algorithm. Note the hyperedge in rule R8.

The graph language captures a type of meaning representation in which semantic predicates and concepts are connected to their semantic arguments by directed edges. The edges are labeled with PropBank-style semantic roles (A0, A1, poss). Nonterminal symbols in this SHRG are complex symbols consisting of a semantic and a syntactic part, notated with the former above the latter.

Since HRG derivations are context free, we can represent them as trees. As an example, Figure 3c shows a derivation tree using the grammar in Figure 2, Figure 3a shows the resulting graph and Figure 3b the corresponding string. Describing graphs as their SHRG derivation trees allows us to use a number of standard algorithms from the NLP literature.

Finally, an *Adaptive Synchronous Hyperedge Replacement Grammar (ASHRG)* is a SHRG $G = \langle N, T, P^*, S, V \rangle$, where V is a finite set of variables. ASHRG production templates are of the same form as SHRG productions, $(A \rightarrow \langle R, Q \rangle, \sim)$, but $A \in N \cup V$ and $Q, R \in N \cup T \cup V$. A production template $p^* \in P^*$ is realised as a set of rules P by substituting all variables v for any symbol $s \in N \cup T$: $P = \{\forall_{v \in V} \forall_{s \in N \cup T} p^*[v/s]\}$. ASHRGs are a useful formalism for defining canonical grammars over the structure of graphs, with production templates describing graph structure transformations without regard to edge labels. We make use of this formalism in the production template R^* in Figure 4a.

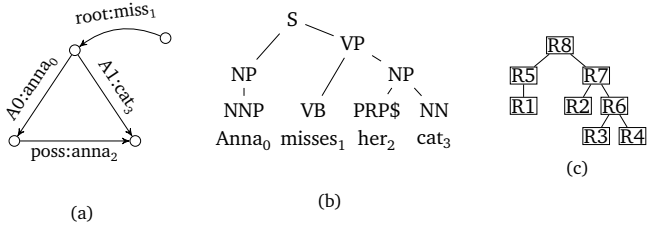


Figure 3: (a) an example meaning representation graph for the sentence *'Anna misses her cat.'*, (b) the corresponding syntax tree. Subscripts indicate which words align to which graph edges. (c) a SHRG derivation tree for (a) using the grammar Figure in 2.

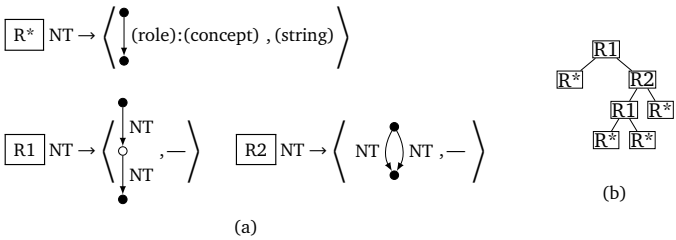


Figure 4: (a) The canonical grammar of width 2. R^* is a production template and values in parentheses denote variables as defined by the ASHRG formalism. (b) A SHRG derivation tree for the MR graph in Figure 3a using the canonical grammar in a, as created by the CanSem algorithm.

3 Learning Grammars from Annotated Data

3.1 Aligning Strings and Graphs

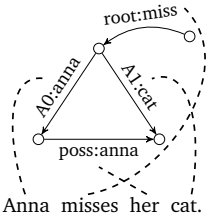


Figure 5: Edge-word alignment example.

Like much of SMT, alignments lie at the center of our semantics-based approach. However, in our case the alignments are between edges of the graph and words of the string. Figure 5 illustrates such an alignment. By listing out edge labels in a linear order, the graph-to-string alignment problem reduces to ordinary token-to-token alignment (Brown et al., 1990). We experiment with two strategies: (1) IBM Model 4 (M4) as implemented in GIZA++ (Och and Ney, 2003), and (2) a novel aligner that relies on the relative structure of the MR graph and the natural language syntax.

For M4, we traverse the graph in a fixed breadth first order to get a sequence of edge labels and feed this, along with the tokenized natural language string, to GIZA++. We then use the edge label order to map the aligned edge labels back to their respective edges.

We also experiment with a novel variant of IBM alignment Model 2 (Brown et al., 1990) that we call the dependency depth based aligner (DEPDEP, or DD for short) which uses depth within the graph and the dependency analysis of the natural language as location. Since the MR and the sentence describe the same thing, it seems reasonable to assume a certain degree of shared structure. To encode this notion, we place a Gaussian distribution over the difference between the depth of the graph edge and words in the dependency tree and weight the alignment choice by this probability. In this way, we favor aligning words to edges that are at a similar depth in the graph j to the depth of the word m in the dependency analysis.

The algorithm is concisely defined with the following equation, where a_i is the index of the edge aligned to the i^{th} word, f is a dependency parse with words f_i, e is a graph comprising edges e_i, and j_{a_i} and k_i are the edge and dependency depth of e_{a_i} and f_i, respectively.

$$p(f, a|e) = \prod_{i=1}^{n} p(j_{a_i}|k_i)p(f_i|e_{a_i}) \quad (1)$$

$$p(j_{a_i}|k_i) = \frac{\mathcal{N}(j_{a_i} - k_i|\mu, \sigma)}{\sum_{j'} \mathcal{N}(j' - k_i|\mu, \sigma)} \quad (2)$$

We estimate the mean μ and variance σ^2 of $p(j_{a_i}|k_i)$ with EM at the same time as the translation probabilities $p(f_i|e_{a_i})$.

3.2 Canonical Semantics Algorithm (CanSem)

Given word–edge alignments, we present two algorithms for rule extraction that both employ the same general strategy for rule extraction: They induce a single context-free derivation for each graph in the training data, and then extract rules from the aligned derivation trees and sentence spans.

Our first strategy for inducing a derivation of each training hypergraph (the "Canonical Semantics" Algorithm, or CanSem) is to specify, *a priori*, a minimal "canonical grammar" which is capable of producing every training example. A theorem by (Lautemann, 1988), proved by (Bodlaender, 1998), guarantees that a canonical grammar of width k is sufficient for graphs of maximum treewidth k. We extract a minimal grammar by incrementally increasing its width until the training data can be fully explained. The canonical grammar rules learned in this algorithm are effectively SHRG rule templates which ignore edge labels. Figure 4a shows a canonical grammar of width 2 needed to parse the graph in Figure 3a.

This grammar then allows us to immediately acquire a derivation tree given a graph alone (see Figure 4b); we can then use a standard technique (Galley et al., 2004) for acquiring a set of rules from an aligned derivation tree-string pair.

3.3 Syntactic Semantics Algorithm (SynSem)

Intuition suggests that additional linguistic information might aid in the selection of general, well-formed rules. Our second algorithm (the "Syntactic Semantics" Algorithm, or SynSem) is based on this assumption.

The procedure is described in Algorithm 1; to describe the notation, let each training example consist of (1) a sentence $S = s_1, s_2, \ldots, s_n$; (2) a constituency parse of S, defined by a set C of constituents; (3) a directed single-source connected hypergraph $H = (V, E)$; and (4) alignments $a : S \rightarrow E \cup \{\texttt{null}\}$. For convenience, denote the subspan of S contained in a constituent $c \in C$ (equivalently, the yield of c) as $S(c)$.

A constituent $c \in C$ is in the *frontier set* F if there exists some connected subgraph h of H such that $s \in S(c)$ if and only if $a(w) \in h$. Let $f(c)$ denote this h. c is in the *minimal frontier set* $\hat{F}$ if $c \in F$ and there exists no $c' \subset c \in F$.

Algorithm 1 EXTRACT-RULE

1: $R \leftarrow \emptyset$
2: **while** $\hat{F} \neq \emptyset$ **do**
3: $\quad c = pop(\hat{F})$
4: $\quad h$ is a new hyperedge with type matching $f(c)$
5: $\quad R = R \cup \{(h, f(c), c)\}$.
6: $\quad H = H[f(c)/h]$
7: $\quad S = S[S(c)/h]$
8: **end while**

On an abstract level, algorithm 1 matches minimal parse constituents to aligned graph components, and incrementally collapses these into nonterminals until the entire graph is consumed. Figure 2 shows the set of rules extracted by this algorithm from the MR graph, parse, and alignments in Figure 4.

In describing both algorithms, we have thus far assumed that every edge E is aligned to at least one word. As this is not always the case in practice, heuristics similar to those used in (Galley et al., 2006) may be used to attach the remaining edges.

4 Parsing and Translating with SHRG

4.1 Translation Pipeline

To build a translation system between two languages, we first extract an SHRG for each language from semantically annotated monolingual data using one of the algorithms from section 3. We then assign weights to each rule in the SHRG to transform it into a *probabilistic* SHRG, using one of the methods described below in Section 4.2.

Given a pair of weighted SHRGs, translation is a two-step process. First, we transform the source language string into an MR graph using the source SHRG (sometimes called "semantic parsing" or "analysis"). This is accomplished by parsing the string with the string projection of the SHRG and then applying the resultant derivation to generate the corresponding graph. The algorithm amounts to standard CKY string parsing with complexity $\mathcal{O}(n^3)$ in the size of the input.

We then transform the 1-best graph into the output string using the target SHRG (the "generation" task). This involves parsing the graph using the graph projection of the SHRG and then constructing the corresponding string yielded by the derivation. While parsing arbitrary graphs with SHRGs is NP-complete, we use a polynomial time chart parsing algorithms (which are exponential in the maximum size of the graph fragments on the rule right hand side) for connected graphs (Drewes et al., 1997).

In the case of the CanSem algorithm, we use a parser specialized for the canonical HRG which can be parsed even more efficiently in $\mathcal{O}(n^c)$, with c the maximum number of rule right hand side nodes (worst case $c = 3$ for experiments in this paper).

Finally, we rerank the natural language output by incorporating a language model (Heafield, 2011; Dyer et al., 2010). For the SynSem algorithm, this is integrated into the parsing algorithm via cube pruning (Chiang, 2007). In the CanSem algorithm reranking is performed on an k-best list of generated natural language output using a standard n-gram language model. Hypothesis meaning representations might be similarly reranked using a 'language model' defined on MR graphs, but we leave this for future work. In our experiments, language model weights were selected empirically based on initial evaluations of the development set.

4.2 Parameter Estimation

As with CFGs, there are two strategies for estimating the parameters of a probabilistic SHRG. One is to treat the derivations induced by the CanSem and SynSem algorithms as observed, and obtain maximum-likelihood estimates for the grammar by simply counting the number of times each production occurs in the training data.

The alternative approach is to employ the EM algorithm given a synchronous parse chart (Oates et al., 2003). Synchronous parsing extends graph parsing by identifying all possible derivations which yield both a specified string and a specified graph; the complexity of synchronous parsing is therefore approximately the product of string and graph parsing.

We initially tested both methods with both rule extraction algorithms on the development set.

For the CANSEM algorithm, EM with a Dirichlet prior of 0.1 performed better, whereas the SYNSEM algorithm obtained better results using counting.

5 Evaluation

5.1 Data

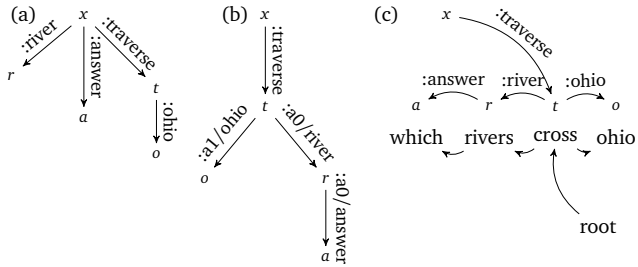


Figure 6: Two different ways of converting the GEOQUERY Prolog expression `answer(A,(river(A),traverse(A,ohio)))` to a MR graph: (a) language-neutral, (b) English-biased and (c) an illustration of how (b) matches the English dependency analysis.

Our experiments use the GEOQUERY data set (Tang and Mooney, 2001), originally a parallel corpus of 880 English questions about US geography paired with Prolog style database queries and later translated into Chinese (Lu and Ng, 2011). For English there are gold Penn Treebank-style syntax annotations as well as gold alignments pairing every word with the best predicate in the query. For Chinese, we make use of automatic parses provided by the Stanford Parser (Levy and Manning, 2003).

The database queries—expressions in an unambiguous formal language—serve as a rough encoding of sentence meaning, which we use as our meaning representation in the machine translation pipeline. Though they do not, strictly speaking, encode a linguistic notion of semantics, a statistical MT system can still learn meaningful associations with this language-independent representation. (For instance the German *'Gib mir die Bevölkerung von Kalifornien!'* [*'Give me the population of California!'*] would match the the same Prolog query as English *'How many people live in California?'*.) For input to our system, we automatically translate the Prolog expressions into graphs.

We are interested in how differences between the syntax and semantic representation might impact the translation process, and we use two different graph representations to test this. One, shown in Figure 6a (corresponding to the question '*Which rivers cross Ohio?*'), is produced by only looking at the query expression itself (GQN). The second (GQE), shown in Figure 6b is a transformation of GQN to more closely match the English syntax using the gold alignments. Note that this reshaped graph better fits the assumptions of the SYNSEM algorithm and should, in theory, produce better SHRGs for English. It is less clear whether an English biased intermediate representation would perform better for translation, as it could conceivably hurt translation to and from other languages.

We use the standard 600 train/280 test sentence split (Kwiatkowski et al., 2010), and run 10

	Prec.	Rec.	f_1
DD	**74.8**	46.9	**57.7**
M4	54.6	**53.7**	54.1

Table 1: Evaluation of English alignment, vs. gold alignments

fold cross-validation on the training data during development. We also use the standard list of named entities paired with the corresponding edges to create some fallback rules for handling previously unseen named entities.

5.2 Alignment

Table 1 shows results for the alignment algorithms described in Section 3.1 on English and the GQN MR. We report precision, recall and f_1-measure on alignment pairs. The DEPDEP algorithm (DD) performs somewhat better than IBM Model 4 (M4) with respect to f_1, and substantially better in terms of precision.

5.3 Analysis: String to Graph

We use the Smatch measure (Cai and Knight, 2012) to evaluate analysis into MR graphs, which is essentially an f_1-score on edge labels under the optimal mapping of hypothesis nodes onto reference nodes. Table 2 shows results for the analysis task in both English and Chinese. We report results of 10-fold cross validation on the training set, reserving the test data for the evaluation of the end-to-end MT system. We apply both rule extraction algorithms to the GQN data set; for English, where we have gold alignments and gold parses (which allow us to obtain DD alignments), we also vary the alignment model. We observe that the SYNSEM procedure uniformly outperforms CANSEM for analysis. These results highlight the importance of alignment quality: Gold alignments unsurprisingly lead to improved analysis, and in accordance with our expectations the syntactically-guided DD alignments appear to help SYNSEM but not CANSEM.

	CANSEM			SYNSEM		
	M4	DD	GOLD	M4	DD	GOLD
EN	67.9	56.4	72.4	81.5	81.8	84.4
ZH	67.8	–	–	76.8	–	–

Table 2: Evaluation of analysis (f_1), vs. gold MRs in development set

	CANSEM			SYNSEM		
	M4	DD	GOLD	M4	DD	GOLD
EN	51.89	48.82	55.24	52.47	42.91	53.3
ZH	50.28	–	–	45.82	–	–

Table 3: Evaluation of generation (BLEU), vs. gold strings in development set

5.4 Generation: Graph to String

We evaluate text generated from gold MR graphs using the well-known BLEU measure (Papineni et al., 2002). Table 3 shows results for English (EN) and Chinese (ZH), varying rule extraction and alignment model as before. As before, M4 alignments help CANSEM more than DD alignments; however, here the trend also carries through to SYNSEM. Also in contrast to the analysis results, the two systems perform comparably on their best English results, and CANSEM outperforms SYNSEM on Chinese.

While not targeted directly at the generation task (and not comparable to the existing literature, which reports BLEU scores on the test set), these results are promising: They are close to state-of-the-art for generation on the GEOQUERY data set, and future research might focus on optimizing generation specifically.

5.5 Translation: String to String

Finally, Table 4 shows results for the end-to-end machine translation system for English to Chinese, evaluated on the test set. We again experiment with both rule extraction algorithms in English (because DD alignments are not available for Chinese, Chinese rule extraction always uses M4).

Here CANSEM substantially outperforms SYNSEM, regardless of the data set and the choice of alignment algorithm. Also notable is the fact that the switch from GQN to GQE hurts performance with CANSEM but improves it with SYNSEM.

	CANSEM		SYNSEM	
	M4	DD	M4	DD
GQN	42.74	36.84	28.22	28.34
GQE	38.88	35.14	32.24	31.20

Table 4: Evaluation of translation (BLEU), vs. gold strings in the test set

5.6 Discussion

Several broad trends are apparent from these experimental results. The first is a partial confirmation of our hypothesis that syntactic information (in various forms) is useful in guiding the acquisition and application of semantic grammars: this is apparent in SYNSEM's gains on analysis and possibly generation, and the fact that SYNSEM performs better on GQE.

In this light the fact that CANSEM performs comparatively better on translation is somewhat surprising—we would expect overall translation results to be improved as a result of improved analysis and comparable generation. We hypothesize that the discrepancy in translation scores is due to the consistency of the grammars learned by CANSEM: Because it induces a standard derivation for MRs regardless of the source language, any incorrect rules that it learns are nonetheless shared across languages.

We also observe that the system generates many output sentences that have identical meaning but markedly different syntax and lexical choice from the reference translation (Figure 8). An inspection of the *k*-best list (Figure 7) for one translation input reveals various such candidates. This confirms that the translation pipeline works as expected: Sentences in the

Reference	what state has the sparsest population density
$k1$	what state has the least population density
$k2$	which state has the least population density
$k3$	what is the state with the smallest population
$k4$	what state has the smallest population
$k5$	what is the state with the smallest population density

Figure 7: k-best list for a sample CANSEM translations

Sample sentence	Reference
what is the density of texas	what is the population density of texas
what is the tallest mountain in america	what is the highest mountain in the us
what rivers the most running through it	which state has the most rivers running through it

(a) CANSEM

Sample sentence	Reference
give me cities with the largest population	what city has the largest population
what is the population of washington	how many people live in washington
what are in washington	how many rivers in washington

(b) SYNSEM

Figure 8: Sample translation output.

source language are analyzed into a language-independent meaning representation, and that meaning representation is then used to generate a semantically equivalent sentence in the target language.

The scores reported for SYNSEM are especially heartening in light of the fact that a standard phrase-based SMT system (Koehn et al., 2007), trained and tuned on the same corpus, achieves a BLEU score of 45.13 for ZH–EN translation. Note that the semantic translation task (at least as formulated here) is strictly harder than direct translation, as the test set contains numerous sentences annotated with identical meaning representations but different natural language realizations. We are optimistic about the potential for an extended version of the current system in which generation is conditioned on both semantics and source language.

6 Related Work

We view our semantics-based approach to MT as a continuation of recent work in statistical MT (SMT) that abstracts away from the surface string level by capturing syntactic reorderings in translation (Yamada and Knight, 2001; Gildea, 2003; Eisner, 2003; Collins et al., 2005), or using larger syntactic fragments instead of phrases (Galley et al., 2004, 2006; Chiang, 2007). These systems combine the benefits of rule-based MT and SMT by defining their translation model using syntactic translation rules from the source syntax, into the target syntax, or both. Our syntax-driven approach to rule extraction is inspired by (Chiang, 2007, 2010), while the canonical grammar approach is based on (Galley et al., 2004, 2006). However, we induce synchronous graph grammars between surface form and meaning representation, instead of transfer rules between source and target form. As with other translation work using synchronous tree grammars, such as synchronous TSG (Chiang, 2010) and synchronous TAG (DeNeefe and

Knight, 2009), our SHRGs can also be applied in both directions.

However, none of these SMT approaches use an intermediate semantic representation. A lot of research has been done in the early days of MT on translation systems using such representations (Uchida, 1987; Nirenburg, 1989; Landsbergen, 1989). These systems usually required hand-crafted rules and large knowledge bases and do not learn translation models from data automatically. Until recently, because of their good performance especially in narrow domains, rule-based MT was the predominant paradigm in deployed MT systems. In contrast, while our system adopts a semantic transfer based paradigm, we learn weighted transfer rules into and from the meaning representation automatically to build a true statistical semantics-based MT system.

In the semantic parsing literature, there are other learning based approaches to analysis into meaning representations. Zettlemoyer and Collins (2005) use an automatically induced, semantically augmented CCG and a log-linear model to parse into lambda expressions, and Ge and Mooney (2005) integrate syntactic parsing with semantic parsing for recovering Prolog queries. Lu et al. (2008) learn a generative model over tree shaped meaning representation and natural language sentences. Wong and Mooney (2006)'s WASP system is similar to ours because it draws on techniques from SMT, using word alignment algorithms to learn synchronous CFGs which translate between syntax and semantics. In fact, Jones et al. (2012) recasts many semantic parsing approaches as tree transduction, which is closely related to synchronous grammar parsing (Shieber, 2004). To our knowledge we are the first to address semantic parsing into graph-based representations as a learning task using synchronous graph grammars.

In generation, learning the parameters of statistical generation models is popular, but not much attention has been paid to the scenario where no handwritten rules exist or the mapping between semantic structure and output language is unknown in the training data (the scenario we assume in this paper). The WASP system (Wong and Mooney, 2006) can also be used as a generator. Lu and Ng (2011) automatically learn to generate English and Chinese from sentences paired with lambda calculus. Other examples are (Varges and Mellish, 2001) who learn a semantic grammar form a semantically annotated treebank automatically, and (DeVault et al., 2008) who infer a TAG for generation automatically from semantically annotated example sentences.

Formal language approaches to probabilistic tree transformation are popular (e.g. in syntax-based MT) and recently a formulation of such methods as tree transducers (Comon et al., 2007) has gained prominence in NLP (Knight and May, 2009; Graehl and Knight, 2004). In contrast, little work has been done on methods for graphs in NLP. The standard model for graph-shaped meaning representations in NLP are feature structures, which can be constructed from strings using unification grammars (Moore, 1989). However, while powerful in representation, unification grammars have unfavorable algorithmic properties, lack an intuitive probabilistic extension, and require hand-built rules. Other formal devices to accept and transduce feature structure graphs have rarely been discussed. Notable exceptions are Quernheim and Knight (2012) who discuss formal devices to accept and transduce feature structure graphs, and Bohnet and Wanner (2010) who present a toolkit for manually engineering graph-to-string transducer rules for natural language generation. We believe that we are the first to use hyperedge replacement grammars in the NLP literature and can only refer to the formal HRG survey by (Drewes et al., 1997).

7 Conclusion

We have introduced a new model for semantically-driven statistical machine translation using graph-structured meaning representations. Our approach is based on the class of weighted synchronous hyperedge replacement grammars, a rewriting formalism for graph-string pairs that intuitively extends context-free grammars. We have described unsupervised algorithms for string-graph alignment, and two algorithms for automatic SHRG learning given these alignments.

We have evaluated a semantic machine translation system on the GeoQuery data set, using grammars acquired using each of these algorithms. The results of this evaluation provide a working demonstration of the effectiveness of our machine translation model, and a characterization of the extent to which syntactic information may be used to improve the effectiveness semantic MT.

We hope that our work will motivate further research on the applications of graph grammars to basic problems in natural language processing research. Immediate extensions of the research presented here include integration of a re-ranking model for hypothesized MRs (analogous to language modeling for strings), investigation of other corpora and meaning representation formalisms, and more sophisticated probabilistic models for scoring SHRG derivations. More broadly, our results suggest that SHRGs might be an effective tool for the individual problems of semantic parsing and generation, and indeed for any phenomenon in natural language which can be represented with directed graphs.

Acknowledgments

This research was supported by ARO grant W911NF-10-1-0533 and DARPA/IPTO Contract No. HR0011-12-C-0014.

References

Bar-Hillel, Y. (1960). The present status of automatic translation of languages. In Alt, F. L., editor, *Advances in Computers*. Academic Press, New York.

Bodlaender, H. L. (1998). A partial k-arboretum of graphs with bounded treewidth. *Theor. Comput. Sci.*, 209(1-2):1–45.

Bohnet, B. and Wanner, L. (2010). Open source graph transducer interpreter and grammar development environment. In *Proceedings of the Seventh conference on International Language Resources and Evaluation*, Valletta, Malta. European Language Resources Association (ELRA).

Brown, P. F., Cocke, J., Pietra, S. A. D., Pietra, V. J. D., Jelinek, F., Lafferty, J. D., Mercer, R. L., and Roossin, P. S. (1990). A statistical approach to machine translation. *Computational Linguistics*, 16(2):79–85.

Cai, S. and Knight, K. (2012). Smatch: an evaluation metric for semantic feature structures. submitted.

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

Chiang, D. (2010). Learning to translate with source and target syntax. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, ACL '10, pages 1443–1452.

Collins, M., Koehn, P., and Kučerová, I. (2005). Clause restructuring for statistical machine translation. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, ACL '05, pages 531–540.

Comon, H., Dauchet, M., Gilleron, R., Löding, C., Jacquemard, F., Lugiez, D., Tison, S., and Tommasi, M. (2007). *Tree Automata Techniques and Applications*. INRIA.

DeNeefe, S. and Knight, K. (2009). Synchronous tree adjoining machine translation. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 2-Volume 2*, pages 727–736. Association for Computational Linguistics.

DeNero, J., Pauls, A., and Klein, D. (2009). Asynchronous binarization for synchronous grammars. In *Proceedings of the ACL-IJCNLP 2009 Conference Short Papers*, ACLShort '09, pages 141–144.

DeVault, D., Traum, D., and Artstein, R. (2008). Practical grammar-based NLG from examples. In *Proceedings of the Fifth International Natural Language Generation Conference*, INLG '08, pages 77–85.

Drewes, F., Habel, A., and Kreowski, H. (1997). Hyperedge replacement graph grammars. In Rozenberg, G., editor, *Handbook of Graph Grammars and Computing by Graph Transformation*, pages 95–162. World Scientific.

Dyer, C., Lopez, A., Ganitkevitch, J., Weese, J., Ture, F., Blunsom, P., Setiawan, H., Eidelman, V., and Resnik, P. (2010). cdec: A decoder, alignment, and learning framework for finite-state and context-free translation models. In *Proceedings of ACL*.

Eisner, J. (2003). Learning non-isomorphic tree mappings for machine translation. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics-Volume 2*, pages 205–208. Association for Computational Linguistics.

Galley, M., Graehl, J., Knight, K., Marcu, D., DeNeefe, S., Wang, W., and Thayer, I. (2006). Scalable inference and training of context-rich syntactic translation models. In *ACL 2006*.

Galley, M., Hopkins, M., Knight, K., and Marcu, D. (2004). What's in a translation rule. In *Proceedings of HLT/NAACL*, volume 4, pages 273–280. Boston.

Ge, R. and Mooney, R. J. (2005). A statistical semantic parser that integrates syntax and semantics. In *Proceedings of the Ninth Conference on Computational Natural Language Learning*, CONLL '05, pages 9–16. Association for Computational Linguistics.

Genzel, D. (2010). Automatically learning source-side reordering rules for large scale machine translation. In *Proceedings of the 23rd International Conference on Computational Linguistics*, pages 376–384. Association for Computational Linguistics.

Gildea, D. (2003). Loosely tree-based alignment for machine translation. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics-Volume 1*, pages 80–87. Association for Computational Linguistics.

Graehl, J. and Knight, K. (2004). Training tree transducers. In *HLT-NAACL*, pages 105–112.

Hajič, J., Ciaramita, M., Johansson, R., Kawahara, D., Martí, M. A., Màrquez, L., Meyers, A., Nivre, J., Padó, S., Štěpánek, J., Straňák, P., Surdeanu, M., Xue, N., and Zhang, Y. (2009). The conll-2009 shared task: Syntactic and semantic dependencies in multiple languages. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL 2009): Shared Task*, pages 1–18, Boulder, Colorado. Association for Computational Linguistics.

Heafield, K. (2011). KenLM: Faster and smaller language model queries. In *Proceedings of the Sixth Workshop on Statistical Machine Translation*, Edinburgh, UK. Association for Computational Linguistics.

Hovy, E., Marcus, M., Palmer, M., Ramshaw, L., and Weischedel, R. (2006). OntoNotes: the 90% solution. In *Proceedings of the Human Language Technology Conference of the NAACL, Companion Volume: Short Papers*, pages 57–60. Association for Computational Linguistics.

Jones, B., Johnson, M., and Goldwater, S. (2012). Semantic parsing with bayesian tree transducers. In *ACL 2012*.

Knight, K. and May, J. (2009). Applications of weighted automata in natural language processing. In *Handbook of Weighted Automata*. Springer.

Koehn, P., Hoang, H., Birch, A., Callison-Burch, C., Federico, M., Bertoldi, N., Cowan, B., Shen, W., Moran, C., Zens, R., Dyer, C., Bojar, O., Constantin, A., and Herbst, E. (2007). Moses: open source toolkit for statistical machine translation. In *Proceedings of the 45th Annual Meeting of the ACL on Interactive Poster and Demonstration Sessions*, ACL '07, pages 177–180.

Kwiatkowski, T., Zettlemoyer, L., Goldwater, S., and Steedman, M. (2010). Inducing probabilistic CCG grammars from logical form with higher-order unification. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 1223–1233. Association for Computational Linguistics.

Landsbergen, J. (1989). The Rosetta project. In *Machine Translation Summit II*, Munich, Germany.

Lautemann, C. (1988). Decomposition trees: Structured graph representation and efficient algorithms. In *Proceedings of the 13th Colloquium on Trees in Algebra and Programming*, CAAP '88, pages 28–39, London, UK, UK. Springer-Verlag.

Levy, R. and Manning, C. D. (2003). Is it harder to parse Chinese, or the Chinese Treebank? In *Proceedings of ACL 2003*, pages 439–446.

Lu, W., Ng, H., Lee, W., and Zettlemoyer, L. (2008). A generative model for parsing natural language to meaning representations. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 783–792. Association for Computational Linguistics.

Lu, W. and Ng, H. T. (2011). A probabilistic forest-to-string model for language generation from typed lambda calculus expressions. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 1611–1622. Association for Computational Linguistics.

Moore, R. C. (1989). Unification-based semantic interpretation. In *Proceedings of ACL 1989*, pages 33–41.

Nirenburg, S. (1989). New developments in knowledge-based machine translation. In Alatis, J. E., editor, *Georgetown University Round Table on Languages and Linguistics 1989: "Language teaching, testing, and technology: lessons from the past with a view toward the future"*, pages 344–357. Georgetown University Press.

Oates, T., Doshi, S., and Huang, F. (2003). Estimating maximum likelihood parameters for stochastic context-free graph grammars. In *Inductive Logic Programming: 13th International Conference*, pages 281–298.

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

Oepen, S. and Lønning, J. (2006). Discriminant-based MRS banking. In *5th International Conference on Language Resources and Evaluation*.

Palmer, M., Gildea, D., and Kingsbury, P. (2005). The proposition bank: An annotated corpus of semantic roles. *Computational Linguistics*, 31(1):71–106.

Papineni, K., Roukos, S., Ward, T., and Zhu, W.-J. (2002). BLEU: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, ACL '02, pages 311–318.

Quernheim, D. and Knight, K. (2012). Towards Probabilistic Acceptors and Transducers for Feature Structures. In *Proceedings of the Sixth Workshop on Syntax, Semantics and Structure in Statistical Translation*.

Shen, L., Xu, J., and Weischedel, R. (2010). String-to-dependency statistical machine translation. *Computational Linguistics*, 36(4):649–671.

Shieber, S. M. (2004). Synchronous grammars as tree transducers. In *Proceedings of TAG+7*, pages 88–95.

Tang, L. R. and Mooney, R. J. (2001). Using multiple clause constructors in inductive logic programming for semantic parsing. In *12th European Conference on Machine Learning*.

Uchida, H. (1987). ATLAS: Fujitsu machine translation system. In *Machine Translation Summit*, Japan.

Varges, S. and Mellish, C. (2001). Instance-based natural language generation. In *Proceedings of the second meeting of the North American Chapter of the Association for Computational Linguistics on Language technologies*, pages 1–8. Association for Computational Linguistics.

Venugopal, A. and Zollmann, A. (2006). Syntax augmented machine translation via chart parsing. In *Proceedings on the Workshop on Statistical Machine Translation. New York City: Association for Computational Linguistics*, pages 138–141.

Weaver, W. (1955). Translation. In *Machine translation of languages*, volume 14, pages 15–23. MIT Press, Cambridge, MA.

Wong, Y. W. and Mooney, R. J. (2006). Learning for semantic parsing with statistical machine translation. In *Proceedings of the main conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics*, HLT-NAACL '06, pages 439–446.

Yamada, K. and Knight, K. (2001). A syntax-based statistical translation model. In *Proceedings of the 39th Annual Meeting on Association for Computational Linguistics*, ACL '01, pages 523–530.

Zettlemoyer, L. and Collins, M. (2005). Learning to map sentences to logical form: Structured classification with probabilistic categorial grammars. In *Proceedings of the Twenty-First Conference Annual Conference on Uncertainty in Artificial Intelligence (UAI-05)*, pages 658–666, Arlington, Virginia. AUAI Press.

Zollmann, A., Venugopal, A., Och, F., and Ponte, J. (2008). A systematic comparison of phrase-based, hierarchical and syntax-augmented statistical MT. In *Proceedings of the 22nd International Conference on Computational Linguistics - Volume 1*, COLING '08, pages 1145–1152.

Answering Yes/No Questions via Question Inversion

Hiroshi Kanayama[1] *Yusuke Miyao*[2] *John Prager*[3]

(1) IBM Research - Tokyo, Koto-ku, Tokyo, Japan
(2) National Institute of Informatics, Chiyoda-ku, Tokyo, Japan
(3) IBM T.J.Watson Research Center, Yorktown Heights, NY, USA
hkana@jp.ibm.com, yusuke@nii.ac.jp, jprager@us.ibm.com

Abstract

This paper investigates a solution to yes/no question answering, which can be mapped to the task of determining the correctness of a given proposition. Generally it is hard to obtain explicit evidence to conclude a proposition is *false* from an information source, so we convert this task to a set of factoid-style questions and use an existing question answering system as a subsystem. By aggregating the answers and confidence values from a factoid-style question answering system we can determine the correctness of the entire proposition or the substitutions that make the proposition false. We evaluated the system on multiple-choice questions from a university admission test on world history, and found it to be highly accurate.

Keywords: question answering, facts validation, yes/no question, question inversion.

Proceedings of COLING 2012: Technical Papers, pages 1377–1392,
COLING 2012, Mumbai, December 2012.

1 Introduction

Yes/no question answering (Green and Carberry, 1994; Hirschberg, 1984) can be equated to the task of determining the correctness of a given proposition. The target of such a mechanism is not limited to explicit interrogative questions since any general declarative sentences can in principle be validated. For example, consider the following two propositions, where (1) is true and (2) false[1]:

(1) Chirac was the president of France in 2000.

*(2) Chirac was the president of Germany in 2000.

As suggested by this example, a false proposition can often be produced by replacing a part of a true proposition. During a conversation, a human with knowledge of the facts might not only say that the utterance (2) is wrong, but respond with something like "not Germany, but France." Therefore we believe this kind of validation system we are proposing here may have application outside of a strict question-answering framework.

In spite of the importance of the process, yes/no-style question answering has not been intensively studied, compared to other type of question answering, such as the factoid-style question answering (Ravichandran and Hovy, 2002; Bian et al., 2008) and definition question answering (Xu et al., 2003; Cui et al., 2005). Even though there are only two possible answers, yes or no, such questions can be quite hard to answer. Search technologies cannot be applied easily because in many cases we can not rely on the existence of explicit negative evidence in the information source, *e.g.* "Chirac was *not* a president of Germany."

This paper tackles yes/no question answering by exploiting an existing system for factoid-style question answering. The key idea, inspired by *question inversion* (Prager et al., 2006), involves generation of factoid questions by replacing some parts of a given proposition with abstract (*i.e.* ungrounded) expressions. For example, to determine the correctness of (1) and (2), two question sentences are generated for each proposition, with abstracted parts in italics and *anticipated answers* in brackets[2].

(1a) *He* was the president of France in 2000. [Chirac]

(1b) Chirac was the president of *this country* in 2000. [France]

(2a) *He* was the president of Germany in 2000. *[Chirac]

(2b) Chirac was the president of *this country* in 2000. *[Germany]

DeepQA (Ferrucci, 2012), the factoid-style question answering system used here accepts these sentences with focal words (*e.g.* nouns with 'this', or pronouns such as 'he') as input questions, generates a hit-list of candidate answers, and assigns confidence values to candidate answers. Usually the system's output to the question is the top-ranked answer, the answer with the highest confidence value. Our hypothesis is that if an original proposition

[1] '*' denotes a false proposition.

[2] '*' before brackets denotes the anticipated answer should not be answered because the original proposition is false.

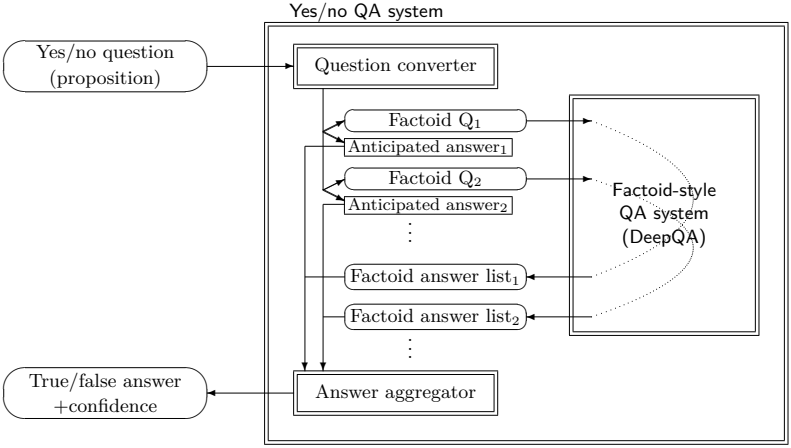


Figure 1: The flow of yes/no question answering using the factoid-style question answering.

is true, then the anticipated answer should be the top-ranked answer from DeepQA, but for false propositions, the anticipated answer should either have a low confidence value, or not be ranked high, or both. Since DeepQA outputs "France" for (1b) and (2b) as the top-ranked answer, we can use that as positive evidence for proposition (1), and negative evidence for proposition (2). The exact formula for processing the results of the generated questions is discussed in Section 4.

There is no natural repository of true and false facts which can be used for evaluation of the fact validation system. It is even difficult to artificially generate wrong facts from the set of correct facts without bias. Therefore we chose to use questions of multiple-choice of true (or false) statements in a university admission test as a reliable benchmark. In this context, wrong statements are carefully produced by the examining board by mixing a correct statement with partially wrong information.

The main contribution of this paper is to provide a novel framework for benchmarking and discussion of a method of processing yes/no questions, and propose a method to appropriately make use of existing technologies of factoid-style question answering and its underlying information sources.

Figure 1 illustrates the overall architecture to solve yes/no questions with an existing factoid-style question answering system, DeepQA. The yes/no question answering system takes a proposition as an input and outputs the correctness of the proposition with a confidence value. DeepQA is used as a subsystem, which takes a factoid-style question as an input and outputs an answer list with confidence values.

The key methods are conversion from yes/no questions into factoid-style questions, and aggregation of the results of factoid answers to determine yes or no. Rather than the implementation of fully automatic system, this study focuses on the methodologies and the headroom analysis of this fact validation. Thus the conversion of questions is done

manually, but it is conducted by using specific algorithms and can be automated using standard techniques and resources. The subsequent stages are automatic and accuracies of several methods will be compared.

Section 2 describes some of the prerequisites for the discussion in this paper, the yes/no type question answering, the existing factoid-style question-answering system, the original idea of question inversion, and the Japanese university admission test that was used for the evaluations. Section 3 shows the method of question conversion and problems in question generation. Section 4 describes the answer aggregation module, which selects the most appropriate answer lists for the fact validation. The experimental results appear in Section 5, and Section 6 provides the further discussion on the results and comparison with another approach. Section 7 concludes this paper.

2 Background

2.1 Yes/no question answering

From linguistic point of view, a yes/no question is defined as an interrogative sentence that is answered by either yes or no. A yes/no question in English language is often generated by placing a positive or negative form of an auxiliary verb (*e.g.* does, can, isn't, etc.) before the subject.

However, yes/no questions are not always correctly answered by just one of two answers, yes or no. Hirschberg (Hirschberg, 1984) pointed out that yes/no questions can have several functions, including *scalar implicature* such as (3Q), where the answer (3A) makes the correction to the proposition in the question.

(3Q) Did you invite Mary?

(3A) I invited Bob.

Green et al. (Green and Carberry, 1994) investigated such indirect answers (*i.e.* other than yes/no) to yes/no questions from discourse representation based on Rhetorical Structure Theory (Mann and Thompson, 1987). They provided an algorithm to generate appropriate answers that fill the gap between the question and the discourse contexts. The process is designed to provide the automatic responses in a dialog system.

This paper aims at fact validation of propositions, so the main outputs are yes or no. Though the initial input in our test domain is not in the form of interrogative question but declarative expressions, the intrinsic task is the same; indeed we will show that our approach has the capability of suggesting the true fact from a factually incorrect input.

2.2 Factoid-style question answering system

We use an open-domain factoid-style question answering system, DeepQA (Ferrucci, 2012), which returns multiple answers with each confidence value ranging from 0 to 1. DeepQA first analyzes the input question sentence, identifies its answer type amongst other question processing, generates candidate answers from searches performed corpora including encyclopedia, news paper articles and thesauri, gathers evidences for each candidate answer by evaluating a collection of feature scorers on it, and calculates each candidate's final confidence value by accumulating these scores and weighting them via a model of learned on trained data.

In the candidate answer generation phase, the candidates are not restricted to terms which have the answer type determined by question analysis, but any term found in returned documents which passes certain filtering criteria can be selected as a candidate. The coincidence between the type of the candidate and the type of question is used as one of the features that support the candidate answer to be correct.

A confidence value is calculated using multiple features computed from some analysis that the candidate answer should be correct. Each feature type is weighed by a model output from a machine learning process trained on a repository of several thousand question-answer pairs. The confidence values, calculated independently for each candidate, estimate the likelihood to be the correct answer (so the sum of confidence values across all answers may exceed 1, reflecting the fact that multiple answers may be correct). Thus there is a strong correlation between the confidence value and the precision of the answer.

According to (unpublished) experiments, DeepQA achieved an accuracy of 67% on TREC 2002 (Voorhees, 2002) question set, trained using about 8,000 QA pairs from other years of TREC and another data source.

2.3 Question Inversion

Question inversion (Prager et al., 2006) is a method to generate new questions from a different point of view to rerank the candidate answers in factoid-style question answering. It can be used with any QA system. For example, when a QA system solves question (4),

(4) What was the capital of Germany in 1985?

the system first generates several candidate answers, like "Berlin", "Moscow", etc. The next step is to generate the abstract inverted question (4i), by removing Germany, the *pivot term*.

(4i) Of what country was *CANDIDATE* the capital in 1985?

A series of grounded inverted questions is generated by replacing *CANDIDATE* in turn with each of the candidate answers to (4i) (*e.g.* "Of what country was Berlin the capital in 1985?"). These inverted questions are processed by the QA system, and the "inverted" answers (*e.g.* "Germany", "Soviet Union" etc.) along with their scores are gathered. Original candidate answers ("Berlin", "Moscow") are given credit when the corresponding inverted question generates the pivot ("Germany", in this case). The candidate's final score is calculated as a function of its original score and the credit from inversion.

2.4 Examination for university admission

For this study we use the National Center Test for University Admission[3], which is the standardized achievement test for high school students who wish to enter universities in Japan. The examination on world history was selected because most of the questions are solvable as fact-validation problems with general knowledge. The questions are provided in a multiple-choice style as exemplified in Table 1, which shows the description of the question and the four candidate statements. In this case only one of four is a correct statement,

[3]http://www.dnc.ac.jp/

From 1-4 below, choose the one sentence that is correct in regard to the person/people that it describes.	
1	Ouyang Xiu and Su Shi were the best known writers of the Tang dynasty.
2	Yen Chen-ching was the best known chirographer in the Sung dynasty.
3	Wang Anshi of the Sung era implemented the reform called the New Policies.
4	Qin Hui fought with the war hawks on the relationship with Yuan.

Table 1: An example of questions in the world-history examination (from 2009 National Center Tests on World History). The correct answer is 3.

and the others are incorrect, so we can frame the individual choices as test cases for fact validation. The original examinations are provided in Japanese, and this paper used their English translation, which was translated by a native speaker with domain knowledge.

This particular examination is also used for evaluating a task of recognition of textual entailment (Miyao et al., 2012; Shima et al., 2011). These true or false statements can be test cases of hypotheses which can be entailed from relevant texts found in encyclopedia articles. The multiple-choice solution using textual entailment will be compared to our approach in Section 6.2.

3 Question conversion

The question conversion module generates multiple factoid-style questions to be input to DeepQA from the original proposition. As described in Section 1, because the component to perform the inversion was not completed in this time, this module's function is performed manually in this study so that we can discuss the intrinsic features of yes/no question answering, but this section gives algorithms for each operation.

3.1 Conversion method

First, *pivot terms*, the key entities to be abstracted are selected from the original proposition. While the original work chose a single pivot per question, there is nothing about the question inversion process that requires there be only one pivot/abstract inverted question per input question, and in fact we use multiple. Since most of questions in the test domain contain proper nouns, often multiple proper nouns, we decided to use proper nouns as pivot terms. A back-off method of choosing common nouns is employed when there are no proper nouns. For example, two pivot terms (underlined) are found in proposition (5), which is the second row in Table 1.

(5) Yen Chen-ching was the best known chirographer in the Sung dynasty.

Next, a *type* is determined for each pivot term. For this algorithm, we define a type to be a common noun which is a hypernym of the pivot term. A thesaurus such as WordNet can be used, but choosing the right level of abstraction can be a problem. An effective approach to making this selection involves mining the corpus to determine co-occurrence counts with each possible hypernym (Prager et al., 2001). A simpler solution, which we adopt here as a guideline, is to select the type from the first sentence of Wikipedia articles about the subject term, from the observation that many opening sentences are of the form "X is Y" where X is the title of the article and Y is its type (Kazama and Torisawa, 2007). In the

example above, “calligrapher” and “dynasty” are assigned as types of “Yen Chen-ching” and “Sung dynasty”, respectively.

The pivot term is then replaced with a noun phrase consisting of “this” and the type. When the type is Person and the pivot term is the subject of the head predicate of the proposition, just “he” or “she” is used. From (5), two factoid questions (5a) and (5b) are generated, the substitutions denoted by italic.

(5a) *He* was the best known chirographer in the Sung dynasty.

(5b) Yen Chen-ching was the best known chirographer in *this dynasty*.

For each question there is the *anticipated answer*, that corresponds to the pivot term in the original proposition. The questions are input to DeepQA and the results will be compared to the anticipated answers, “Yen Chen-ching” for (5a) and “Sung dynasty” for (5b).

This idea of generation of new questions is inspired by question inversion, described in Section 2.3. However, the method in this paper is different from the original question inversion (Prager et al., 2006) mainly in three points:

Multiple inverted questions. Instead of a single pivot term, multiple pivot terms are selected and multiple questions are generated in this study. All of them are input to DeepQA and the results are aggregated in a subsequent module. This usage of multiple questions makes the reliance on DeepQA (both its competence and the coverage of the test domain in its corpora) less brittle.

Generation of sentences. In our method the inverted sentences are generated as natural-language questions, instead of internal structures for a specific system. This process is more general.

No need of candidate answers. When question inversion is used for answer reranking in factoid-style question answering, candidate answers must be generated to fill the slot of the original question. In our method, possible answers for a yes/no question are just yes or no, thus traditional candidate generation is not required in the question conversion module.

The questions generated by the method above are input to DeepQA, and the multiple answers with confidence values are returned for each question. Table 2 shows an example.

Ideally the anticipated answer is at the top of the DeepQA’s answer list when the original proposition is correct, although, we cannot rely on this since DeepQA is not perfect. For example, the third question in Table 2 is an unlucky case: the original proposition is correct, but the anticipated answer was at the second place in DeepQA’s answer list, due to complexity of the responsive text in the corpus. The anticipated answer was not returned for the fourth question in Table 2. This problem will be addressed by the answer aggregation module described in Section 4.

	Generated questions	Anticipated answer	1st	2nd	3rd
q_1	In *this country*, Xuanzang traveled to India and brought home the Buddhist scriptures during the Tang period.	[China]	China 0.702	Nepal 0.513	Burma 0.129
q_2	In China, *he* traveled to India and brought home the Buddhist scriptures during the Tang period.	[Xuanzang]	Xuanzang 0.593	Tang 0.147	Buddhism 0.11
q_3	In China, Xuanzang traveled to *this country* and brought home the Buddhist scriptures during the Tang period.	[India]	monk 0.23	India 0.216	Faxian 0.135
q_4	In China, Xuanzang traveled to India and brought home the Buddhist scriptures during *this period*.	[Tang period]	Buddhism 0.441	monk 0.219	Faxian 0.127

Table 2: Examples of answer lists output by DeepQA, for the original (true) proposition "In China, Xuanzang traveled to India and brought home the Buddhist scriptures during the Tang period." The numbers under the answers indicate the confidence values.

3.2 Problematic questions

Besides failures by DeepQA due to just lack of information in the corpus or inability to process it correctly, there are intrinsic problems in the use of DeepQA as the evidence of a fact validation. We identified two major problematic phenomena: *multiple-answer questions* (*i.e.* questions with more than one answer) which may not return the anticipated answer as the first answer for a true proposition, and *attributive questions* which may produce the anticipated answer for a false proposition.

Multiple-answer questions. Proposition (1) in Section 1 has enough context in it to be correctly validated as-is; however, suppose the phrase "in 2000" were dropped, producing proposition (6), with inverted questions (6a) and (6b).

(6) Chirac was the president of France.

(6a) *He was the president of France.* $^{?}$[Chirac]

(6b) Chirac was the president of this country. [France]

(6b) is not problematic, but (6a) has several possible answers ("Mitterand", "Chirac", "Sarkozy", "Hollande", ...), so even if the original proposition (6) is true, "Chirac" may not be the DeepQA's first answer to (6a).

Attributive questions. Proposition (7) below is false. Structurally, (7) asserts two properties of an entity ("Carlos I"), and whether one uses said entity or one of the properties as a pivot can have very different consequences. Using the entity as a pivot is generally a good choice in such situations, since the two (or however many are present) properties can together triangulate a unique answer. This does not happen so readily when a property is chosen as a pivot.

*(7) Carlos I, the King of Spain, was also the King of Portugal.

(7a) He, the King of Spain, was also the King of Portugal. *[Carlos I]

(7b) *Carlos I, the King of this country, was also the king of Portugal.* *?[Spain]

(7a) is confirmed to be a desirable inverted question because DeepQA actually returns "Philip II of Spain" as the first answer. This generates a correct statement, knowledge of which is the point of the question. However, if (7b) is used as an inverted question, "Spain" is returned by DeepQA as the first answer. This is because there is no country which is more associated with "Carlos I" than "Spain", regardless of other contexts in the sentence. This suggests that the kind of pivot term which is an attribute of a specific named entity may not be suitable for generation of inverted questions. This issue will be addressed in the answer aggregation phase discussed in Section 4.

4 Aggregation of answers

This section describes the answer aggregator which processes the multiple answer lists from the generated questions and decides the correctness of the input proposition.

4.1 Answer matching

The initial operation by the answer aggregator is to compare the anticipated answer and the answer lists output by DeepQA. Here we introduce a function $A(q)$, which stands for the rank where the anticipated answer appears in the answer lists for a question q. If the anticipated answer is not found in the answer list for q, $A(q) = \text{nil}$. For example, the third question in Table 2 is expressed as $A(q_3) = 2$.

There is clearly an important signal in whether the anticipated answer appears on the top of the answer list. When the first answer is the anticipated answer (*i.e.* $A(q) = 1$) it will provide evidence that the original proposition is true. On the other hand, when the first answer is different from the anticipated answer (*i.e.* $A(q) \neq 1$), it provides evidence that the original proposition is false.

Given the importance of this test, the matching of the anticipated answer and each candidate answer should be more robust than exact match. We relax the matching criteria in the following ways:

- Case insensitive matching.
- The existence of type name is optional (*e.g.* "Sung dynasty" and "Sung" can match).
- Synonyms, or transliteration variations (*e.g.* "Sung dynasty" and "Song dynasty") can match. In this study a synonym list is created manually by seeing the pairs of the anticipated answers and the first answers of DeepQA, without seeing the result. This can be replaced by an automatic process, such as using Wikipedia redirects.

4.2 Question/answer selection

Now the question is how to interpret multiple answer lists output by DeepQA to determine the correctness of a given proposition. An ultimate goal of this work is to develop an algorithm that combines the support for/against the truth of the different propositions

from all of the inverted questions used. However, we have found that selecting just one question per proposition requires a simpler algorithm, and can give good results.

For simplicity, we use only the top-ranked answer from the desired inverted question, that is, the system outputs true if $A(q) = 1$ and outputs false if $A(q) \neq 1$. Therefore the problem is reduced to selection of q^*, the question for which DeepQA generates the most representative answer list, from the set of questions Q. If we define $R(q)$ to be the *reliability* of question q, then the decision of true or false of the given proposition is formalized as

$$TF(Q) = \begin{cases} \mathsf{true} & A(q^*) = 1 \\ \mathsf{false} & otherwise \end{cases} \\ where \quad q^* = \underset{q \in Q}{argmax}(R(q)) \tag{8}$$

Then the problem is the design of the reliability function $R(q)$. The basic idea is the use of confidence value, relying on the correlation between the confidence value and the precision of the answer in DeepQA's outputs. Define

$$R_1(q) = C(q, 1) \tag{9}$$

where $C(q, n)$ is DeepQA's confidence value for n-th answer to question q. In Table 2, $q^* = q_1$ because q_1 has the highest confidence value for its first answer, then $TF(Q) = \mathsf{true}$ because q_1's first answer, "China" matches the anticipated answer. The correct answer is true, so it shows the method works well for the decision.

Now remember that in Section 3.2 we found two types of questions that are not suitable for fact validation. To consider multiple-answer questions, we can penalize questions where the first answer and second answer have similar confidence values. We revise (9) to

$$R_2(q) = \begin{cases} p_2 R_1(q) & \frac{C(q,2)}{C(q,1)} > \theta_2 \\ R_1(q) & otherwise \end{cases} \tag{10}$$

where p_2 is a penalty value less than 1.0, and θ_2 is a threshold value between 0.0 and 1.0 to estimate whether q is a multiple-answer question.

To take attributive questions into consideration, another penalty value should be multiplied to $R_2(q)$, when the pivot term of q is an *indirect entity*, which is one of the followings:

- A constituent of a phrase modifying other proper noun or content noun (*e.g.* "Akbar of the Mughal Empire")
- Appositive modifiers (*e.g.* "Tenochtitlan, the capital of Aztec empire")

The updated formula is (11). For example, the pivot term of question (7b) meets the second condition above, so the reliability of the question will be penalized with p_3.

$$R_3(q) = \begin{cases} p_3 R_2(q) & q\text{'s pivot term is indirect entity} \\ R_2(q) & otherwise \end{cases} \tag{11}$$

Besides $TF(Q)$, the decision of true or false, the system returns the confidence value $C^*(Q)$ as well, simply defined as $C^*(Q) = C(q^*, 1)$ *i.e.* DeepQA's confidence value of the first answer for the most reliable question.

Number of proper nouns	0	1	2	3	4
Frequency	0	1	53	48	2

Table 3: Distribution of number of proper nouns in statements of the development set.

5 Experiments

5.1 Experimental setup

For the evaluation of yes/no question answering, we used two sets of world history examination data from National Center Test for University Admission, the set in the year of 2009 for development data, and set in the year of 2007 for open test data. They contain 26 and 23 multiple-choice questions each of which has four sentential statements, thus 104 and 92 predicates can be used for true/false validation as development and test data, respectively. 84% of questions require the answerer to choose a correct statement out of four, and rest of them are to choose an incorrect statement out of four.

Some statements need preprocessing to be self-contained statements, because sometimes necessary information is found in the description of the question, instead of the statements to be chosen. This preprocessing is same as that in the use of the same data for textual entailment (Miyao et al., 2012). For example, when a question description says "choose the most appropriate sentence concerning events that occurred during the period of Ming dynasty", the statement (12) need to be updated to (12m) to determine its correctness.

(12) A Buddhist sect called Zen was created.

*(12m) A Buddhist sect called Zen was created during the period of Ming dynasty.

Table 3 shows the distribution of number of proper nouns in statements of the development set. This shows that most of statements have multiple proper nouns to be ungrounded to produce multiple inverted questions.

The main metrics of this experiment is the accuracy of true/false determination, and we also evaluated the accuracy of 4-way multiple-choice questions, which actually examinees answer. To answer such question types, we introduce the *correctness score* $CS(Q)$ defined as (13).

$$CS(Q) = \begin{cases} C^*(Q) & TF(Q) = \mathsf{true} \\ (-1)C^*(Q) & TF(Q) = \mathsf{false} \end{cases} \tag{13}$$

With this score, "select the correct statement" questions can be solved by selecting the statement with the highest $CS(Q)$, that is, the proposition judged true with the highest confidence value, or the one with the lowest confidence when all propositions are judged false. Also "select the wrong statement" questions can be answered by just selecting the lowest $CS(Q)$.

5.2 Experimental results

First we evaluated the accuracies of true/false determination and 4-way selection using the development data (examination in the year of 2009). Baseline for true/false evaluation is

Method	true/false		p	4-way	
Baseline	65.4%	(68/104)		25%	
Model 1	77.8%	(81/104)	.043	50%	(13/26)
Model 2	79.8%	(83/104)	.029	58%	(15/26)
Model 3	81.7%	(85/104)	.017	62%	(16/26)

Table 4: The result of closed test with the development set (2009).

Method	true/false		p	4-way	
Baseline	68.4%	(63/92)		25%	
Model 1	69.6%	(64/92)	.500	57%	(13/23)
Model 2	71.7%	(66/92)	.418	57%	(13/23)
Model 3	79.3%	(73/92)	.046	65%	(15/23)

Table 5: The result of open test with 2007 question set.

the accuracy when returning always false, and baseline for 4-way evaluation is the random choice (theoretically 25%). The proposed method was evaluated with three models. Model 1 uses $R_1(q)$ in equations (8) and (9). Model 2 addresses the problem of multiple-answer questions with $R_2(q)$ in equation (10), where both p_2 and θ_2 were empirically set to 0.5. Model 3 cares also attributive questions with $R_3(q)$ in equation (11). p_3 is empirically set to 0.1 here.

Table 4 shows the results. According to p-value between the baseline and each model in the true/false determination, all models significantly outperformed the baseline. We set parameters so that Model 3 shows the highest accuracies, but the differences amongst the three models were not significant, due to the small test set.

Table 5 shows the result using the test data (the examination in 2007), using parameters set from the development data. Though Model 1 and Model 2 did not show a significant difference from the baseline in terms of the yes/no determination, Model 3 shows highest accuracy and significantly outperformed both the baseline and Model 1. In the 4-way test, all models apparently outperformed the random choice.

The results with the test set showed that referring to both the multiple-answer questions and attributive questions can be appropriately discouraged. For example, when a wrong statement (14) is converted to (14a), the country which ruled Vardhana dynasty, India, was returned as the first answer by DeepQA with high confidence, regardless of the role of Angkor Wat. When Model 3 is used, (14a) was penalized because the pivot term is a modifier to another proper noun. This gave another question (14b) relatively higher priority, resulting in the answer "Chalukya", which is actually the ruling dynasty when Angkor Wat was built. The resulting mismatch with the anticipated answer successfully supports the conclusion that (14) is false.

*(14) Angkor Wat was built during the same period as the Vardhana dynasty ruled in India.

(14a) Angkor Wat was built during the same period as the Vardhana dynasty ruled in this country. ?[India]

(14b) Angkor Wat was built during the same period as this dynasty ruled in India. *[Vardhana dynasty]

5.3 Error analysis

From unsuccessful cases, we found some difficulties. Here is a typical example of DeepQA's limitation. For the correct statement (15), three questions (15a) to (15c) were generated, but DeepQA's results were $A(q) \neq 1$ for all of them. The difficulty comes from ambiguity in "February Revolution", so "Russia" is answered for (15a) due to more popular revolution, and (15c) is difficult to answer.

(15) French provisional government formed after the February Revolution created National Workshops.

(15a) Provisional government of this country formed after the February Revolution created National Workshops. [France]

(15b) French provisional government formed after this revolution created National Workshops. [February Revolution]

(15c) French provisional government formed after the February Revolution created this place. [National Workshops]

Another interesting case was (16). "Agricultural Adjustment Act" and "United States" were answered for (16a) and (16b), respectively, but this statement was wrong because the prices were *raised*, not *lowered*. This is the limitation of DeepQA that answers the most likely proper nouns assuming there is an answer. To overcome this problem, other questions with opposite meaning (*e.g.* The prices of agricultural products were *raised*...) could be generated and submitted to DeepQA. If DeepQA performs well enough, it might return higher confidence values with questions derived from the correct statement.

*(16) The prices of agricultural products were lowered by the Agricultural Adjustment Act in United States.

(16a) The prices of agricultural products were lowered by this law in United States. ?[Agricultural Adjustment Act]

(16b) The prices of agricultural products were lowered by the Agricultural Adjustment Act in this country. ?[United States]

6 Discussion

6.1 Beyond yes/no answer

This system does not just return yes/no answers. For false propositions, the answer by DeepQA can suggest what element makes the proposition false, as we found in the example (7a) and (14b). We call this task *false trigger identification*.

Table 6 shows the frequency of such triggers found in the answer lists. Among the 68 false propositions in the development data, 11 false triggers were found as the first answer of

Question rank	Answer rank	Frequency
1st	1st	11
2nd to 3rd	1st	6
1st	2nd to 5th	2
2nd to 3rd	2nd to 5th	5

Table 6: The frequencies of false triggers found in the answer lists.

q^*, 6 false triggers were found as the first answer to other questions, and 7 were in the lower-ranked answers to one of the questions. For these incorrect statements, our system thus provides useful hints to make corrections to the input.

In 12 out of 46 unsuccessful cases the time period was incorrect as in Question (17) when the 15th century is the correct date. In these cases it is difficult to identify the false trigger because DeepQA was not particularly good at answering questions seeking the proper century.

*(17) In the 11th century, the ruler of the Malacca Sultanate converted to Islam.

6.2 Comparison with entailment techniques

Recently recognition of textual entailment (RTE) (Dagan et al., 2005) is intensively studied. The same world history question dataset was used in the evaluation of entailment task (Miyao et al., 2012). In their experiments, for a statement H in the choices for a question, a relevant sentence T was manually selected among Wikipedia articles so that H entails T only if statement H is true. They reported that the accuracy of 4-way multiple choice question in a world history examination using the best system was 54% (13/24). Since the prerequisites were different and the languages were also different (they used the original Japanese questions while we used English translations) we cannot directly compare the results. However, the accuracy using entailment is lower than that by our closed test in Table 5, and our method does not require the careful selection of a relevant single text from the corpus, so our approach can be argued that it is closer to a real application scenario.

7 Conclusion

This paper proposed a method to determine the correctness of propositions by leveraging the power of information sources accessed through a factoid-style question answering system. This was done by generating questions to be input to a subsystem and postprocessing the answer list produced by the system. This achieved quite encouraging results in both true and false determination, and hence in the overall examination goal of answering multiple choice questions. In this study, although the first stage of the process was manual, we identified kind of questions that are suitable or problematic for fact validation, and we found that reliable questions can be automatically selected by relying on the confidence values output by the factoid-style question-answering system. Besides yes/no determination, the system indicated an ability to identify why an input statement might be incorrect, suggesting our method can be applied in a dialog system that continuously validates the input utterances, or for semantic document content correction based on general knowledge, amongst others.

References

Bian, J., Liu, Y., Agichtein, E., and Zha, H. (2008). Finding the right facts in the crowd: factoid question answering over social media. In *Proceedings of the 17th international conference on World Wide Web*, pages 467–476.

Cui, H., Kan, M.-Y., and Chua, T.-S. (2005). Generic soft pattern models for definitional question answering. In *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 384–391.

Dagan, I., Glickman, O., and Magnini, B. (2005). The PASCAL recognizing textual entailment challenge. In *MLCW*, pages 177–190.

Ferrucci, D. A. (2012). Introduction to "This is Watson". *IBM Journal of Research and Development*, 56(3.4):1:1–1:15.

Green, N. and Carberry, S. (1994). Generating indirect answers to yes-no questions. In *Proceedings of the seventh International Workshop on Natural Language Generation*, pages 189–198.

Hirschberg, J. (1984). Toward a redefinition of yes/no questions. In *Proceedings of 10th COLING and 22nd ACL*, pages 48–51.

Kazama, J. and Torisawa, K. (2007). Exploiting Wikipedia as external knowledge for named entity recognition. In *Proceedings of EMNLP-CoNLL 2007*, pages 698–707.

Mann, W. C. and Thompson, S. A. (1987). Rhetorical structure theory: Description and construction of text structures. In Kempen, G., editor, *Natural Language Generation: New Results in Artificial Intelligence, Psychology, and Linguistics*, pages 85–95. Nijhoff.

Miyao, Y., Shima, H., Kanayama, H., and Mitamura, T. (2012). Evaluating textual entailment recognition for university entrance examinations. *ACM Transactions on Asian Language Information Processing*, 11(4). (to appear).

Prager, J., Radev, D., and Czuba, K. (2001). Answering what-is questions by virtual annotation. In *Proceedings of Human Language Technologies Conference*, pages 1–5.

Prager, J. M., Duboué, P. A., and Chu-Carroll, J. (2006). Improving QA accuracy by question inversion. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*.

Ravichandran, D. and Hovy, E. (2002). Learning surface text patterns for a question answering system. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, pages 41–47.

Shima, H., Kanayama, H., Lee, C., Lin, C., Mitamura, T., Miyao, Y., Shi, S., and Takeda, K. (2011). Overview of NTCIR-9 RITE: Recognizing Inference in TExt. In *Proceedings of NTCIR-9 Workshop Meeting*.

Voorhees, E. M. (2002). Overview of the TREC 2002 question answering track. In *Proceedings of the 11th Text REtrieval Conference (TREC)*, pages 115–123.

Xu, J., Licuanan, A., and Weischedel, R. M. (2003). Trec 2003 QA at BBN: Answering definitional questions. In *Proceedings of the 12th TREC*, pages 98–106.

Improving Topic Classification for Highly Inflective Languages

Jurgita KAPOČIŪTĖ-DZIKIENĖ[1] *Frederik VAASSEN*[2]
Walter DAELEMANS[2] *Algis KRUPAVIČIUS*[1]

(1) KAUNAS UNIVERSITY OF TECHNOLOGY, K. Donelaičio 73, LT-44249 Kaunas, Lithuania
(2) COMPUTATIONAL LINGUISTICS & PSYCHOLINGUISTICS RESEARCH CENTER, Lange Winkelstraat 40-42, B-2000 Antwerp, Belgium
Jurgita.Kapociute-Dzikiene@ktu.lt, Frederik.Vaassen@ua.ac.be, Walter.Daelemans@ua.ac.be, pvai@ktu.lt

ABSTRACT

Despite the existence of many effective methods to solve topic classification tasks for such widely used languages as English, there is no clear answer whether these methods are suitable for languages that are substantially different. We attempt to solve a topic classification task for Lithuanian, a relatively resource-scarce language that is highly inflective, has a rich vocabulary, and a complex word derivation system. We show that classification performance is significantly higher when the inflective character of the language is taken into account by using character n-grams as opposed to the more common bag-of-words approach. These results are not only promising for Lithuanian, but also for other languages with similar properties. We show that the performance of classifiers based on character n-grams even surpasses that of classifiers built on stemmed or lemmatized text. This indicates that topic classification is possible even for languages for which automatic grammatical tools are not available.

TITLE AND ABSTRACT IN LITHUANIAN

Klasifikavimo į temas gerinimas stipriai kaitomoms kalboms

Nepaisant to, jog tokioms plačiai naudojamoms kalboms kaip anglų yra sukurta daug efektyvių metodų, sprendžiančių klasifikavimo į temas uždavinius, neaišku ar šie metodai yra tinkami visiškai skirtingoms kalboms. Siekiame išspręsti klasifikavimo į temas uždavinį gana mažai išteklių šioje srityje turinčiai lietuvių kalbai, kuri yra stipriai kaitoma, turi turtingą žodyną, sudėtingą žodžių darybos sistemą. Pademonstruosime, kad galima pasiekti ženkliai geresnius klasifikavimo rezultatus, kuomet atsižvelgiama į kaitomą kalbos pobūdį: naudojamos simbolių n-gmamos vietoj labiau įprasto žodžių rinkinio. Gauti rezultatai perspektyvūs ne tik lietuvių kalbai, bet taip pat ir kitoms, panašiomis savybėmis pasižyminčioms, kalboms. Pademonstruosime, kad klasifikatorių, naudojančių simbolių n-gramas veikimas netgi efektyvesnis, palyginus su klasifikatoriais, naudojančiais į žodžių kamienus arba lemas transformuotą tekstą. O tai reiškia, kad šį klasifikavimo į temas metodą galima taikyti netgi toms kalboms, kurios neturi specializuotų automatinių gramatinių įrankių.

KEYWORDS : Character n-grams, topic classification, Lithuanian.
KEYWORDS IN LITHUANIAN : Simbolių n-gramos, klasifikavimas į temas, lietuvių kalba.

Proceedings of COLING 2012: Technical Papers, pages 1393–1410,
COLING 2012, Mumbai, December 2012.

1 Introduction

With the exponential growth in the number of electronic documents, automatic text categorization has become one of the key techniques to help control and organize the constant influx of information. The ever-increasing amount of articles, e-mails, comments, and other types of texts leaves no opportunity for manual classification.

Initially, the main focus of text categorization research was on proposing and applying supervised text classification methods on different benchmark data collections, such as Reuters-21578 (Lewis, 1992; Lewis et al., 2004), 20 Newsgroups (Lang, 1995), Ohsumed (Hersh et al., 1994; Lewis et al., 2004), etc. When text classification methods achieved sufficient accuracy, research topics started to diversify, focusing instead on integrating different sources of information (semantic information, world knowledge), developing increasingly autonomous classification methods (semi-supervised or unsupervised classification), applying the approach to diverse classification tasks (genre classification, author classification), etc.

Whereas currently proposed methods solve text classification tasks very effectively, they often require information that is not immediately present in the text. Using part-of-speech information, for instance, requires the availability of accurate part-of-speech taggers. The same can be said about world knowledge, which requires the existence of ontologies or common sense databases. For English and other major languages, a wide range of annotated corpora, grammatical tools and databases are available, but this is not the case for other, more resource-scarce languages. Furthermore, most research currently focuses on a limited selection of languages, and there is no guarantee that the techniques that work best to classify English texts will also be most effective for languages that are substantially different.

In this paper, we investigate whether the widely accepted bag-of-words approach that seems to work well for topic classification in English is also the best method for topic classification in Lithuanian, which is a highly inflectional language. We also assess whether it is possible to perform topic classification on Lithuanian without resorting to external resources such as extra grammatical annotation layers or automatic grammatical parsers, since Lithuanian has a limited number of these resources. We posit that our findings can also be applied to other, similar languages (according to such properties as inflectional morphology, complexity of word derivation, etc.) such as Latvian (the only other living Baltic language) or Slavic languages.

Section 2 contains an overview of related work. In Section 3, we describe the Lithuanian language and its properties. We outline the methodology of our topic classification experiments in Section 4 and present the results in Section 5. We discuss these results in Section 6 and evaluate their implications for text classification procedures in general, and for inflectionally rich languages specifically.

2 Related work

In this paper we focus on supervised machine learning methods (for review see Kotsiantis, 2007) applied to a text categorization task (for review see Sebastiani, 2002).

The most important features when performing topic classifications are usually those lexical features that clearly refer to topic-specific terminology. Function words are less important for this task, and so is most grammatical information. Therefore, prior to topic classification, the

documents are usually pre-processed in such a way that this topic-specific lexical information is easily accessible. This may involve the removal of stop words, function words; stemming or lemmatization; spelling normalization; word segmentation; etc. Stemming and lemmatization are advisable for highly inflective languages, spelling normalization for languages having a highly variable orthography (e.g. Arabic), word segmentation is demanded for languages having concatenative compounding (e.g. Swedish or German) and inevitable for the languages that have no whitespace between words (e.g. some Asian languages such as Chinese or Japanese). However, the gain one can expect from pre-processing steps like stemming or lemmatization seems to be strongly dependent on the language of the dataset. Some comparative experiments have revealed that stemming improves the classification performance of various machine learning algorithms on English texts (Radovanovic & Ivanovic, 2006). However, in Gaustad & Bouma (2002)'s experiments on Dutch data, it had no influence on classification results. Stemming can even adversely affect accuracy, as shown by Wahbeh et al. (2011) on Arabic data. As for lemmatization, Leopold & Kindermann (2002) report that it led to no significant classification improvements on German, sometimes even yielding worse results.

Regardless of these pre-processing steps, the most popular feature representation for topic classification seems to be the bag-of-words approach. In some cases, it may be interesting to add word bigrams, on the condition that these are not entirely redundant. Indeed, adding bigrams does not usually help classification performance (on the contrary, performance often drops), since the feature space becomes even more sparse. Boulis & Ostendorf (2005), however, showed that carefully selected token bigrams can offer improvements over the bag-of-words approach. They propose a measure to evaluate the redundancy of token bigrams based only on their related words. Tan et al. (2002) use a lexical analyzer to determine and include the most important token bigrams and demonstrate a positive influence on classification results. Nastase et al. (2007) use syntactically related word pairs (verb + each of its arguments, noun + each of its modifiers) instead of token n-grams and report an improvement over the pure bag-of-words representation.

Stop word removal, stemming, lemmatization and lexical analysis are language dependent, and are therefore necessarily based on external resources –lists of stop words, grammar tools or dictionaries– resources that resource-scarce languages may not always have. It is however also possible to use character level n-grams. Character n-grams can often capture the language-dependent elements without having to resort to external stemmers, lemmatizers or word segmentation tools. Character n-grams will highlight highly relevant word segments, and since each string is decomposed into small parts, any error can affect only a limited number of these parts, which makes character n-grams ideal to use with informal texts containing many typographical errors.

Peng et al. (2003a, 2003b) skipped language dependent pre-processing and feature selection stages and demonstrated that a language modelling approach on character n-grams gives superior classification results for English and competitive results for Chinese and Japanese. For English, the most accurate performance was achieved with n-grams of 3 characters or more, peaking at 6-grams with Witten-Bells smoothing. Bigrams (or higher order) with different smoothing techniques worked best for Chinese, whereas 6-grams (or more) with absolute or Written-Bell smoothing worked best for Japanese. Wei et al. (2008)'s experimented on Chinese texts, showing good performance for combinations of character unigrams and bigrams, or combinations of unigram, bigrams and trigrams. Peng & Schuurmans (2003) demonstrated that a chain-

augmented Naïve Bayes classifier can work equally well on character n-grams as on bag-of-words for Chinese, Japanese, English and Greek.

Sometimes character level n-grams are applied on pre-processed texts. Khreisat (2006) applied character trigrams on Arabic texts after stop word removal and spelling normalization. A similar approach was applied to Farsi by Bina et al. (2008) who report that 4-grams give the best results.

Unfortunately, we cannot give any examples of topic classification experiments for Lithuanian, since no automatic text categorization tasks have yet been described for this language. Consequently, this paper will be the first attempt at finding a good method to perform topic classification on Lithuanian text, taking into account language-specific characteristics, while keeping the use of external information sources to a minimum.

3 The Lithuanian language

Lithuanian is considered one of the most archaic and conservative living Indo-European languages (Gimbutas, 1963) and it has retained a lot of cognates to many words found in such languages as Sanskrit or Latin.

Lithuanian has a rich vocabulary and word derivation system. The Academic Dictionary of Lithuanian (Naktinienė et al., 2005) has more than 0.6 million headwords (e.g. Oxford English Dictionary has about 0.3 million headwords, Deutsches Wörterbuch – the largest lexicon of the German language – has around 0.33 million). As an example of Lithuanian's rich vocabulary, in the Academic Dictionary of Lithuanian, there are more than 1300 synonyms for the Lithuanian word "eiti" (*to go*) (Piročkinas, 2012) – the majority of which are derived from onomatopeias – and for some of these synonyms it is impossible to find equivalents in other languages. Lithuanian has 25 prefixes for nouns and 78 suffixes for diminutives and hypocoristic words (Ulvydas, 1965). Diminutives and hypocoristic words (which are especially frequent in fiction and spoken language) very often have two/three suffixes, and in rare cases the number of suffixes can go as high as six. E.g. the word "sesuo" (*sister*) can be found in texts as "sesutė", "sesužėlė", "seserytė", etc.

Of all living Indo-European languages, Lithuanian has the richest inflectional morphology, making it more complex than even Latvian or Slavic languages (Savickienė et al., 2009). Different parts-of-speech of a word are expressed through inflections and through different endings. As an example of Lithuanian's inflectional complexity, one can calculate all the possible word forms for the Lithuanian word "aukštas" and its English translation, "*high*" (see FIGURE 1).

In Lithuanian, nominal words (nouns, adjectives, numerals, pronouns, and participles) are inflected by 7 (+2) cases, 2 (+1) genders, and 2 numbers. There are 7 "common" cases, and 2 additional forms of locative that are still found in spoken language and idiomatic use. Nouns have 5 declensions (12 inflection paradigms), adjectives have 3 (9 inflection paradigms). E.g. the noun "vanduo" (*water*) of a masculine gender can be found in Lithuanian texts written as "vanduo", "vandens", "vandeniui", "vandenį", "vandeniu", "vandenimi", "vandenyje", "vandenie", "vandenys", "vandenų", "vandenims", "vandenis", "vandenimis", "vandenyse". The word "vanduo" is an example of one of the inflection paradigms, where the part "vand" (i.e. the root of word) always remains stable. Nominal words usually have 2 grammatical genders, except for adjectives, which have an additional neuter gender that is not declined.

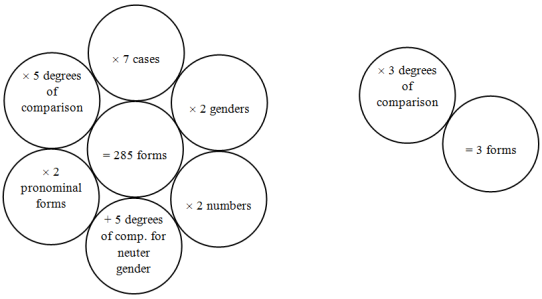


FIGURE 1 – The adjective "aukštas" (*high*) has 285 different forms in Lithuanian (a); and 3 forms in English (b).

Lithuanian also has pronominal forms of adjectives, ordinal numerals, pronouns and active forms of participles. Pronominal forms are unknown to English, German, French, and to hundreds of other languages (they are however present in Russian, for instance). They are used to highlight an object and its properties from other similar objects. Pronominal forms are composed by appending the anaphoric pronoun to the ending of the word and composing more complex dissyllabic endings. E.g., the ordinal number "pirmas" (*the first*) has a simple ending indicating masculine gender in singular number, namely "as". The pronominal form, "pirmasis", is formed by adding an extra ending, "is". Both endings can be inflected further, which can even lead to trisyllabic endings, as is the case of the plural locative "pirmuosiuose", which has an ending made up of three elements: "uos", "iuos" and "e". Despite these extra endings, the root "pirm" is retained.

Adjectives, adverbs, and some forms of verbs have 3 (+2) degrees of comparison, where two of them are usually used only in fiction and spoken language and have no equivalents in English. Degrees of comparison in Lithuanian are expressed by different endings and suffixes, e.g. singular masculine gender noun "gražus" (*beautiful*), "gražesnis" (*more beautiful*), "gražiausias" (*the most beautiful*), where "graž" remains stable.

The most ancient Indo-European languages had 2 or 3 degrees of comparison for demonstratives (pronouns), but in most existing languages only 2 (sometimes 1) are left. Meanwhile, Lithuanian retained 3 degrees of comparison: describing objects located close to the speaker; objects located not close to the speaker, but close to the listener; objects located far from both, i.e. the speaker and the listener. All these words are also inflected, but are too short to retain stable parts longer than one symbol. E.g. the pronoun "šis" (*this/these*) can be found as "šio", "šiam", "šį", "šiuo", "šiame", "šie", "šių", "šiems", "šiuos", "šiais", "šiuose".

Lithuanian language also has compound nouns, consisting of two or even three roots. E.g. "saulėtekis" (*sunrise*) ("saulė", *sun* + "tekėti", *to rise*), "sienlaikraštis" (*wall newspaper*) ("siena", *wall* + "laikas", *time* + "raštas", *writing*), etc. But only the last of the compound words can change its inflection form. E.g. "sienlaikraštis", "sienlaikraščio", "sienlaikraščiui", etc. The same can be said of some fixed combinations of words, e.g. "žemės ūkis" (*agriculture*), "saulės smūgis" (*sunstroke*), where the first word is stable and the second is inflected ("žemės ūkio",

"žemės ūkiui", "žemės ūkį", etc.), whereas both words can be inflected when they occur separately.

Verbs have 3 conjugations, and are inflected by 4 tenses, 3 persons, 2 numbers, and 3 moods. Some conjugated verb forms are also reflexive. This can be expressed by adding particle "si" or "s" at the end of word or after extra added prefixes, e.g. "nusiprausti" (particle "si" after prefix "nu"), "praustis" (*to have a wash*) (particle "s" at the end of word). Both words have the same root "praus". Phrasal verbs in Lithuanian have extra prefixes (14 in total), whereas in English they are expressed by two words: i.e. verb + preposition/adverb. E.g. "nubėgti" (*to run away*), "prabėgti" (*to run by*), "įbėgti" (*to run in*), "išbėgti" (*to run out*), all having different prefixes but the same stable root "bėg". Other forms of verbs are non-conjugative (e.g. participles, adverbial participles, verbal adverbs, and some forms of gerund), but can be inflected by tense, case, gender, number, and have active and passive forms. Non-conjugative forms of verbs retain the same root, but have different suffixes and endings in different inflection forms.

4 Methodology

Our goal is to see if we can carry out automatic topic classification despite the complexity of Lithuanian (and, by extension, of other Baltic and Slavic languages). Since for these relatively resource-scarce languages, we do not always have access to freely available dictionaries, annotated datasets or grammatical tools, we also want to avoid using such external resources and use only the information present in the text's surface representation.

Section 4.1 will describe the datasets we used and Section 4.2 will give a formal description of the classification task. In Section 4.3, we will describe which different feature types we experimented with and formulate two hypotheses we would like to test through these experiments. Section 4.4 describes the actual classification process, including optimization of the classifier and evaluation of the models' performance.

4.1 Datasets

All of our experiments were carried out on three different datasets to make sure our findings generalize over different domains.

- "*Lietuvos rytas*". The information for this dataset was crawled from the internet forum of the largest daily newspaper in Lithuania, "Lietuvos rytas" (Lietuvos rytas, 2012). The structure of the forum is hierarchical: it contains a number of topics (root categories), each of which may contain several sub-topics (sub-categories). Within a topic, the discussion is determined by a collection of posted messages. The data was crawled automatically, resulting in a collection of documents, each containing a single forum post, and grouped according to their root category. Root categories (topics) are very general including such areas as "Business", "Politics", "Sports", etc. The "Lietuvos rytas" dataset is thus composed of 8,936 text documents grouped into 11 topics.
- "*Supermamos*". This dataset contains information from the largest internet forum for women in Lithuania (Supermamos, 2012). The structure of this forum is analogous to the "Lietuvos rytas" forum, and was processed similarly. The root categories are more specific compared to "Lietuvos rytas", and concern topics such as "Parenting and Education", "Maternity and Paternity", "Health and Beauty", etc. The created dataset contains 11,353 text documents grouped into 14 topics.

- "*Rinkimų programos*'04". This dataset contains political information, extracted from the programs of major political parties for the Lithuanian parliamentary elections of 2004. The programs were manually processed by a domain expert following classification rules determined in Volkens (2002): each program was split into small phrases and labeled with appropriate categories indicating policy domains, such as "External Relations", "Freedom and Democracy", "Political system", "Economy", etc. The "Rinkimų programos'04" dataset contains 2,388 text documents grouped into 8 topics (7 are related to policy domains and one is neutral).

As we already mentioned, the "Lietuvos rytas" and "Supermamos" datasets contain messages from internet forums, and thus represent spoken Lithuanian language in written form. "Rinkimų programos'04" contains texts extracted from formal documents and represents normative written Lithuanian.

The classification of "Lietuvos rytas" and "Supermamos" is a challenging task, even for a domain expert, because some posts are very general, and attaching them to any of the possible topics is very difficult. Moreover, internet forums have their own specific jargon full of informal words, foreign language insertions, barbarisms, and other expressions that are outside standard language norms. Besides, "Lietuvos rytas" and especially "Supermamos" are full of diminutives and hypocoristic words. Typographical and grammatical errors can also not be avoided. A particularly difficult problem is that informal written language often imitates spoken language. In the case of Lithuanian, some grammatical forms are very often pronounced without the ending vowel (such as "mergyt" → "mergyte" (*lassie*, in vocative case); "eit" → "eiti" (*to go*, infinitive); "schemoj" → "schemoje", (*schema*, in locative case)). This trend is reflected in informal written language. Moreover, Lithuanian uses the Latin script supplemented with diacritics (ą, č, ę, ė, į, š, ų, ū, ž), but in internet messages, diacritics are very often replaced with matching Latin letters (e.g.: ą → a, č → c, ę → e, etc.) or pairs of letters expressing the same sounds as in English (e.g.: č [tʃ] → ch, š [ʃ]→ sh, etc.). "Rinkimų programos'04", consisting of formal written language only, avoids the above-mentioned problems.

TABLE 1 shows the number of topics (classes) and the majority and random baselines (determined by calculating the sum of the squared probability of the classes, or $\sum_i P(c_i)^2$)) for each dataset.

Dataset	Number of topics	Number of text documents	Random baseline	Majority baseline
Lietuvos rytas	11	8,936	0.145	0.247
Supermamos	14	11,353	0.093	0.137
Rinkimų programos'04	8	2,388	0.167	0.231

TABLE 1 – Number of topics and text documents, random and majority baselines for all datasets.

4.2 Formal description of the task

The mathematical formulation of the topic classification task we are attempting to solve is given below.

Let $d \in D$ be a text document, belonging to a document space D. In our case we have three document spaces, related to different datasets: "Lietuvos rytas", "Supermamos", and "Rinkimų programos'04".

Let C be a finite set of classes (topics): $C=\{c_1, c_2, \ldots, c_N\}$. In our case $2 < N << \infty$ – i.e. we have multi-class classification problem, because "Lietuvos rytas" has N=11 classes; "Supermamos" – N=14 classes; "Rinkimų programos'04" – N=8 classes.

Let D^L be a training set, containing instances I – i.e. document feature vectors d' (where d' corresponds to document d) with their appropriate class labels: $I=\langle d', c\rangle$. "Lietuvos rytas", "Supermamos", "Rinkimų programos'04" contain single-labeled instances only: i.e. the text document d cannot be attached to more than one class c.

Let function γ be a classification function mapping text documents to classes, $\gamma : D \rightarrow C$. Function γ determines how d was labelled with c. In "Lietuvos rytas" and "Supermamos", the labeling was performed automatically by the users by replying to forum messages; in "Rinkimų programos'04", the annotation was performed by a domain expert.

Let Γ denote a supervised learning method which given training D^L as the input, can return a learned classification function γ' (defined as a model) as the output: $\Gamma(D^L)\rightarrow\gamma'$.

4.3 Experimental setup

The datasets "Lietuvos rytas", "Supermamos" and "Rinkimų programos'04" were pre-processed using different pre-processing techniques (TABLE 2):

Dataset	Total number of tokens	Number of distinct tokens	Number of distinct tokens after stemming	Number of distinct tokens after lemmatization
Lietuvos rytas	331,068	70,760	41,291	37,743
Supermamos	703,220	116,144	67,811	63,919
Rinkimų programos'04	29,745	7,426	4,474	3,320

TABLE 2 – Statistics about pre-processed datasets.

- Stemming. This pre-processing technique includes case normalization, the replacement of numbers with a general placeholder, and stemming. Words were stemmed using a Lithuanian stemmer (Krilavičius & Medelis, 2010) based on the Porter stemming algorithm (Porter, 1980; Willett, 2006). The Lithuanian stemmer eliminates endings and some suffixes. It is rule-based and can therefore cope with some irregular words (barbarisms, words with replaced diacritic letters, and words with grammatical errors). In some cases, stemming can cause the loss of meaning (e.g. "sala" (*island*), "salė" (*hall*), and "salti" (*to malt*) will be stemmed to the same "sal"). This pre-processing technique allows reducing the number of distinct tokens by ~42% for "Lietuvos rytas" and "Supermamos", and by ~40% for "Rinkimų programos'04".
- Lemmatization. This preprocessing technique includes the replacement of numbers by a placeholder. Documents were lemmatized using the Lithuanian morphological analyzer-lemmatizer "Lemuoklis" (Zinkevičius, 2000; Daudaravičius et al., 2007), which also solves morphological disambiguation problems. Its accuracy on normative Lithuanian texts is ~94% (Rimkutė & Daudaravičius, 2007). "Lemuoklis" transforms words into their lemmas by replacing the words' endings by their main lemma ending (but it does not touch suffixes and prefixes). However, the lemmatizer is dictionary-based, and can therefore not cope with unknown words. Case normalization was not necessary, because

"Lemuoklis" transforms the letters of generic words into lower case and leaves proper nouns capitalized. Unrecognized words are not modified in any way. "Lemuoklis" was unable to recognize ~15% of tokens in "Lietuvos rytas"; ~26% in "Supermamos", and ~2.5% in "Rinkimų programos'04". These numbers are in line with research by Giesbrecht & Evert (2009), who showed that for German data, every seventh token is incorrectly recognized when working with internet forum data, whereas almost no errors occur in (standard-language) fiction texts. Lemmatization allowed reducing the number of distinct tokens by ~47% for "Lietuvos rytas", by ~45% for "Supermamos", and by ~55% for "Rinkimų programos'04".

It should be noted that even though it is a common pre-processing step, we chose not to remove stop words from our datasets. This choice was made because of the following two reasons: 1) there is no list of stop words available for Lithuanian; 2) stop words are not a big problem in Lithuanian given the inflective nature of the language (e.g. Lithuanian does not have articles; phrasal verbs are expressed with a single token; negations are usually expressed by single token also, where participle "ne" (*not*, *no*) is attached as the prefix, etc.).

We now have three versions of the "Lietuvos rytas", "Supermamos" and "Rinkimų programos'04" datasets: one version with unmodified words, one with stemmed words, and one in which the words are lemmatized. Both the lemmatizer and the stemmer significantly reduce the number of distinct features, mapping several forms of the same word to a single feature. This should improve topic classification accuracy, but both the lemmatizer and the stemmer are in fact external resources, which may not be available for many resource-scarce languages, or which may not be accurate enough. Therefore, we will not only perform experiments with unigrams of lemmatized or stemmed words, but we will also attempt to classify the documents by using character n-grams. It is our expectation that the character n-grams will capture intrinsically what the lemmatizer and stemmer do explicitly, i.e. the terms present in the text, regardless of their inflection or derivational affixes.

The different feature types we will compare for all three datasets are thus the following:

- Unigrams based on word tokens.
- Unigrams based on stemmed words.
- Unigrams based on lemmatized words.
- Character n-grams based on word tokens.

The results of the corresponding experiments are described in Section 5.

Our first hypothesis is that a simple bag-of-words approach, which has shown its efficiency in topic classification tasks in English and in related languages, will not be the best technique for Lithuanian. We expect stemming and lemmatization to improve the classification results for Lithuanian significantly, due to the language's complex morphology and the importance of inflection.

Our second hypothesis is that it may not be necessary to use grammatical tools such as stemmers or lemmatizers – which may not be available for many resource-scarce languages – to capture the influence of this complex morphology and inflectional system. Character n-grams can capture these influences intrinsically. We expect character n-grams to perform significantly better than the bag-of-words approach, and we anticipate they will reach similar performance as the stemmed or lemmatized words.

4.4 Classification

We attempt to find a method Γ which could create the model γ' that best approximates γ. Since topic classification of text has never been solved for Lithuanian, we selected Support Vector Machines (a detailed description of SVMs is presented in Cortes & Vapnik (1995)) as Γ, based on comparative research indicating that SVMs perform best on a variety of text classification experiments. Additionally, SVMs can cope with the following problems which typically occur in text classification tasks:

- High-dimensional feature spaces. A very large feature set is not uncommon when dealing with text documents. E.g. if all token unigrams in a dataset (when $D^L = D$) would be considered as features (following the very common bag-of-words approach), "Lietuvos rytas" would have 70,760 features, "Supermamos" – 116,144; "Rinkimų programos'04" – 7,426.
- Sparseness of feature vectors. Each *d'* usually contains only a small number of features with non-zero values; these vectors are therefore considered sparse. E.g. in a token unigram approach (assuming $D^L = D$), each *d'* would contain ~14 non-zero feature values (among 70,760) in "Lietuvos rytas"; ~18 (among 116,144 features) in "Supermamos"; ~5 (among 7,426 features) in "Rinkimų programos'04".
- Heterogeneous use of features: i.e. when instances belonging to the same class do not share any common features. This is often the case when classifying short texts. Each *I* contains ~36 words in "Lietuvos rytas"; ~60 in "Supermamos"; only ~12 in "Rinkimų programos'04".

Additionally, SVMs do not require one to perform aggressive feature selection, which may result in a loss of information. This is especially important in our case, i.e. when working on Lithuanian, where there may be important information in both its rich vocabulary and in its complex word derivation system. The implementation of Support Vector Machines we use in our experiments is LIBSVM (Chang & Lin, 2011), using the radial basis kernel function. Most parameters were left to their default values, except for the cost parameter and the gamma parameter in the radial basis function, which were optimized using a grid search.

5 Results

All results reported in TABLE 3 – TABLE 5 are based on 10-fold cross-validation. For each experiment, the learner parameters were determined using a grid search, using macro-averaged F-scores as the optimization target.

	Lietuvos rytas		
	acc.	macro-F	micro-F
Tokens-1	0.271	0.190	0.265
Stems-1	0.305	0.221	0.300
Lemmas-1	0.303	0.232	0.300
Characters-4	0.324	0.237	0.317
Random baseline	*0.091*		
Majority baseline	*0.247*		

TABLE 3 – Accuracies, macro-averaged and micro-averaged F-scores for the different feature types on "Lietuvos rytas".

	Supermamos		
	acc.	macro-F	micro-F
Tokens-1	0.327	0.277	0.316
Stems-1	0.360	0.320	0.353
Lemmas-1	0.365	0.323	0.358
Characters-4	0.398	0.356	0.392
Random baseline	*0.071*		
Majority baseline	*0.137*		

TABLE 4 – Classification results for "Supermamos".

	Rinkimų programos'04		
	acc.	macro-F	micro-F
Tokens-1	0.298	0.238	0.282
Stems-1	0.326	0.262	0.306
Lemmas-1	0.363	0.291	0.346
Characters-4	0.376	0.290	0.351
Random baseline	*0.126*		
Majority baseline	*0.231*		

TABLE 5 – Classification results for "Rinkimų programos'04".

To determine whether the performances of the classifiers trained with each feature type were significantly different from each other, we performed approximate randomization testing (Noreen, 1989; Yeh, 2000).

For "Lietuvos rytas", all differences are significant to a very high degree ($p <= 0.0001$), with the following exceptions: the difference between the "Stems-1" and the "Lemmas-1" results are not statistically significant; the difference between "Characters-4" and "Stems-1" is significant to a high degree (accuracy: $p = 0.001$, macro-F: $p = 0.006$, micro-F: $p = 0.003$); the difference between "Characters-4" and "Lemmas-1" is only significant in terms of accuracy ($p = 0.0005$) and micro-F ($p = 0.001$).

For "Supermamos", all differences are significant to a very high degree ($p <= 0.0001$), with the exception of the difference between "Stem-1" and "Lemmas-1", which is not significant.

Finally, for "Rinkimų programos'04", all differences are significant to a very high degree ($p <= 0.0001$), with the following exceptions: the difference between "Token-1" and "Stems-1" in terms of macro-F is slightly less significant ($p = 0.007$), as is the difference between "Characters-4" and "Stem-1" ($p = 0.002$); the differences between "Characters-4" and "Lemmas-1" are not statistically significant.

6 Discussion

Before we could compare the character n-gram performance to the performance of the word unigrams (tokens, stemmed words or lemmatized words), it was necessary to determine the optimal size of the n-grams. As reported by Peng et al. (2003a, 2003b) (see Section 2), for English, the character n-gram performance is competitive starting from trigrams, and peaks at 6-grams. Chinese, on the other hand, performed best with bigrams. For classification of Farsi texts,

character 4-grams seem to be the recommended feature type (Bina et al., 2008). The optimal size of the character n-grams thus appears strongly dependent on the language of the dataset. No research regarding the optimal n-gram size for Lithuanian has yet been published.

We expect word roots to give us the most useful information for a topic classification task, and n-grams of a length that manages to capture these roots most often should therefore be the most powerful. Based on the fact that Lithuanian has an average word length of 6 characters, and that most endings are 2 characters long, we expect that 4-grams will yield the best classification performance.

To test this hypothesis, we carried out experiments on all three datasets using all n-gram sizes ranging from trigrams to 7-grams. FIGURE 2 shows a graph mapping the n-gram performance (in terms of micro-averaged F-score) on all three datasets.

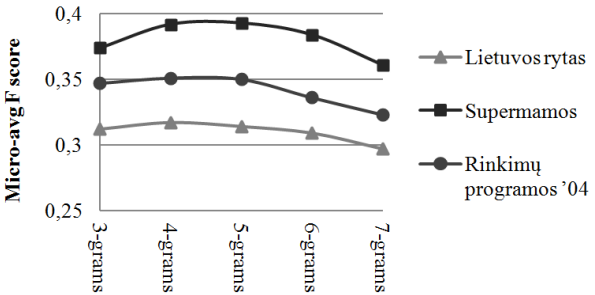


FIGURE 2 – Micro-averaged F-scores for n-gram sizes from 3-grams to 7-grams on all three datasets.

As expected, performance reaches a peak for character 4-grams, though the difference with 5-grams is subtle. Shorter string segments (trigrams) do not appear to capture enough information, while the longer segments (6-grams, 7-grams) fail to generalize sufficiently. TABLE 6 contains a more detailed performance breakdown.

	Lietuvos rytas	Supermamos	Rinkimų programos'04
3-grams	0.312	0.374	0.347
4-grams	0.317	0.392	0.351
5-grams	0.314	0.393	0.350
6-grams	0.309	0.384	0.336
7-grams	0.297	0.361	0.323

TABLE 6 – Micro-averaged F-scores for n-gram sizes from 3-grams to 7-grams on all three datasets.

A glance at TABLE 3 – TABLE 5 in Section 5 shows that our first hypothesis is confirmed: the simple bag-of-words approach (Tokens-1) is easily beaten when using a stemmer or a lemmatizer. For both the "Lietuvos rytas" and "Supermamos" datasets, stemming and lemmatization both account for an increase of 0.04 in micro-averaged F-score. On "Rinkimų programos'04", stemming does cause a small boost in performance, but lemmatization seems to

perform even better. The reason why lemmatization improves performance over stemming on the "Rinkimų programos'04" dataset, while it only matches the stemmer's scores on the other datasets, is that "Rinkimų programos'04" contains formal written text. As we described in Section 4.1, "Lietuvos rytas" and "Supermamos" both contain conversations from Internet forums, which are much more informal and contain non-standard language. The lemmatization package "Lemuoklis" is unable to deal with much of this "chat" data, while it does a better job lemmatizing the formal language in the political dataset.

More interestingly, the results also support our second hypothesis: the character 4-gram scores (which were the best n-gram scores, as shown in TABLE 3) are consistently higher than the scores for the other feature types. This means that not only is it possible to perform topic classification for Lithuanian using features that require no external resources, this approach even outperforms the standard classifiers trained on unigrams of stemmed or lemmatized words. The experiments were performed using different datasets, representing both formal and informal Lithuanian, with different particularities and different topic distributions. It can therefore be concluded that the hypothesis does not depend on the corpus or on a specific set of topics, but that it is applicable to Lithuanian in general. Assuming this pattern holds for other languages with similar properties (Latvian and some Slavic languages), this is good news indeed for resource-scarce languages where external tools such as lemmatizers or stemmers may not be available.

If we look more closely at why character n-grams work, we see that they capture patterns which are not always captured by alternative feature types. The lemmatizer reduces the word ending to their base form, but it does not remove suffixes ("mama", *mother*, and "mamytė", *mother* in diminutive, for instance, have the same root "mam", but are treated differently) and prefixes (e.g. "išbėgti", *to run out*, and "bėgti", *to run*, have the same root "bėg", but are treated differently). This means that phrasal verbs are not decomposed and diminutives or hypocoristic words remain untouched by the algorithm. The stemmer, then, is purely rule-based and captures even less of the intricacies of the Lithuanian language. It does not consider the function or meaning of a word in a sentence, and instead mechanically eliminates word endings (and some of the suffixes), resulting in an output that is all but perfect. The word "sveikas" (*healthy*), for instance, is stemmed to "sve" but should be stemmed to "sveik": i.e. ending "as" is correctly eliminated, but "ik" is erroneously treated as a suffix. With character n-grams, on the other hand, one does not need to explicitly define how words are formed. Character n-grams will implicitly capture the relevant inflectional patterns and the base word forms, which make them excellent features for this topic classification task.

The top character 4-gram features for the "Rinkimų programos'04" dataset, for instance, mostly contain the stable parts of words that are not influenced by inflection. In many cases, even, the n-grams capture the semantics of a term, which is stable over different parts of speech. The n-gram "vald", for instance, corresponds to the root of several, related words: the nouns "valdymas" (*management*), "valdžia" (*authority*) and "pavaldumas" (*subordination*); the verb "valdyti" (*to govern*); the phrasal verbs "įvaldyti" (*to master*), "suvaldyti" (*to manage*), etc. It is also one of the roots in the compound nouns "savivaldybė" (*municipality*) ("savas", *own* + "valdyti", *to govern*) and "žemėvalda" (*land-ownership*) ("žemė", *land* + "valdyti", *to govern*). Similarly, the character n-gram "kari" captures the nouns "karininkas" (*officer*) and "kariuomenė" (*army*); the adjective "karinis" (*military*). It captures the part of the noun "kariai" (*soldiers*) that remains stable in all inflected plural forms.

The examples above are lemmas (with the exception of "kariai", which is the plural of "karys", *soldier*), meaning they would not be transformed further by the lemmatizer. The stemmer would shorten the lemmas by eliminating endings and some suffixes, but prefixes and compound words would still remain.

By capturing those parts of words that are stable over semantically related words across different parts of speech, character n-grams help resolve feature sparseness, which is especially problematic for Lithuanian. Since Lithuanian has such a rich vocabulary, the standard bag-of-words approach results in an extremely sparse feature space, which can significantly complicate the construction of a good classification model. Stemming and lemmatization help reduce sparseness by limiting the number of possible realizations of the same lemma, but they do nothing to reduce the number of lemmas. Character n-grams, however, segment text into more frequent, more general substrings, which results in more reliable models.

Character n-grams also don't throw away valuable information as a stemmer or lemmatizer might. Some character n-grams, such as "emės", capture a string that remains stable in fixed combinations of two words, e.g. "žemės ūkis" (*agriculture*), "žemės grąžinimas" (*land restitution*), "žemės nuosavybė" (*land property*), etc. Both the stemmer and the lemmatizer would change "žemės" to "žemė", thus discarding valuable information.

Conclusion and perspectives

We have formulated and experimentally confirmed two hypotheses regarding topic classification for morphologically rich languages such as Lithuanian:

- The standard bag-of-words approach which works best for English and related languages, is not the best approach for these languages. Where English barely benefits from stemming and lemmatization, Lithuanian (and, by extension, likely also other Baltic and Slavic languages) benefit significantly from these pre-processing steps.
- It is possible to capture the intricate morphology of Baltic and Slavic languages without resorting to external tools such as stemmers or lemmatizers, which may not be available (or which may simply not work sufficiently well). Character n-grams implicitly capture relevant patterns and can outperform classifiers trained on stemmed or lemmatized data.

Our assumptions that these hypotheses also hold for Slavic languages are based on the characteristics that they share with Lithuanian. In future research, it will be interesting to experiment with datasets in these languages to see if these assumptions hold.

Acknowledgments

This research is funded by European Union Structural Funds project "Postdoctoral Fellowship Implementation in Lithuania" (No. VP1-3.1-ŠMM-01) and was initiated when the first author was visiting the Computational Linguistics & Psycholinguistics Research Center, Antwerp, Belgium. Frederik Vaassen is funded by the IWT (the Flemish government agency for Innovation by Science and Technology), TETRA project deLearyous.

References

Bina, B., Ahmadi, M. H., and Rahgozar, M. (2008). Farsi Text Classification Using N-Grams and Knn Algorithm, A Comparative Study. In *Proceedings of International Conference on Data Mining*, DMIN 2008, pages 385–390.

Boulis, C., and Ostendof, M. (2005). Text Classification by Augmenting the Bag-of-Words Representation with Redundancy-Compensated Bigrams. In *Proceedings of the SIAM International Conference on Data Mining at the Workshop on Feature Selection in Data Mining* (SIAM-FSDM 2005), Newport Beach, California.

Chang, C. and Lin, C. (2011). LIBSVM: A library for support vector machines. In *ACM Transactions on Intelligent Systems and Technology* (TIST 2011), 2(3), pages 1–27.

Cortes C. and Vapnik, V. (1995). Support-vector networks. *Machine Learning*, 20: 273–297.

Daudaravičius, V., Rimkutė, E. and Utka, A. (2007). Morphological annotation of the Lithuanian corpus. In *Proceedings of the Workshop on Balto-Slavonic Natural Language Processing: Information Extraction and Enabling Technologies* (ACL'07), Prague, Czech Republic, pages 94–99.

Gaustad, T. and Bouma, G. (2002). Accurate Stemming of Dutch for Text Classification. *Language and Computers*, pages 104–117.

Giesbrecht, E. and Evert, S. (2009). Part-of-Speech (POS) Tagging - a solved task? An evaluation of POS taggers for the Web as corpus. In *Proceedings of the Fifth Web as Corpus Workshop* (WAC5), pages 27–35.

Gimbutas M. (1963). *The Balts*. Thames and Hudson, London.

Hersh, W., Buckley, C., Leone, T. J. and Hickman, D. (1994). OHSUMED: an interactive retrieval evaluation and new large text collection for research. In *Proceedings of the Seventeenth Annual International ACM-SIGIR Conference on Research and Development in Information Retrieval* (SIGIR 94), pages 192–201.

Khreisat, K. (2006). Arabic Text Classification Using N-Gram Frequency Statistics. A Comparative Study. In *Proceedings of International Conference on Data Mining*, DMIN 2006, pages 78–82.

Kotsiantis, S. B. (2007). Supervised Machine Learning: A Review of Classification Techniques. *Informatica*, 31: 249–268.

Krilavičius, T. and Medelis, Ž. Lithuanian stemmer. (2010). May, 2012. <https://github.com/tokenmill/ltlangpack/tree/master/snowball/>.

Lang K. (1995). NewsWeeder: Learning to Filter Netnews. In *Proceedings of the 12th International Machine Learning Conference* (ML95), Tahoe, CA, USA, pages 331–339.

Leopold, E. and Kindermann, J. (2002). Text Categorization with Support Vector Machines. How to Represent Texts in Input Space? *Machine Learning*, 46 (1–3): 423–444.

Lewis, D. D. (1992). An evaluation of phrasal and clustered representations on a text categorization task. In *Proceedings of the Fifteenth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval* (SIGIR 92), pages 37–50.

Lewis, D. D., Yang, Y., Rose, T. G. and Li, F. (2004). RCV1: A New Benchmark Collection for Text Categorization Research. *Journal of Machine Learning Research archive*, 5: 361–397.

Lietuvos rytas. (2012). Internet forum of a newspaper "Lietuvos rytas". May, 2012. <http://forum.lrytas.lt/>. (In Lithuanian).

Naktinienė, G., Paulauskas, J., Petrokienė, R., Vitkauskas, V. and Zabarskaitė, J., editors. (2005). *Lietuvių kalbos žodynas (t. 1-20, 1941-2002): elektroninis variantas* [Lithuanian language dictionary (vol. 1-20, 1941-2002): electronic version]. Lietuvių kalbos institutas, Vilnius, Lithuania. < http://www.lkz.lt/dzl.php>. (in Lithuanian).

Nastase, V., Sayyad, J., and Caropreso, M. F. (2007). Using Dependency Relations for Text Classification. Technical Report TR-2007-12, University of Ottawa, Canada.

Noreen, E. W. (1989). *Computer-intensive Methods for Testing Hypotheses: An Introduction*. John Wiley & Sons, New York, NY, USA.

Peng, F., and Schuurmans, D. (2003). Combining Naive Bayes and n-Gram Language Models for Text Classification. In *Proceedings of 25th European Conference on Information Retrieval Research* (ECIR), pages 335–350.

Peng, F., Huang, X., Schuurmans, D., and Shaojun, W. (2003a). Text classification in Asian languages without word segmentation. In *Proceedings of the sixth international workshop on Information retrieval with Asian languages*, 11, pages 41–48.

Peng, F., Schuurmans, D., and Wang, S. (2003b). Language and task independent text categorization with simple language models. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology*, NAACL'03, 1, pages 110-117.

Piročkinas, A. (2012). Dar keli išskirtiniai lietuvių kalbos bruožai [Several distinctive features of Lithuanian language]. *Dialogas*, 15. <http://www.dialogas.com/laikrastis/dar-keli-isskirtiniai-lietuviu-kalbos-bruozai/>. (in Lithuanian).

Porter, M. F. (1980). An algorithm for suffix stripping. *Program*, 14(3): 130–137.

Radovanovic, M. Ivanovic, M. (2006). Document representations for classification of short web-page descriptions. In *Proceedings of the 8th international conference on Data Warehousing and Knowledge Discovery*, DaWaK'06, Krakow, Poland, pages 544–553.

Rimkutė E. and Daudaravičius V. (2007). Morfologinis Dabartinės lietuvių kalbos tekstyno anotavimas [Morphological annotation of the Lithuanian corpus]. *Kalbų studijos*, 11: 30–35. (in Lithuanian).

Savickienė, I., Kempe, V. and Brooks, P. J. (2009). Acquisition of gender agreement in Lithuanian: exploring the effect of diminutive usage in an elicited production task. *Journal of Child Language*, 36: 477–494.

Sebastiani, F. (2002). Machine Learning in Automated Text Categorization. *ACM Computing Surveys*, 34: 1–47.

Supermamos. (2012). Internet forum. May, 2012. <http://www.supermama.lt/forumas/>. (In Lithuanian).

Tan, C. M., Yuan-Fang, W., and Chan-Do, L. (2002). The use of bigrams to enhance text categorization. *Information Processing and Management*, 38(4): 529–546.

Ulvydas K., editor. (1965). *Fonetika ir morfologija (daiktavardis, būdvardis, skaitvardis, įvardis)* [Phonetics and morphology (noun, adjective, numeral, pronoun)], volume 1. Mintis, Vilnius. (in Lithuanian).

Volkens, A. (2002), *Manifesto coding instructions (second Revised Edition)*. Social Science Research Center Berlin (WZB), Wissenschaftszentrum Berlin, Germany.

Wahbeh, A., Al-Kabi, M., Al-Radaideh, Q. A., Al-Shawakfa, E. M., and Alsmadi, I. (2011). The Effect of Stemming on Arabic Text Classification: An Empirical Study. *International Journal of Information Retrieval Research*, 1(3): 54–70.

Wei, Z., Chauchat, J. H. and Miao, D. (2008). Comparing different text representation and feature selection methods on Chinese text classification using character n-grams. *Journées Internationnales d'Analyse des Données Textuelles*, Lyon, France, pages 1175–1186.

Willett, P. (2006). The Porter stemming algorithm: then and now. *Program: electronic library and information systems,* 40 (3): 219–223.

Yeh, A. (2000). More Accurate Tests for the Statistical Significance of Result Differences. In *Proceedings of the 18th International Conference on Computational Linguistics*, 2, pages 947–953.

Zinkevičius, V. (2000). Lemuoklis – morfologinei analizei [Morphological analysis with Lemuoklis]. In: Gudaitis, L. (ed.) *Darbai ir Dienos*, 24: 246–273. (in Lithuanian).

Generating "A for Alpha" When There Are Thousands of Characters

Hiroaki KAWASAKI[1] *Ryohei SASANO*[2]
Hiroya TAKAMURA[2] *Manabu OKUMURA*[2]
(1) Interdisciplinary Graduate School of Science and Engineering, Tokyo Institute of Technology
(2) Precision and Intelligence Laboratory, Tokyo Institute of Technology
kawa@lr.pi.titech.ac.jp, {sasano,takamura,oku}@pi.titech.ac.jp

ABSTRACT
The phonetic alphabet enables people to dictate letters of the alphabet accurately by using representative words, i.e., A for Alpha. Japanese kanji (idiographic Chinese characters) vastly outnumber the letters of the Roman alphabet, and thus Japanese requires an explanatory reading like a phonetic alphabet. We call the explanatory reading of a kanji a "distinctive explanation." Most kanji characters have their homophones, and the role of the distinctive explanations is to enable users to identify a specific kanji character only by listening to the explanation. In this paper, we propose a corpus-based method for automatically generating distinctive explanations for a kanji, in which information about familiarity and homophones of kanji are taken into consideration. Through the kanji-identification experiments, we show that the quality of the explanations generated by the proposed method is higher than that of the manually crafted distinctive explanations.

KEYWORDS: phonetic alphabet, distinctive explanation for a kanji.

Proceedings of COLING 2012: Technical Papers, pages 1411–1424,
COLING 2012, Mumbai, December 2012.

1 Introduction

Japanese has three types of characters: hiragana, katakana, and kanji (Chinese characters). While hiragana and katakana characters are phonograms, kanjis are ideograms, each of which usually has several readings. Most kanjis have homophones, and thus it is difficult to identify a kanji only by its reading. However, sometimes we need to identify a kanji verbally. For example, screen readers need to enable the users, especially visually impaired people who have difficulty seeing things, to identify a kanji only by sound. When we talk over the telephone, we have to exchange information, such as proper names, only by our voice. In such cases, we explain a kanji by using not only its reading but also its properties, compositions, and so on, to reduce the ambiguity. In this paper, we call such an explanation a "distinctive explanation for a kanji" and propose a method for generating distinctive explanations automatically.

One concept similar to a distinctive explanation is the phonetic alphabet, e.g., A for Alpha, B for Bravo, and C for Charlie. While a distinctive explanation explains kanjis, the phonetic alphabet explains letters and numbers. One of the major differences between them is the number of target characters. Whereas the phonetic alphabet for English deals with only 26 letters and 10 numbers, the distinctive explanation for kanjis deals with thousands of kanji characters.

The distinctive explanation leverages various aspects of the target kanji such as its Chinese reading, its Japanese reading, and its radicals, to reduce ambiguity. Each kanji has several readings, some of which are derived from Chinese readings and the others are Japanese traditional readings. If a kanji has a distinctive reading, the reading can be used to generate the distinctive explanation. Some kanjis can be divided into left and right parts and we can explain a kanji without ambiguity by using them. For example, the kanji "評 (*hyou*)" can be divided into "言 (*gon*)" and "平 (*hei*)." By listening to the information of "言 (*gon*)" and "平 (*hei*)," we can identify "評 (*hyou*)." Words including the target kanji, such as "購入 (*kou-nyū*, purchase)" for "購 (*kou*)," are also effective in identifying the target kanji because they can reduce the ambiguity.

Some screen readers (Lazar et al., 2007) already have functions for outputting distinctive explanations for kanjis. A screen reader is a software application for visually impaired people that reads aloud a text on a computer screen. This function enables visually impaired people to use e-mail, read news, view Web pages, and operate other complex applications. However, the existing screen readers use some distinctive explanations that do not make a target kanji easily identifiable, such as "*aya* for[1] *aya-ori* (綾織, twill)" for "綾 (*aya*)." "綾織 (*aya-ori*, twill)" is not a word with which most people are familiar, and thus we cannot identify the target kanji "綾 (*aya*)" easily. Watanabe et al. (2003) pointed out that the main factors that prevent the users from identifying the target kanji are the low familiarity and the homophones of the words used in the distinctive explanations.

Vocabulary and word familiarity vary among age groups, regions, and social backgrounds. It is hard work to remake a distinctive explanation for each kanji in accordance with changes in the target audience. We try to automate both the acquisition of vocabulary and word familiarity, and the generation of the distinctive explanations.

[1]In fact, distinctive explanations are expressed by using a Japanese word "*no*" as in "*kou-nyū no kou*." "*no*" is a Japanese postposition that can represent a wide range of semantic relations. It is similar to "for" in English. In this paper, we therefore refer to distinctive explanations by using "for" as in "*kou* for *kou-nyū*."

In this paper, we propose a method for automatically generating distinctive explanations for a kanji, and aim to improve the kanji identification rate. Our system automatically generates distinctive explanations using the knowledge of familiarity and homophones derived from a large text corpus. Automatic methods for generating the distinctive explanation can easily adapt to the users.

2 Distinctive explanation for kanji

With the growth of computers, screen readers that have functions for producing voice outputs of distinctive explanations have become popular among visually impaired people. Accordingly, people argued over the problem of what distinctive explanations should be outputted from screen readers.

Ooyama et al. (1996) proposed a spoken explanation generator, PLANET. This system can explain a kanji, especially those used in peoples' names, only by sound. When explaining, the system uses the information such as words containing the target kanji and the components of the kanji.

Watanabe et al. (2005a) manually produced distinctive explanations on the basis of children's vocabulary familiarity obtained by a kanji dictation survey. When producing distinctive explanations, they prioritized words that had higher familiarity than others and no homophones. Moreover, they avoided using negative expressions and English words like "*kin*[2] *for* gold," so that generated distinctive explanations would be suitable for elementary school students. The identification rate for their generated distinctive explanations was 14.1% higher than that for the existing distinctive explanations in the experiments of kanji dictation involving elementary school students.

Nishida et al. (2005) proposed distinctive explanations based on the meanings of a kanji. For example, the traditional distinctive explanation of "情報 (*jou-hou*, information)" was "*jou* for *jou-netsu* (情熱, passion)" and "*hou* for *hou-koku* (報告, report)." On the other hand, in distinctive explanations based on the meanings, "情報 (*jou-hou*, information)" was explained as "*i-n-fo-mē-sho-n* (インフォメーション, information)," "*chou-hou* (諜報, intelligence)," or "*hi-mitsu-jou-hou* (秘密情報, confidential information)." Experimental results showed no differences between the identification rates of the traditional distinctive explanations and explanations based on the semantics. Since those main target words are those that appear in a thesaurus, we cannot easily compare them and our distinctive explanations.

Watanabe et al. (2005b) classified in detail the composition of the traditional distinctive explanations derived from screen readers. Distinctive explanations can be classified into three types in accordance with their configuration:

Type 1 consists of a word that includes the target kanji and the reading of the target kanji in the word.

"*kou* for *kou-nyū* (購入, purchase)" for "購 (*kou*)"

Type 2 consists of the distinctive reading of the target kanji. This type uses the reading that other kanjis never have.

"*sakura* (桜, cherry blossom)" for "桜 (*sakura*)"

[2]In Japanese, *kin* means gold.

Type 3 uses the components of radicals of the target kanji or consists of meanings of the target kanji that forms a word only by itself.

"*kawa* with *sanzui*[3]" for "河 (*kawa*)"

They reported that Type 1 distinctive explanations were most common. Type 1 is suitable for use in statistical treatment, and therefore we aim to generate Type 1 distinctive explanations automatically in this paper.

Watanabe et al. also investigated the factors that make it difficult to identify the target kanji (Watanabe et al., 2003). We enumerate the major factors reported in their work:

Factor 1 The low familiarity of words such as "千代紙 (*chi-yo-gami*, Japanese paper with colored figures)."

Factor 2 The presence of homophones such as "購買 (*kou-bai*, purchase)" and "勾配 (*kou-bai*, gradient)."

Factor 3 The target kanji itself is difficult, such as "爾 (*ji*, thou)."

Factors 1 and 2 can be improved by selecting a suitable word for the distinctive explanation, but Factor 3 cannot because of the difficulty of the target kanji itself. Our study aims to improve the rate of identifying the target kanji from the distinctive explanation, and hence we focus on Factors 1 and 2.

3 Automatic generation of distinctive explanation for kanji

We propose an interactive system that automatically generates distinctive explanations for a kanji. Figure 1 shows the overview of our system. The first step outputs one Type 1 distinctive explanation. If the user cannot recall a kanji, the second step outputs another distinctive explanation.

The first step uses a single word that has high familiarity and few homophones. However, some kanjis are hard to identify from one word unambiguously; for example, "科 (*ka*)." While the most common words that include "科 (*ka*)" are "科学 (*ka-gaku*, science)," "教科 (*kyou-ka*, subject)," and "単科 (*tan-ka*, single subject)," all have several homophones such as "化学 (*ka-gaku*, chemistry)" for "科学 (*ka-gaku*)," "強化 (*kyou-ka*, reinforcement)" for "教科 (*kyou-ka*)," and "炭化 (*tan-ka*, carbonization)" and "単価 (*tan-ka*, unit price)" for "単科 (*tan-ka*, single subject)," and thus it is hard to identify "科 (*ka*)" unambiguously from a Type 1 distinctive explanation. For such a kanji, our system proceeds to the second step, and generates another distinctive explanation for the kanji by using a word that ensures a high kanji identification rate when combined with the word presented by the first step of our system.

The first example of the system's input in Figure 1 is the kanji "購 (*kou*)." The system outputs the distinctive explanation "*kou* for *kou-nyū*" by our first step and describes it to the user. The user identifies the correct kanji "購 (*kou*)" from the distinctive explanation. The second example of the system's input is the kanji "科 (*ka*)." The system outputs the distinctive explanation "*ka* for *ka-gaku*" by our first step and describes it to the user. The user cannot identify

[3] *sanzui* means "氵."

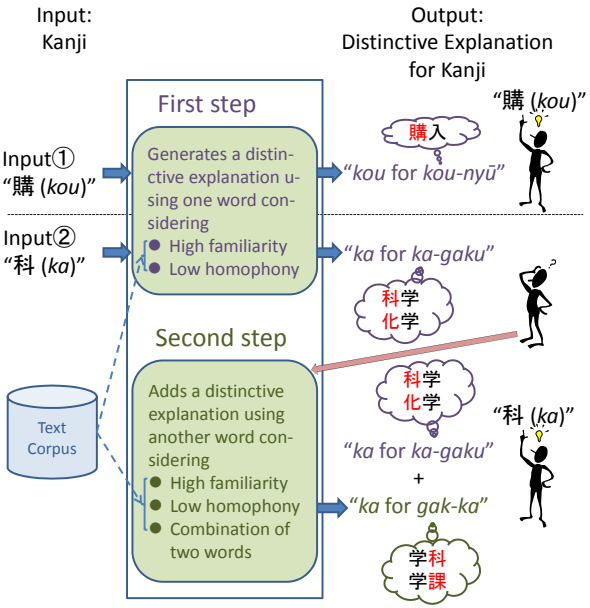


Figure 1: The overview of our system. If “購 (*kou*)” is input, the first step outputs “*kou* for *kou-nyū*.” In this case, the user will think of the correct kanji since there is no ambiguity. If “科 (*ka*)” is input, the first step outputs “*ka* for *ka-gaku*.” However, there are plural candidates such as “科 (*ka*)” and “化 (*ka*).” In such case, the user asks our system to generate an additional distinctive explanation. The second step then outputs "*ka* for *gak-ka*," and the user can identify the correct kanji.

the correct kanji because the reading “*ka-gaku*” has many homophones. Then the user asks our system to generate an additional distinctive explanation. Our system outputs the second distinctive explanation “*ka* for *gak-ka*,” and the user identifies the correct kanji “科 (*ka*)” from the two distinctive explanations.

3.1 Generation of distinctive explanations in the first step

The first step generates a Type 1 distinctive explanation as follows:

i. Extract words from the corpus that have more than two characters and include the target kanji.

ii. Calculate the score for each word w:

$$score_1(w) \triangleq p(w)^{\alpha} \cdot u_1(w)^{\beta}, \tag{1}$$

where

$p(w)$: the probability of w in the corpus,

$u_1(w)$: the ratio of the frequency of w to the frequency of all words in the corpus that have the same reading as w,

α: a parameter that reflects the importance of familiarity ($\in [0, 1]$),

β: a parameter that reflects the importance of not having homophones ($\in [0, 1]$).

The probability $p(w)$ is calculated as follows:

$$p(w) \triangleq \frac{c(w)}{\sum_{w' \in W} c(w')}, \tag{2}$$

where

$c(w)$: the frequency of w in the corpus,

W: the set of words that appear in the corpus.

The ratio $u_1(w)$ is calculated in the following way:

$$u_1(w) \triangleq \frac{c(w)}{\sum_{w' \in H(w)} c(w')}, \tag{3}$$

where

$H(w)$: the set of words that have the same reading as w.

iii. Select the word with the highest score and then use it to generate a Type 1 distinctive explanation.

In Equation (1), $p(w)$ represents the degree of familiarity and $u_1(w)$ represents the degree of uniqueness. For example, the most common words that include "購 (*kou*)" are "購読 (*kou-doku*, subscription)," "購入 (*kou-nyū*, purchase)," and "購買 (*kou-bai*, purchase)." While "購読 (*kou-doku*, subscription)" has higher probability than the other words, it has homophones such as "鉱毒 (mining pollution)." $u_1(w)$ functions to reduce the score of such ambiguous words. As a result, our system prioritizes the output of the distinctive explanation using the word "購入 (*kou-nyū*, purchase)" rather than "購読 (*kou-doku*, subscription)."

3.2 Generation of distinctive explanations in the second step

The second step generates an additional distinctive explanation that reduces ambiguity when combined with the distinctive explanation by the first step.

i. Extract words from the corpus that have more than two characters and include the target kanji.

ii. Give a score for each pair of the word selected in the first step w_1, and a word extracted in i. The score is calculated in the following way:

$$score_2(w_1, w) \triangleq score_1(w_1) \cdot score_1(w) \cdot u_2(w_1, w)^{\gamma}, \tag{4}$$

where

$u_2(w_1, w)$: the ratio of the number of the target kanjis to the number of the kanjis that we can recall from the pairs composed of w_1 and w,

γ: a parameter that reflects the importance of combination ambiguity ($\in [0, 1]$).

The ratio $u_2(w_1, w)$ is calculated in the following way:

$$u_2(w_1, w) \triangleq \frac{\min(c(w_1), c(w))}{\sum_{(w_1', w') \in C} \min(c(w_1'), c(w'))}, \tag{5}$$

where

C: the set of candidate pairs of words that have the same reading as w_1 and w and can make the user recall a kanji.

iii. Select the word with the highest score and then generate a distinctive explanation by using w besides w_1.

Equation (4) consists of a product of $score_1(w_1)$, $score_1(w)$, and $u_2(w_1, w)^{\gamma}$. $u_2(w_1, w)^{\gamma}$ represents the uniqueness when using two words. For example, the distinctive explanation "*ka* for *ka-gaku*" and "*ka* for *tan-ka*" evoke at least two kanjis: "科 (*ka*)" and "化 (*ka*)." The candidate kanjis for the distinctive explanation "*ka* for *ka-gaku*" are "科 (*ka*)" from "科学 (*ka-gaku*, science)" and "化 (*ka*)" from "化学 (*ka-gaku*, chemistry)." Similarly, those for "*ka* for *tan-ka*" are "科 (*ka*)" from "単科 (*tan-ka*, single subject)" and "化 (*ka*)" from "炭化 (*tan-ka*, carbonization)" and "価 (*ka*)" from "単価 (*tan-ka*, unit price)." Thus we have $C = \{($ 科学, 単科 $), ($ 化学, 炭化 $)\}$. The term $u_2(x, y)^{\gamma}$ reduces the scores of such ambiguous distinctive explanations. Our system outputs distinctive explanations that are less ambiguous, such as "*ka* for *ka-gaku*" and "*ka* for *gak-ka*."

4 Experiments

4.1 Experimental Settings

We used three corpora in experiments:

- Google Japanese N-gram corpus (Google corpus)
- Yomiuri newspaper corpus (Yomiuri corpus)
- Balanced Corpus of Contemporary Written Japanese (BCCWJ)

The Google Japanese N-gram corpus (Kudo and Kazawa, 2007) was constructed from 20 billion Japanese sentences on the Web and consists of 255 billion words. The Yomiuri newspaper corpus consists of 400 million words in Yomiuri newspaper articles from 1991 to 2004. The

BCCWJ (Maekawa, 2008) is a balanced corpus of 100 million words of contemporary written Japanese.

We used MeCab (Kudo et al., 2004), an open-source morphological analyzer, to separate the corpus into words. We applied the IPA dictionary with the parameter estimated by Conditional Random Fields (CRF) based on IPA corpus[4]. We also used the Simple Kana to Kanji conversion program (SKK) dictionary[5] to utilize words longer than those in the IPA dictionary. This is because the words in the IPA dictionary tend to be too short for our method. For example, although we want to use "炒め物 (*ita-me-mono*, fried food)" to generate a distinctive explanation for "炒 (*ita*)," there is no entry for "炒め物" in the IPA dictionary, and thus "炒め物" is divided into two words: "炒め (*ita-me*, fried)" and "物 (*mono*, object)." On the other hand, there is an entry for "炒め物" in the SKK dictionary, and in such cases we consider the word as a candidate word for generating distinctive explanations.

We used the kanjis selected from top 2,000 frequently occurring kanjis in the Google corpus for experiments. This is because we want to focus on the performance of generating distinctive explanations, and thus we want to ignore errors caused by Factor 3: the target kanji itself is difficult. Note that since the total frequency of the 2,000 kanjis covered more than 99% kanji occurrences, our experimental setting is very practical. We set the parameters (α, β, γ) to $(0.1, 1.0, 1.0)$ in accordance with the results of a preliminary experiment.

4.2 Comparison of the three corpora

We first conducted a preliminary experiment to evaluate distinctive explanations generated by the first step to examine which corpus is the most preferable for our method. We prepared four distinctive explanations for each kanji: three are generated by using each corpus (Google corpus, Yomiuri corpus, and BCCWJ) and one is the distinctive explanation used in the screen reader PC-Talker XP for comparison. We randomly selected 100 kanjis for an evaluation from the 2,000 most frequently occurring kanjis that had Type 1 distinctive explanations in PC-Talker XP such as "*kou* for *kou-nyū* (購入, purchase)" for "購 (*kou*)."

The distinctive explanations were evaluated by eight human subjects. These distinctive explanations were written on paper, randomly shuffled, and shown to the subjects. Each subject was shown only one distinctive explanation for each kanji. Since there were eight human subjects and four types of distinctive explanations were generated for each kanji, each distinctive explanation was evaluated by two subjects. The subjects were requested to think of the most likely kanji from the presented distinctive explanation and to choose one from the following choices:

a. I thought of a kanji that matched the target kanji.

b. I thought of a kanji that did not match the target kanji.

c. I could not think of any kanji.

We did not conduct a kanji dictation test when we evaluated the distinctive explanations. Japanese speakers cannot always write out kanjis that they can identify, probably because of

[4]http://en.sourceforge.jp/projects/ipadic/
[5]http://openlab.ring.gr.jp/skk/index.html

	Google corpus	Yomiuri corpus	BCCWJ	PC-Talker XP
a	179	170	185	185
b	15	15	9	9
c	6	15	6	6
IR[%]	89.5	85.0	92.5	92.5

Table 1: The identification rates for three corpora.

the spread of computers, which let us input kanjis simply by selecting the correct one instead of actually writing it. This situation is similar to the fact that some English speakers cannot always spell familiar words correctly.

We calculated the identification rate (IR), i.e., the percentage of successfully identified kanji, for each corpus, as follows:

$$\mathrm{IR} = \frac{n(\mathbf{a})}{n(\mathbf{a}) + n(\mathbf{b}) + n(\mathbf{c})} \times 100\ [\%] \tag{6}$$

where $n(x)$ is the number of times choice x is selected.

Table 1 shows the identification rates for three corpora. PC-Talker XP and our system based on BCCWJ achieved the highest identification rate (92.5%). We conducted the McNemar's test and confirmed that there were significant differences at the 0.05 significance level between our system with Yomiuri corpus and PC-Talker XP, and between our system with Yomiuri corpus and with BCCWJ.

Table 2 shows the examples of distinctive explanations and their evaluation. Distinctive explanations generated by the Yomiuri corpus (newspaper articles) tend to use difficult words such as "*gai* for *gai-bou* (外貌, exterior)" generated for "貌 (*bou*)" and "*gi* for *yo-gi-nai* (余儀無い, unavoidable)" generated for "儀 (*gi*)," while there are easier words such as "美貌 (*bi-bou*, beauty)" and "儀式 (*gi-shiki*, ceremony)." This tendency would be one of the factors of the lower identification rate of distinctive explanations generated by the Yomiuri corpus.

4.3 Evaluation of the proposed method

We then evaluated the proposed method with both steps. We used the output of PC-Talker XP as a comparison. On the basis of the results in Sec. 4.2, we used BCCWJ as the corpus in this experiment. We used 100 kanjis randomly selected from the 2,000 most frequent kanjis that were not limited to kanjis that had Type 1 distinctive explanations in PC-Talker XP. Table 3 shows the number of kanjis for each type used in PC-Talker XP.

We evaluated 200 distinctive explanations: 100 generated from our method and 100 extracted from PC-Talker XP. Sixty subjects were each shown 50 distinctive explanations. Thus each distinctive explanation was evaluated by 15 subjects.

When evaluating our system, we first asked the subjects to think of a kanji from the distinctive explanation generated by the first step described in the paper. If the subjects could not think of a kanji, we asked the subjects to look at the distinctive explanation generated by the second step and to think of a kanji. After that, we asked the subjects to choose one from the following:

Kanji	Google corpus	Yomiuri corpus	BCCWJ	PC-Talker XP
儀	"*gi* for *sou-gi*" 葬儀 (2/2) funeral	"*gi* for *yo-gi-na-i*" 余儀無い (0/2) unavoidable	"*gi* for *gi-shiki*" 儀式 (2/2) ceremony	"*gi* for *gi-shiki*" 儀式 (2/2) ceremony
貌	"*bou* for *bi-bou*" 美貌 (2/2) beauty	"*bou* for *gai-bou*" 外貌 (0/2) exterior	"*bou* for *bi-bou*" 美貌 (2/2) beauty	"*bou* for *bi-bou*" 美貌 (2/2) beauty
感	"*kan* for *kan-ji*" 感じ (0/2) feeling	"*kan* for *kan-jiru*" 感じる (2/2) feel	"*kan* for *kan-jiru*" 感じる (2/2) feel	"*kan* for *kan-shin-suru*" 感心する (2/2) be impressed
遥	"*you* for *you-hai*" 遥拝 (0/2) worshipping from afar	"*you* for *you-hai*" 遥拝 (0/2) worshipping from afar	"*you* for *you-hai*" 遥拝 (0/2) worshipping from afar	"*haru* for *haru-ka-kanata*" 遥か彼方 (2/2) far away
餅	"*hei* for *sen-bei*" 煎餅 (1/2) rice cracker	"*mochi* for *kiri-mochi*" 切餅 (1/2) sliced cracker	"*hei* for *sen-bei*" 煎餅 (2/2) rice cracker	"*mochi* for *mochi-tsuki*" 餅つき (1/2) mochi pounding
欄	"*ran* for *kū-ran*" 空欄 (2/2) blank	"*ran* for *ran-kan*" 欄干 (0/2) parapet	"*ran* for *ran-kan*" 欄干 (1/2) parapet	"*ran* for *ran-gai*" 欄外 (2/2) margin
点	"*ten* for *kyo-ten*" 拠点 (1/2) stronghold	"*ten* for *kyo-ten*" 拠点 (1/2) stronghold	"*ten* for *kan-ten*" 観点 (0/2) standpoint	"*ten* for *ten-sū*" 点数 (2/2) score
輸	"*yu* for *yu-nyū*" 輸入 (2/2) importation	"*yu* for *yu-nyū*" 輸入 (2/2) importation	"*yu* for *yu-nyū*" 輸入 (2/2) importation	"*yu* for *yu-shutu-su-ru*" 輸出する (2/2) export

Table 2: Examples of distinctive explanations and their evaluation. "$(n/2)$" means n subjects out of two chose **a**.

- I thought of a specific kanji from only the first distinctive explanation, and the kanji was
  $\mathbf{a_1}$. correct, $\mathbf{b_1}$. wrong.
- I thought of a specific kanji from the first and the second distinctive explanation, and the kanji was
  $\mathbf{a_2}$. correct, $\mathbf{b_2}$. wrong.
- **c**. I could not think of any kanji.

To evaluate our whole system, we regarded $\mathbf{a_1}$ and $\mathbf{a_2}$ as positive answers and calculated the identification rate (IR_2) as follows:

$$\mathrm{IR}_2 = \frac{n(\mathbf{a_1}) + n(\mathbf{a_2})}{n(\mathbf{a_1}) + n(\mathbf{b_1}) + n(\mathbf{a_2}) + n(\mathbf{b_2}) + n(\mathbf{c})} \times 100.\ [\%] \tag{7}$$

We also evaluated the distinctive explanations generated only by the first step in our system for comparison. For this evaluation, we regarded $\mathbf{a_1}$ as a positive answer and calculated the identification rate (IR_1) as follows:

$$\mathrm{IR}_1 = \frac{n(\mathbf{a_1})}{n(\mathbf{a_1}) + n(\mathbf{b_1}) + n(\mathbf{a_2}) + n(\mathbf{b_2}) + n(\mathbf{c})} \times 100.\ [\%] \tag{8}$$

Type	#	Example
Type 1	93	"*ka* for *ka-zei-su-ru* (課税する, tax)" "*ken* for *ken-ka-su-ru* (喧嘩する, fight)"
Type 2	0	-
Type 3	7	"*yorokobi-wo-imi-suruka* (happiness)" "*tsuchi-wo-hutatu-kasaneta kei* (to stack soil on soil)"

Table 3: The number of kanjis for each type used in PC-Talker XP.

	Our system using BCCWJ	PC-Talker XP
$\mathbf{a_1}$	1,181	1,301
$\mathbf{b_1}$	28	58
$\mathbf{a_2}$	163	-
$\mathbf{b_2}$	22	-
c	106	141
IR[%]	IR_1: 78.7, IR_2: 89.6	IR_{SR}: 86.7

Table 4: Evaluation results of our system outputs and distinctive explanations in screen reader.

To evaluate the distinctive explanations in PC-Talker XP, we asked the subjects to choose one from the following options:

a. I thought of a specific kanji that was correct.

b. I thought of a specific kanji that was wrong.

c. I could not think of any kanji.

For evaluating the screen reader, we regarded **a** as a positive answer and calculated the identification rate (IR_{SR}) as follows:

$$IR_{SR} = \frac{n(\mathbf{a})}{n(\mathbf{a}) + n(\mathbf{b}) + n(\mathbf{c})} \times 100. \ [\%] \tag{9}$$

Table 4 shows the results. We confirmed that our whole system outperformed PC-Talker XP. We conducted the McNemar's test and confirmed that our whole system (IR_2) and PC-Talker XP significantly differed at the 0.05 level. Distinctive explanations generated by our system seem to be longer and take more time to listen to than those of PC-Talker XP. However, users do not always need to hear the entire distinctive explanation of our system to think of a kanji. Table 5 shows the average length of distinctive explanations shown to the subjects. In 80.6 %[6] of cases, the target kanjis were correctly thought of in the first step. In addition, the average length of the first step's output was shorter than that of distinctive explanations of PC-Talker XP. The average length of our system output was 8.14, which was shorter than that of PC-Talker XP, and thus the comparison of (IR_2) and (IR_{SR}) is fair.

The identification rate of the first step (IR_1) was lower than both those of the screen reader (IR_{SR}) and the rate in Sec. 4.2 (IR). We think there are two reasons for this. The first

[6] $(1,181 + 28)/1,500 = 0.806$

	First step	Second step	The average length
Our system	6.80	13.93	8.14
PC-Talker XP	-	-	8.96

Table 5: The average lengths of distinctive explanations shown to the subjects.

Kanji	Our system		PC-Talker XP
	First step	Second step	
悟	"*go* for *kaku-go*" 覚悟 (9/15) preparation	"*sato* for *sato-ri*" 悟り (14/15) enlightenment	"*go* for *kaku-go, sato-ru*" 覚悟; 悟る (13/15) preparation; to realize
課	"*ka* for *ka-dai*" 課題 (13/15) assignment	"*ka* for *ka-zei*" 課税 (15/15) taxation	"*ka* for *ka-zei-suru*" 課税する (6/15) tax
灌	"*kan* for *yu-kan*" 湯灌 (1/15) wash a dead body	"*kan* for *kan-gai-you-sui*" 灌漑用水 (7/15) irrigation	"*kan* for *kan-gai-suru, soso-gu*" 灌漑する; 灌ぐ (1/15) irrigate; pour
藍	"*ran* for *ga-ran*" 伽藍 (0/15) temple	"*ai* for *ai-hara*" 藍原 (4/15) family name	"*ai* for *ai-iro*" 藍色 (12/15) indigo blue
圭	"*kei* for *kei-ji-rou*" 圭二郎 (2/15) first name	"*kei* for *kei-ichi*" 圭一 (4/15) first name	"*tsuchi-wo-hutatu-kasaneta kei*" (11/15) to stack soil on soil
嘩	"*ka* for *hū-hu-gen-ka*" 夫婦喧嘩 (4/15) marital quarrel	"*ka* for *ō-gen-ka*" 大喧嘩 (5/15) big fight	"*ken* for *ken-ka-suru*" 喧嘩する (1/15) fight
嘉	"*ka* for *ka-ei*" 嘉永 (0/15) era name	"*ka* for *ka-de-na*" 嘉手納 (0/15) place-name	"*yoroko-bi-wo-i-mi-suru ka*" 嘉 (1/15) that means happiness

Table 6: Examples of distinctive explanations generated by the whole our system and distinctive explanations in PC-Talker XP and their evaluation. "$(n/15)$" means n subjects out of 15 chose a positive answer.

is the differences between evaluation methods. While the prior evaluation (IR) contained the possibility of positive evaluation when subjects thought of multiple kanjis, this evaluation (IR_1) did not. The second is the use of different kanjis. While kanjis are limited to those that have Type 1 distinctive explanations in PC-Talker XP in the prior evaluation (IR), all kanjis were allowed in this evaluation. Even for the kanji unsuitable for the Type 1 distinctive explanation, our system has to output Type 1 distinctive explanations.

Table 6 shows examples of distinctive explanations and their evaluation. We confirmed that our system generates a better distinctive explanation for "課 (*ka*)" and "灌 (*kan*)." In the case of "課 (*ka*)," 13 subjects out of 15 successfully identified "課 (*ka*)" from distinctive explanations generated by the first step. However, two subjects could not identify the target kanji. This would be because our system outputted the distinctive explanation using the word "課題 (*ka-dai*, assignment)," which has homophones such as "過大 (*ka-dai*, excessive)" and "仮題 (*ka-dai*, a tentative title)." Since the remaining two identified the correct kanji by using distinctive explanations generated by the second step, we confirmed the effectiveness of the

proposed two-step method. On the other hand, only six subjects identified kanji from distinctive explanations of the screen reader. The cause of the low identification rate of the screen reader may be that subjects thought of "*ka-sei-suru* (加勢する, assist)" instead of "*ka-zei-suru* (課税する, tax)." In the case of "灌 (*kan*)," only one subject out of 15 identified "灌 (*kan*)" from our first step or the screen reader. However, when other subjects looked at distinctive explanations generated by the second step, seven identified "灌 (*kan*)." It can be inferred from these results that "灌 (*kan*)" is difficult to identify but our two-step approach is effective in such a case.

Conversely, the examples where our system was worse than the screen reader were the cases of "藍 (*ai* or *ran*)" and "圭 (*kei*)." In the case of "藍 (*ai* or *ran*)," our first step could not make anyone identify the kanji. In addition, even our second step made only four subjects identify the kanji. However, the screen reader succeeded in making 12 subjects identify the kanji. This is because our system cannot capture the specific features of "藍 (*ai* or *ran*):" in Japanese, "藍色 (*ai-iro*, indigo blue)" is a color. While the screen reader used words related to the color, our system used "伽藍 (*ga-ran*, temple)" and "藍原 (*ai-hara*, which is a Japanese family name)." The neither word includes information about the color. Such kanji-specific information is important but our system cannot use it well. For "圭 (*kei*)," our system used ambiguous words, and most subjects failed to identify the kanji[7]. In contrast, the screen reader achieved a high identification rate for this kanji, by using the distinctive explanation "Write 土 on top of 土 (*tsu-chi*, soil)." The use of kanji components or radicals as in this example by the screen reader, on top of our method, will further improve the identification performance.

Although we selected the top 2,000 frequent kanjis for the candidates of evaluation in order to eliminate the difficult kanji, some difficult kanjis still appeared. For example, in the case of "嘉[8] (*ka*)," our system and the screen reader made barely any subjects identify the kanji. We think that this is because "嘉 (*ka*)" itself is difficult.

Conclusion

In this paper, we proposed a method for automatically generating distinctive explanations for a kanji using a text corpus. The proposed method took into account familiarity and homophones of kanjis. As a result of human evaluation, we confirmed that distinctive explanations generated by our system outperformed those in existing screen readers.

Our future work involves incorporation of intonation, application to Chinese, and user adaptation. Intonation of words can help generate good distinctive explanations. For example, "橋 (*hashi*, bridge)" and "箸 (*hashi*, chopstick)" have the same readings but different intonations, so we think intonation can be a clue for identifying a kanji. Kanjis are used in not only Japanese but also Chinese. Since our proposed method is language-independent, our method can be applied to distinctive explanations for Chinese. Finally, we are considering generating distinctive explanations that consider the user attributes. For example, users who have studied the law will be familiar with legal terms but not medical terms. To adapt our system to different users, we can select a corpus that is suitable for them.

[7] "*Kei-ji-rou*" has homophones such as "慶二郎," "敬二郎," and "啓二郎" and "*kei-ichi*" have homophones such as "恵一," "慶一," and "啓一." All are Japanese male names.

[8] "嘉 (*ka*)" means happiness, but this kanji is rarely used.

References

Kudo, T. and Kazawa, H. (2007). Japanese web n-gram corpus version 1. *Google/Linguistic Data Consortium*.

Kudo, T., Yamamoto, K., and Matsumoto, Y. (2004). Applying conditional random fields to Japanese morphological analysis. *In Proceedings of EMNLP*, pages 230–237.

Lazar, J., Allen, A., Kleinman, J., and Malarkey, C. (2007). What frustrates screen reader users on the web: A study of 100 blind users. *Int. J. Hum. Comput. Interaction*, pages 247–269.

Maekawa, K. (2008). Balanced corpus of contemporary written Japanese. *Proceedings of the 6th Workshop on Asian Language Resources*, pages 101–102.

Nishida, M., Horiuchi, Y., and Ichikawa, A. (2005). Kanji transformation using semantic information for blind people. *IEICE technical report. Welfare Information technology (In Japanese)*, 105(186):1–6.

Ooyama, Y., Asano, H., and Matsuoka, K. (1996). Spoken-style explanation generator for japanese kanji using a text-to-speech system. 3:1369–1372.

Watanabe, T., Teruyoshi, F., Watanabe, B., Sawada, M., and Kamata, K. (2003). A consideration on shosaiyomi (explanatory expressions) used in screen readers for visually-impaired persons. *Technical report of IEICE. HCS (In Japanese)*, 102(599):25–28.

Watanabe, T., Watanabe, B., Okada, S., Yamaguchi, T., Oosugi, N., and Sawada, M. (2005a). A study on shosaiyomi of screen readers kanji writing test using newly devised shosaiyomi. *IEICE technical report. Welfare Information technology (In Japanese)*, 105(373):7–12.

Watanabe, T., Watanebe, B., Fujinuma, T., Osugi, N., Sawada, M., and Kamata, K. (2005b). Major factors that affect comprehensibility of shosaiyomi (explanatory expressions) used in screen readers: Consideration based on classification of shosaiyomi and kanji writing test. *The transactions of the Institute of Electronics, Information and Communication Engineers. D-I (In Japanese)*, 88(4):891–899.

A machine learning approach for phenotype name recognition

Maryam Khordad[1] *Robert E. Mercer*[1] *Peter Rogan*[1,2]
(1)Department of Computer Science
(2)Department of Biochemistry
The University of Western Ontario, London, ON, Canada
mkhordad@csd.uwo.ca, mercer@csd.uwo.ca, progan@uwo.ca

ABSTRACT
Extracting biomedical named entities is one of the major challenges in automatic processing of biomedical literature. This paper proposes a machine learning approach for finding phenotype names in text. Features are included in a machine learning infrastructure to implement the rules found in our previously developed rule-based system. The system also uses two available resources: MetaMap and HPO. As we are not aware of any available corpus for phenotype names, a corpus has been constructed. Since manual tagging of the corpus was not possible for us, we started tagging only HPO phenotypes in the corpus and then using a semi-supervised learning method, the tagging process improved. The evaluation results (F-Score 92.25) suggest that the system achieved good performance and it outperforms the rule-based system.

KEYWORDS: Phenotype, Named Entity Recognition, MetaMap, Human Phenotype Ontology.

Proceedings of COLING 2012: Technical Papers, pages 1425–1440,
COLING 2012, Mumbai, December 2012.

1 Introduction

A large amount of biomedical knowledge is available in the biomedical literature. So, automatic processing of biomedical literature to capture and formalize this embedded information is very demanding. Current Natural Language Processing (NLP) systems try to extract from the biomedical literature different types of knowledge such as, protein–protein interactions (Leroy et al., 2003) (He and DiMarco, 2005) (Fundel et al., 2007) (kiong Ng and Wong, 1999) (Yu et al., 2005) (Bui et al., 2011), new hypotheses(Swanson, 1986) (Hristovski et al., 2005) (Hristovski et al., 2006), relations between drugs, genes, and cells (Rindflesch et al., 2000) (Friedman et al., 2001) (Tanabe et al., 1999), relations between genes and diseases (Rindflesch et al., 2003) (Coulet et al., 2010), protein structure (Humphreys et al., 2000) (Gaizauskas et al., 2003), and protein function (Andrade and Valencia, 1998) (Valencia, 2005).

Fundamental to each of these applications is the Named Entity Recognition and Classification (NERC) task. Research over the past years has shown that recognizing the names of biomedical objects is not a simple task. Factors that preclude a straightforward procedure include the existence of an ever-increasing large (millions) set of entity names, a penchant for the use of abbreviations, the use of synonyms, and the fact that some biological entities have complex names consisting of many words, like "increased erythrocyte adenosine deaminase activity", and lacking agreement on the name boundaries, even among biologists (Leser and Hakenberg, 2005).

Named Entity Recognition and Classification in the biomedical domain has been extensively studied. As a consequence, many methods have been proposed. Generic methods, like MetaMap (Aronson, 2001) and mgrep (Dai et al., 2008) recognize and classify many kinds of entities in text. Some methods, however, are specialized to recognize particular types of entities like gene or protein names (Krauthammer et al., 2000) (Gaizauskas et al., 2003), diseases and drugs (Rindflesch et al., 2000) (Xu et al., 2008) (Segura-Bedmar et al., 2008), mutations (Horn et al., 2004), and properties of protein structures (Gaizauskas et al., 2003). Each method employs one or more of the following techniques (Leser and Hakenberg, 2005): (1) dictionary-based techniques (like (Krauthammer et al., 2000)) which match phrases from the text against existing dictionary entries, (2) rule-based techniques ((Fukuda et al., 1998), for instance) which make use of lexical and linguistic rules to find entity names in the text, and (3) machine learning techniques (for example, (Nobata et al., 1999)) which treat the NER task as a classification problem. Some methods use hybrid approaches to find named entities: ChemSpot (Rocktäschel et al., 2012) blends Conditional Random Fields with dictionary matching to identify chemicals in texts, a biomedical name entity recognizer (Gong et al., 2009) uses POS tagging, rules and a dictionary, and a protein name recognizer (Seki and Mostafa, 2005) uses rules, a probabilistic model and a dictionary to find protein names in biomedical text.

Every day many research experiments are performed to discover the role of DNA sequence variants in human health and disease and the results of these experiments are published in the biomedical literature. Because of the large quantity of information, a reliable automatic system to extract this information for future organization is desirable. Human phenotypes comprise a very important part of this knowledge. Phenotypes are the observable characteristics of a cell or organism, including its appearance, its morphology, physiology and ways of life. A phenotype of an organism is determined by the interaction of its genetic constitution and the environment. Skin colour, height and behaviour are some examples of phenotypes.

Currently, many systems which use phenotypes to find information like phenotype-genotype re-

lations (for instance, (Coulet et al., 2010)) use only dictionary-based techniques to recognize the phenotypes in the text. Although many biomedical-term-specialized dictionaries are available, we are not aware of a dictionary which is both comprehensive and ideally suited for phenotype name recognition. For example, the Unified Medical Language System (UMLS) Metathesaurus (Humphreys et al., 1998) is a very large, multi-purpose, and multi-lingual vocabulary database that contains more than 1.8 million concepts. All concepts in the Metathesaurus are assigned to at least one semantic type, but *Phenotype* is not one of them. So, it alone is not adequate to distinguish between phenotypes and other objects in text. In addition, some phenotype names do not exist in the UMLS MetaThesaurus. The Pharmacogenetics Knowledge Base (PharmGKB) (Klein et al., 2001) attempts to collect all knowledge of how human genetic variation impacts drug–response phenotypes. It is a high quality database queried by clinicians and bioinformaticians. It is a manually curated database that summarizes published gene–drug–phenotype relationships. Nevertheless this manual curation process is not sustainable considering the growth of the scientific literature in this domain. The Online Mendelian Inheritance in Man (OMIM) (McKusick, 2007) is the most important single information source about human genes and genetic phenotypes (Robinson and Mundlos, 2010). Nonetheless, OMIM does not use a controlled vocabulary to describe the phenotypic features in its clinical synopsis section making it inappropriate for data mining purposes (Robinson and Mundlos, 2010). And, it is manually curated. The Human Phenotype Ontology (HPO) (Robinson and Mundlos, 2010) has been developed using information from OMIM. Although it contains approximately 10,000 terms, it is incomplete. Also, new phenotypes are being constantly introduced to the biomedical world and HPO currently is being refined, corrected, and expanded manually, meaning that it will have difficulty keeping pace with all the new phenotypes.

To our knowledge the only method to extract phenotype names automatically from text is the rule-based system we proposed in (Khordad et al., 2011). In the current paper we discuss a machine-learning-and-dictionary-based NER system that recognizes the human phenotype names in molecular biology literature which has been inspired by the rule-based method mentioned above. We have integrated existing databases (UMLS Metathesaurus(Humphreys et al., 1998) and the Human Phenotype Ontology (HPO) (Robinson and Mundlos, 2010)) and tools (MetaMap (Aronson, 2001) and BANNER (Leaman and Gonzalez, 2008)) to achieve our goal.

2 Background

2.1 MetaMap

MetaMap, a program developed by the National Library of Medicine (NLM) (Aronson, 2001), provides a link between biomedical text and the structured knowledge in the Unified Medical Language System (UMLS) Metathesaurus. To map phrases in the text to concepts in the UMLS Metathesaurus, MetaMap analyzes the input text lexically and semantically. First, MetaMap tokenizes the input text. In the tokenization process the input text is broken into meaningful elements, like words. After part of speech tagging and shallow parsing using the Specialist Lexicon, the text has been broken into phrases. Phrases then undergo further analysis: Each phrase is mapped to a set of candidate UMLS concepts, each candidate being given a score that represents how well the phrase matches the candidates. An optional last step is word sense disambiguation (WSD) which chooses the best candidate with respect to the surrounding text (Aronson, 2001).

MetaMap is configurable with options for vocabularies and data models in use, output format

```
Phrase: "at diagnosis."
>>>>> Phrase
diagnosis
<<<<< Phrase
>>>>> Candidates
Meta Candidates (6):
  1000 Diagnosis [Finding]
  1000 Diagnosis (Diagnosis:Impression/interpretation of study:
       Point in time:^Patient:Narrative) [Clinical Attribute]
  1000 Diagnosis (Diagnosis:Impression/interpretation of study:
       Point in time:^Patient:Nominal) [Clinical Attribute]
  1000 diagnosis (diagnosis aspect) [Qualitative Concept]
  1000 DIAGNOSIS (Diagnosis Study) [Research Activity]
   928 Diagnostic [Functional Concept]
<<<<< Candidates
>>>>> Mappings
Meta Mapping (1000):
  1000 diagnosis (diagnosis aspect) [Qualitative Concept]
<<<<< Mappings
```

Figure 1: MetaMap output for “at diagnosis.”

and algorithmic computations. An example of the human-readable output format for the text “at diagnosis.” is shown in Figure 1. MetaMap finds 6 candidates for this phrase and after WSD it maps the phrase to the “diagnosis aspect” concept. In UMLS each Metathesaurus concept is assigned to at least one semantic type. In Figure 1 the semantic type of each concept is given in the square brackets. Semantic types are categorized into groups, called Semantic Groups (SG), that are subdomains of biomedicine such as Anatomy, Living Beings and Disorders (McCray et al., 2001). Each semantic type belongs to exactly one SG.

2.2 The Human Phenotype Ontology (HPO)

The Human Phenotype Ontology (HPO) (Robinson and Mundlos, 2010) provides a standardized vocabulary of phenotypic abnormalities encountered in human disease. The HPO was constructed using information initially obtained from the Online Mendelian Inheritance in Man (OMIM) (McKusick, 2007) expanded with synonym terms. The hierarchical structure in the HPO represents the subclass relationship. The HPO currently contains over 9500 terms.

2.3 The Rule-Based Method

In (Khordad et al., 2011) we proposed a rule-enhanced dictionary-based named entity recognition system based on MetaMap. The block diagram of the system is shown in Figure 2. In this system the input text is processed and each Noun Phrase is tagged by MetaMap. The UMLS semantic types are categorized into 15 more general and comprehensive categories called Semantic Groups (SG) (McCray et al., 2001). The definition of the SG *Disorders* is close to the meaning of Phenotype. SG *Disorders* contains 12 semantic types. (Khordad et al., 2011) categorized these semantic types into two categories : *Phenotypes* and *Phenotype Candidates*. Using the MetaMap output, Disorder Recognizer considers noun phrases in the Phenotype category as phenotypes and searches for the Phenotype Candidates in HPO to see whether they are phenotypes or not. OBO-Edit (an open source Java program) (Day-Richter et al., 2007)

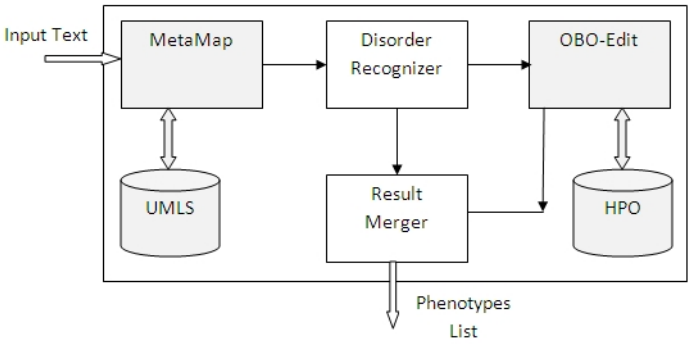


Figure 2: Rule-Based System Block Diagram.

has been used for searching in HPO. To improve the results 5 stylistic and linguistic rules are employed. These rules include:

- Rule 1: Resolve the acronym referencing problem by making and using a list of acronyms occurring in a paper.
  Often the names of phenotypes are used in acronym form. This makes it difficult to recognise them. However we can find the full form of them in their first usage.
- Rule 2: The semantic type of a noun phrase is the semantic type assigned by Metamap to its head.
  Sometimes MetaMap breaks a noun phrase to different parts with different Semantic Types. Using this rule it is possible to assign one semantic type to the whole noun phrase.
- Rule 3: If a phrase is "modifier (from the list of special modifiers (Burgun et al., 2009)) + [Anatomy] or [Physiology]" it is a phenotype.
  Some phenotypes follow a special pattern. There is a list of special modifiers (Burgun et al., 2009) which if come before a Noun Phrase in the SG *Anatomy* or *Physiology* make a phenotype.
- Rule 4: If the single form of a phrase is a phenotype, the plural form is a phenotype, too.
  This rule applies while searching in HPO, when the phenotype number in HPO is not in agreement with the noun phrase number in text.
- Rule 5: If the head of a phenotype candidate phrase is a phenotype, the whole phrase is a phenotype.
  This rule is also useful for searching in HPO. Sometimes a noun phrase in text contains a word in HPO but it is surronding with adjective and adverbs.

2.4 BANNER

BANNER (Leaman and Gonzalez, 2008) is an open-source biomedical named entity recognition system implemented using second order conditional random fields (CRF), a machine learning technique. The BANNER architecture is illustrated in Figure 3. A BANNER input file contains a text which has been separated into sentences. Each sentence is taken individually and is tokenized. The tokenization process in BANNER breaks tokens into either a contiguous block of

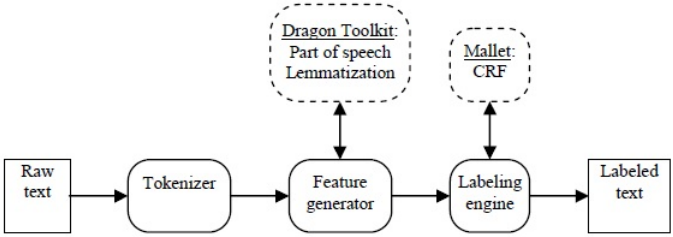


Figure 3: BANNER Architecture (Leaman and Gonzalez, 2008)

letters and/or digits or a single punctuation mark. As an example, the string "Bub2p-dependent" is broken into 3 tokens: "Bub2p", "-", and "dependent". In the next step features are assigned to each individual token. Each feature is a name/value pair for use by the machine learning algorithm. And finally in the labeling process, each feature gets exactly one label. BANNER makes use of the Mallet CRF (McCallum, 2002) in both feature generation and labeling. The set of machine learning features used in BANNNER is listed in Table 1.

Feature set definition	Description
The part of speech which the token plays in the sentence	Provided by the Dragon toolkit implementation of the Hepple tagger.
The lemma for the word represented by the token, if any	Provided by the Dragon toolkit.
A set of regular expression features	Includes variations on capitalization and letter/digit combinations.
2, 3 and 4-character prefixes and suffixes	
2 and 3 character n-grams	Including start-of-token and end-of-token Indicators
Word class Convert	upper-case letters to "A", lowercase letters to "a", digits to "0" and other characters to "x"
Numeric normalization	Convert digits to "0"
Roman numerals	
The names of the Greek letters	

Table 1: Set of features in BANNER (Leaman and Gonzalez, 2008)

BANNER considers a token window of 2 to make features, meaning that the features of each token contains the features of the two previous and the two following tokens.

BANNER has been used for NER in the Gene names and the disease names domains, and it has achieved results comparable with other NER systems in these domains.

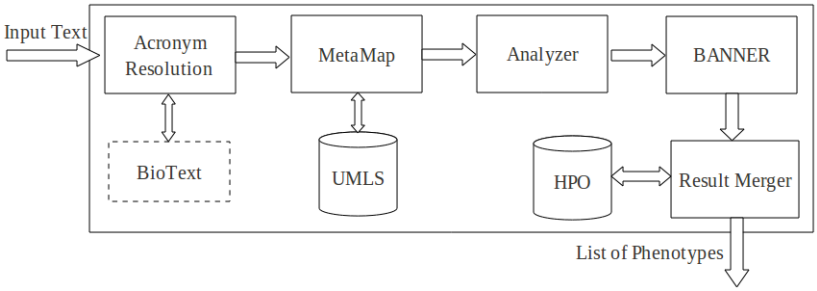


Figure 4: System Block Diagram

3 Proposed Method

The rule-based system described in Section 2.3 achieves good results. However, it, like other rule-based methods, has some shortcomings. As the rules are extracted from a small corpus, they may be overfitted to that corpus and cannot cover new phenotypes in other texts. And if we wish to use it on a larger corpus, we may need to add extra rules. It is not easy to analyse the errors manually to generate the new rules. So, we decided to extend the capabilities of our rule-based system using machine learning methods.

BANNER is open-source and it provides a good infrastructure for NER. Its results are convincing and features can be easily added. Therefore starting with BANNER and its CRF method, the rule-based method was incorporated to it.

The block diagram of the system is illustrated in Figure 4. The first phase in this system is finding acronyms in the input text and resolving them. Usually, papers indicate the local unambiguous reference for each acronym used at its first usage. So acronym resolution is done, making a list of local full forms for acronyms (according to rule 1 of the rule-based method). Then the output which does not contain any acronym is processed by MetaMap. According to our settings, MetaMap finds composite noun phrases with up to 3 simple phrases and prints out the syntax of each noun phrase. Each noun phrase is also tagged with a semantic type. The Analyzer analyses the MetaMap output and changes it to BANNER input format. Our feature-enhanced BANNER takes each sentence separately and using the features, finds some phenotypes. In the last step the system searches for the HPO phenotypes in the text and found phenotypes are added to our list of phenotypes. The details of how we made use of rules and features in our machine learning method are explained in the following sections.

3.1 Incorporating the rules

Rule 1 *Acronym Resolution* implements Rule 1. It finds the full forms of acronyms using their local unambiguous reference in the text and replaces acronyms with their unabbreviated forms. BioText (Schwartz and Hearst, 2003) has been used to make a list of acronyms and their full forms in the text.

Rule 2 This rule is implemented in *Analyzer*. Analyzer finds the noun phrases and their heads from the MetaMap output. If Metamap breaks a noun phrase into different parts and

assigns different semantic types to them, the Analyzer assigns the semantic type of the head to the whole phrase. An example of such a phrase and the Analyzer output is shown in Figure 5. The phrase "of Diamond-Blackfan anemia patients" is broken into two noun phrases "Diamond-Blackfan anemia" and "patients", each with a different semantic type. MetaMap output shows that the head of the whole phrase is "patients" so analyzer assigns its semantic type, i.e. *Patient or Disabled Group*, to the whole phrase.

Rule 3 To help our machine learning method learn this rule, three binary features were added to the system: *Special Modifier*, *Anatomy* and *Physiology* to indicate whether a noun phrase is in the Special Modifier, Anatomy or Physiology classes. Furthermore, some sentences containing this class of phenotypes (*Special Modifier* + [Anatomy] Or [Physiology]) were added to our training set.

Rule 4 When the Result Merger searches for HPO phenotypes in the text, it searches for their singular and plural forms.

Rule 5 This rule is considered in Result Merger: if the head of a noun phrase is found in HPO the whole noun phrase is tagged as a phenotype.

3.2 Adding features

To implement the rule-based system completely, Rule 3 was added as three features to the CRF, in addition to the features already possessed by BANNER. Finally, some other features which seemed to be helpful were added to the system. These features were tested several times and finally the best set of features were selected. These features include:

1. Phenotype: This feature comes from the rule-based method (see Section 2.3). In this method semantic types in SG *Disorders* are categorized into two categories. This feature is a binary feature that indicates whether a noun phrase is in the *Phenotypes* category.
2. Phenotype Candidates: This feature is a binary feature that indicates whether a noun phrase is in the *Phenotype Candidates* category of SG *Disorders*.
3. Special Modifier: A binary feature that indicates whether a noun phrase is in the list of special modifiers (Burgun et al., 2009).
4. Anatomy: A binary feature which means a noun phrase is in SG *Anatomy* or not.
5. Physiology: A binary feature which means a noun phrase is in SG *Physiology* or not.
6. List Separator: We found out that usually in biomedical literature when a number of elements in a list are phenotypes, there is a good chance that the other elements are phenotypes too. The List Separator feature was added to designate the availability of list indicators (*and* or *comma*) in the sentence.
7. Semantic type: The semantic type which is assigned by MetaMap to noun phrases. This feature is null for other phrases.
8. NP: A binary feature which indicates whether a token is a part of a noun phrase.
9. POS: The part of speech of a token. This feature is available in BANNER.
10. Lemma: The lemma of each token. This is available in BANNER.
11. NPstart: A binary feature to indicate whether a token is the first token in a noun phrase.
12. NPend: A binary feature to designate whether a token is the last token in a noun phrase.
13. 2, 3 and 4-character prefixes and suffixes: This is a part of BANNER.
14. 2 and 3 character n-grams: This is a part of BANNER.

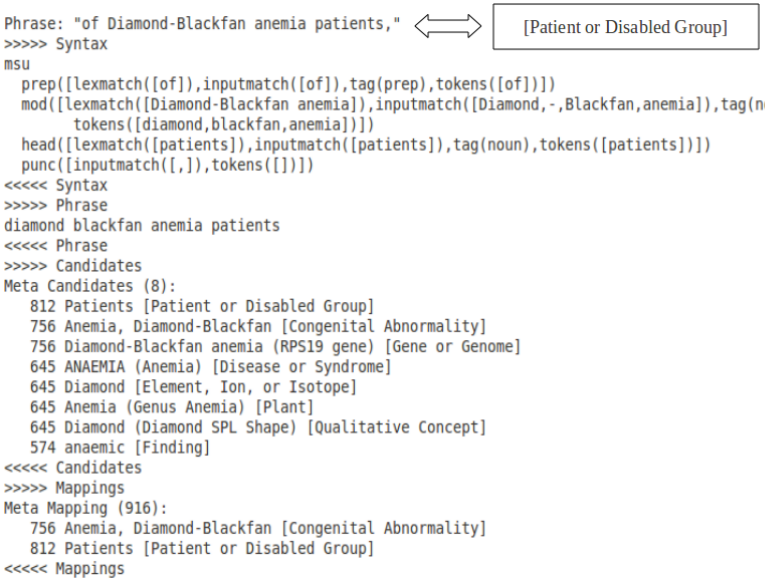


Figure 5: Analyzer Example

Also, it should be remembered that BANNER makes a window of size 2 for each token, i.e. features of each token contain features of the 2 tokens before and the 2 tokens after it. We used the default configuration of BANNER. The NP feature comes from the MetaMap results and the POS feature is provided by BANNER. They do not align perfectly, but it is the task of the CRF to find a solution using these conflicting features.

3.3 Corpus

The most significant problem for us in using the machine learning method was the lack of an annotated specialized corpus of sufficient size. As no one before has used machine learning on phenotype name recognition, no corpus was available for us to train and test our system with. Therefore we had to make our own corpus with a sufficient number of sentences. However, making a large corpus is a very difficult and time consuming task. So we decided to use semi-supervised learning to make our corpus.

3.3.1 Collecting the papers

To find the papers which are relevant to phenotypes, we started with two available databases: PubMed (2009) and BioMedCentral (2004). All HPO phenotypes were searched for in these databases and every paper which contained at least three different phenotypes was added to our collection. In this way we found 100 papers which were used to train the system. We had another 13 papers which had been collected for the development of the rule-based method (see Section 2.3) and were annotated manually. These 13 papers were used to test the system.

3.3.2 Annotating the corpus

As it was not possible for us to annotate the 100 papers manually, we used a semi-supervised learning method starting with the information provided by HPO. First, HPO phenotypes were searched for in the set of papers and were tagged as phenotypes. These papers along with their tags made our initial training corpus. When annotating the corpus, it should be noted that the last phase of the system (the Result Merger) was omitted because all HPO-annotated phenotypes were already annotated.

The trained model was used to annotate the training set again. The newly found phenotypes were analysed manually and the correct ones were added as annotations to the training set. Also, on each iteration, the system was tested using the test set to find out how many iterations would be sufficient for training the system. This process was repeated several times until we reached the results that we were satisfied with when testing the last model on the test set.

One important point to mention is that we only included positive sentences (sentences with at least one phenotype) in our training set, because the number of negative sentences was far greater than positive sentences and it prevented the system from training efficiently. Therefore, whenever new phenotype names were found, the number of sentences in the training set increased for the next iteration.

4 Evaluation

We compared the performance of our system against the rule-based system (Khordad et al., 2011), which is the only specialized system for phenotype name recognition that we are aware of. The final training corpus which is made from 100 papers contains 2755 sentences and 4233 annotated phenotypes. All sentences in this corpus include at least one phenotype. A test set is collected from 13 papers and includes 216 sentences and 373 phenotypes.

To evaluate the system, 10-fold cross validation has been used. Also the system has been tested using a separate test set. Table 2 gives the details of the results. The base system is our machine learning method ignoring the Result Merger. Result Merger finds HPO phenotypes in text and adds them to the list of phenotypes. The results after using Result Merger are mentioned in the column labelled *HPO added*. The rule-based system has been tested using the test set and the results are displayed in Table 3. To calculate these results, a returned phrase is considered to be a true phenotype if its head contains a phenotype. For example, in the phrase “acute myloid leukemia” the head is “leukemia”, a phenotype that is confirmed by its inclusion in HPO. However, in the phrase“Diamond-Blackfan anemia patients” the correct phenotype is: “Diamond-Blackfan anemia”. If the system returns “Diamond-Blackfan anemia patients” as the phenotype, it is deemed false.

As this table demonstrates, the results are comparable with other named entity recognition systems which are specialized for finding other biomedical entities even though they may have larger training corpora. For example BANNER has been trained and tested for finding gene names, using BioCreative 2 GM corpus containing 15,000 sentences (7500 for training and 7500 for testing) which is much larger than our current corpus. However our results are really better than Banner’s (Precision 85.09, Recall 79.06 and F-measure 81.96)(Leaman and Gonzalez, 2008). Although our task is different these results mean that our system is performing well.

In addition the machine learning method outperformed the rule-based method, even though the corpus is not fully annotated. We believe that if we had a fully annotated corpus the system would achieve an even better performance.

	Base system			HPO Added		
	Precision	Recall	F-measure	Precision	Recall	F-measure
10-Fold	82.83	68.13	74.35	86.89	98.33	92.25
Test Set	93.44	57.37	71.09	95.76	90.88	93.25

Table 2: System Evaluation Results

	Precision	Recall	F-measure
Machine Learning Method	95.76	90.88	93.25
Rule-Based Method	88.34	73.19	80.05

Table 3: Comparing the system with the Rule-Based Method

To have an idea of how well our annotation process works, we selected 100 random sentences from our corpus. Then, we tagged these sentences manually. There were 157 phenotypes in these 100 sentences. Our semi-supervised machine learning method found 142 phenotypes and 1 of its phenotypes was incorrect, i.e. it missed 16 phenotypes. So, the annotation process misses about 10 percent of the phenotypes.

5 Discussion

Finding the best set of features is one of the most important parts of developing the system. Tables 4 and 5 show the role and importance of each feature. To illustrate the contribution of each feature in Table 4 we considered a small set of features as the basic set of features for our system. These features came from the rule-based system and are very important in signifying phenotype names in the text. Then, in each line a feature is added to the basic feature set until we have the complete feature set in the last line. Adding some features (Phenotype candidates, List separators, Semantic Types, and Lemma) drops the results slightly but the results of the last feature set is better than the previous feature sets.

Analysing Table 4, it may seem that including some features is not necessary. Table 5 illustrates the role of each feature in a different way. In each line of Table 5 only one feature is ignored from the feature set and the system is tested using the separate test set. Note that the results are calculated without adding the HPO. As one can see, removing each feature reduces the results slightly. The exception to this modest reduction in performance is the removal of the NP feature which causes a significant drop in precision and recall, because not having NP information causes errors in finding NP end and NP start. In some cases (Lemma, Phenotype candidates, NP start and NP end, and Physiology) ignoring the feature causes small improvement in precision or recall. However the F-score is always less than the F-score of final results.

Reviewing the results, it has been found that both the rule-based and machine learning methods are dependent on MetaMap. MetaMap does make mistakes. MetaMap makes some errors in finding the boundaries of NPs and in determining the semantic types. NP boundaries and semantic types are features used by both methodologies, so the errors made by MetaMap have effects on the performance of each system. However the rule-based system is more dependent on MetaMap output and errors in MetaMap output changes the results completely. But the machine learning system is more robust and it sometimes finds the correct phenotype names despite Metamap errors. For example consider the sentence *"Diamond-Blackfan anemia is a rare inherited bone marrow failure syndrome."*. The phrase *Diamond-Blackfan anemia* is a phenotype but MetaMap assigns the [Gene or Genome] semantic type to it, which is not in the SG *Disorders*.

Features	Precision	Recall	F-Score
Phenotype, Anatomy, Physiology, Special Modifier	89.89	47.72	62.34
+Phenotype candidates	90.72	47.18	62.07
+List Separator	90.41	40.48	55.92
+Semantic type	90.24	49.59	64
+NP	90.24	49.59	64
+POS	91.15	55.22	68.77
+Lemma	92.72	54.69	68.79
+NP start, NP end	93.44	57.37	71.09

Table 4: Contribution of each additional feature

Ignored Feature	Precision	Recall	F-Score
Anatomy	92.82	55.49	69.45
Lemma	93.57	54.69	69.03
List Separator	91.66	56.03	69.54
Phenotype	92.54	56.56	70.20
NP	81.81	38.6	52.45
Phenotype Candidate	92.64	57.37	70.85
NP start and NP end	93.18	54.95	69.13
Physiology	93.21	55.22	69.35
POS	92.05	52.81	67.11
Semantic Type	91.89	54.69	68.56
Special Modifier	92.44	55.76	69.56

Table 5: Contribution of each feature

So the rule-based system does not tag it as a phenotype. The phrase *missing vertebrae* is another example of MetaMap errors. MetaMap does not consider this phenotype name as one NP. It separates this phenotype name into two phrases *missing* and *vertebrae*.

In addition determining the boundary of an NP is very important in the rule-based system. MetaMap has an option to make larger NPs by merging simpler NPs. If we only use simple NPs, we cannot get larger phenotype names as one NP and the system will miss them. The phrase *Partial hypoplasia of the corpus callosum* is an example of a phenotype name with composite NPs. On the other hand if we use composite NPs, the head of the NP may change and the semantic type of the NP may change as a result. This is problematic for the rule-based system. The phrase *the associations of facial dysmorphism* is an example. The word "associations" is the head of this phrase, so the semantic type [Mental Process] is assigned to it and it is not tagged as a phenotype name by the rule-based system.

On the other hand, there are some cases in which MetaMap assigns the correct semantic type to a phrase and found a good boundary for a phenotype name but the machine learning method does not mark it as a phenotype. For example "arhinia" is not tagged as a phenotype by the machine learning method in the following sentence *"These phenotypes may resemble that of the only confirmed case of an individual with a lethal compound heterozygous PAX6 mutation and may include anophthalmia, arhinia and severe central nervous system defects"* although MetaMap assigns the semantic type [Congenital Abnormality] (which is in the category of Phenotypes) to it.

Table 6 shows how many false negatives and true positives are available in the final results for both the machine learning and the rule-based methods. And Table 7 illustrates the percent of these errors caused by MetaMap or the boundary of noun phrases.

	Rule-based	Machine Learning
True positives	273	339
False negatives	100	34

Table 6: number of TPs and FNs in each method

	Rule-based	Machine Learning
MetaMap errors	37%	11.76%
NP boundary errors	26%	8.82%

Table 7: Analysis of NPs

Comparing the precision errors gave interesting observations. There are no common errors between the two systems. False positive errors in the rule-based system were caused by the rules not being discriminating enough. For each returned phrase, one of the rules produced that phrase. But there are exceptions to each rule that can cause these false positive errors. The exceptions to the rules do not cause any problem for the machine learning system. The machine learning system was able to learn all of these exceptions. For the false phenotypes returned by the machine learning system, analysis indicates that none of these would be suggested by application of a rule in the rule-based system, explainging why there are no common errors.

Conclusion

In this paper we discussed the development of a named entity recognition system which is specialized to find phenotype names in biomedical literature. The system has been generated using machine learning. Some of its features are based on the rules found in our previous rule-based method (Khordad et al., 2011). We added some other features. The system makes use of MetaMap and HPO.

As there was no annotated corpus available for training the machine learning system, a corpus was annotated using a semi-supervised learning method and HPO. The corpus does not fully annotate all phenotype names. About 10 percent of the phenotype names are missed.

The system has been evaluated using both a 10-fold cross validation and a separate test set and the results are really promising.

The current system is extremely dependent on MetaMap output, although it is less dependent than the rule-based system. Still, when MetaMap gives erroneous output, it makes it difficult for our system to work correctly. In addition the corpus is not completely annotated and its annotation has some errors.

Despite these problems, the system achieved an F-Score of 92.25 and its performance is comparable to other NER systems which are specialized for finding other entities in biomedical literature.

To improve the system performance, it is imperative to overcome the phenotype name recognition errors initiated by MetaMap parsing errors. Using a more reliable partial parser to provide the NPs for MetaMap's mapping to UMLS concepts, instead of using the Metamap embedded

parser, may fix this problem. Furthermore, adding more features to the machine learning system can be considered. Finally, a machine learning method other than CRF might achieve better results.

Acknowledgments

This work was partially funded through a Natural Sciences and Engineering Research Council of Canada (NSERC) Discovery Grant to Robert E. Mercer.

References

Andrade, M. A. and Valencia, A. (1998). Automatic extraction of keywords from scientific text: application to the knowledge domain of protein families. *Bioinformatics*, 14(7):600–607.

Aronson, A. R. (2001). Effective mapping of biomedical text to the umls metathesaurus: the metamap program. In *AMIA Symposium*, pages 17–21.

Bui, Q.-C., Katrenko, S., and Sloot, P. M. A. (2011). A hybrid approach to extract protein-protein interactions. *Bioinformatics*, 27(2):259–265.

Burgun, A., Mougin, F., and Bodenreider, O. (2009). Two approaches to integrating phenotype and clinical information. *AMIA Annual Symposium proceedings*, 2009:75–79.

Coulet, A., Shah, N. H., Garten, Y., Musen, M. A., and Altman, R. B. (2010). Using text to build semantic networks for pharmacogenomics. *Journal of Biomedical Informatics*, 43(6):1009–1019.

Dai, M., Shah, N. H., Xuan, W., Musen, M. A., Watson, S. J., Athey, B. D., and Meng, F. (2008). An efficient solution for mapping free text to ontology terms. In *AMIA Summit on Translational Bioinformatics, San Francisco, CA*.

Day-Richter, J., Harris, M. A., Haendel, M., OBO, T. G. O., and Lewis, S. (2007). OBO-Edit an ontology editor for biologists. *Bioinformatics*, 23(16):2198–2200.

Friedman, C., Kra, P., Yu, H., Krauthammer, M., and Rzhetsky, A. (2001). GENIES: a natural-language processing system for the extraction of molecular pathways from journal articles. *Bioinformatics (Oxford, England)*, 17 Suppl 1(suppl_1):S74–82.

Fukuda, K., Tamura, A., Tsunoda, T., and Takagi, T. (1998). Toward information extraction: identifying protein names from biological papers. *Pac Symp Biocomput*, pages 707–718.

Fundel, K., Küffner, R., and Zimmer, R. (2007). Relex - relation extraction using dependency parse trees. *Bioinformatics*, 23(3):365–371.

Gaizauskas, R., Demetriou, G., Artymiuk, P. J., and Willett, P. (2003). Protein structures and information extraction from biological texts: the PASTA system. *Bioinformatics*, 19(1):135–143.

Gong, L.-J., Yuan, Y., Wei, Y.-B., and Sun, X. (2009). A hybrid approach for biomedical entity name recognition. In *Biomedical Engineering and Informatics, 2009. BMEI '09. 2nd International Conference on*, pages 1 –5.

He, X. and DiMarco, C. (2005). Using lexical chaining to rank protein-protein interactions in biomedical texts. In *BioLink 2005: Workshop on Linking Biological Literature, Ontologies and Databases: Mining Biological Semantics, Conference of the Association for Computational Linguistics (poster Presentation)*.

Horn, F., Lau, A. L., and Cohen, F. E. (2004). Automated extraction of mutation data from the literature: application of MuteXt to G protein-coupled receptors and nuclear hormone receptors. *Bioinformatics*, 20(4):557–568.

Hristovski, D., Friedman, C., Rindflesch, T. C., and Peterlin, B. (2006). Exploiting semantic relations for literature-based discovery. *AMIA Annual Symposium proceedings*, pages 349–353.

Hristovski, D., Peterlin, B., Mitchell, J. A., and Humphrey, S. M. (2005). Using literature-based discovery to identify disease candidate genes. *I. J. Medical Informatics*, 74(2-4):289–298.

Humphreys, B. L., Lindberg, D. A., Schoolman, H. M., and Barnett, G. O. (1998). The Unified Medical Language System: an informatics research collaboration. *J Am Med Inform Assoc*, 5(1):1–11.

Humphreys, K., Demetriou, G., and Gaizauskas, R. (2000). Two applications of information extraction to biological science journal articles: enzyme interactions and protein structures. *Pacific Symposium on Biocomputing. Pacific Symposium on Biocomputing*, pages 505–516.

Khordad, M., Mercer, R. E., and Rogan, P. (2011). Improving phenotype name recognition. *Canadian Conference on AI*, 6657:246–257.

kiong Ng, S. and Wong, M. (1999). Toward routine automatic pathway discovery from on-line scientific text abstracts. *Genome Informatics*, 10:104–112.

Klein, T. E., Chang, J. T., Cho, M. K., Easton, K. L., Fergerson, R., Hewett, M., Lin, Z., Liu, Y., Liu, S., Oliver, D. E., Rubin, D. L., Shafa, F., Stuart, J. M., and Altman, R. B. (2001). Integrating genotype and phenotype information: an overview of the PharmGKB project. *The pharmacogenomics journal*, 1(3):167–170.

Krauthammer, M., Rzhetsky, A., Morozov, P., and Friedman, C. (2000). Using BLAST for identifying gene and protein names in journal articles. *Gene*, 259(1-2):245–252.

Leaman, R. and Gonzalez, G. (2008). Banner: An executable survey of advances in biomedical named entity recognition. In *Pacific Symposium on Biocomputing*, pages 652–663.

Leroy, G., Chen, H., and Martinez, J. D. (2003). A shallow parser based on closed-class words to capture relations in biomedical text. *Journal of Biomedical Informatics*, 36(3):145–158.

Leser, U. and Hakenberg, J. (2005). What makes a gene name? named entity recognition in the biomedical literature. *Briefings in Bioinformatics*, 6(4):357–369.

McCallum, A. K. (2002). Mallet: A machine learning for language toolkit. http://mallet.cs.umass.edu.

McCray, A., Burgun, A., and Bodenreider, O. (2001). Aggregating UMLS semantic types for reducing conceptual complexity. *Proceedings of Medinfo*, 10(pt 1):216–20.

McKusick, V. (2007). Mendelian Inheritance in Man and Its Online Version, OMIM. *The American Journal of Human Genetics*, 80(4):588–604.

Nobata, C., Collier, N., and ichi Tsujii, J. (1999). Automatic term identification and classification in biology texts. In *In Proc. of the 5th NLPRS*, pages 369–374.

Rindflesch, T. C., Libbus, B., Hristovski, D., Aronson, A. R., and Kilicoglu, H. (2003). Semantic relations asserting the etiology of genetic diseases. *AMIA Annual Symposium Proceedings*, pages 554–558.

Rindflesch, T. C., Tanabe, L., Weinstein, J. N., and Hunter, L. (2000). Edgar: Extraction of drugs, genes and relations from the biomedical literature. In *Pacific Symposium on Biocomputing*, volume 5, pages 514–525.

Robinson, P. N. and Mundlos, S. (2010). The human phenotype ontology. *Clinical genetics*, 77(6):525–534.

Rocktäschel, T., Weidlich, M., and Leser, U. (2012). Chemspot: a hybrid system for chemical named entity recognition. *Bioinformatics*, 28(12):1633–1640.

Schwartz, A. S. and Hearst, M. A. (2003). A simple algorithm for identifying abbreviation definitions in biomedical text. *Pacific Symposium on Biocomputing. Pacific Symposium on Biocomputing*, pages 451–462.

Segura-Bedmar, I., Martinez, P., and Segura-Bedmar, M. (2008). Drug name recognition and classification in biomedical texts: A case study outlining approaches underpinning automated systems. *Drug Discovery Today*, 13(17-18):816 – 823.

Seki, K. and Mostafa, J. (2005). A hybrid approach to protein name identification in biomedical texts. *Inf. Process. Manage.*, 41(4):723–743.

Swanson, D. (1986). Fish oil, raynaud's syndrome, and undiscovered public knowledge. *Perspect. Bio. Med*, 30:7–18.

Tanabe, L., Scherf, U., Smith, L. H., Lee, J. K., Hunter, L., and Weinstein, J. N. (1999). MedMiner: an Internet text-mining tool for biomedical information, with application to gene expression profiling. *BioTechniques*, 27(6):1210–1217.

Valencia, A. (2005). Automatic annotation of protein function. *Current opinion in structural biology*, 15(3):267–274.

Xu, R., Supekar, K., Morgan, A., Das, A., and Garber, A. (2008). Unsupervised method for automatic construction of a disease dictionary from a large free text collection. *AMIA Annual Symposium proceedings*, pages 820–824.

Yu, H., Zhu, X., Huang, M., and Li, M. (2005). Discovering patterns to extract protein-protein interactions from the literature: Part ii. *Bioinformatics*, 21(15):3294–3300.

Improving Combinatory Categorial Grammar Parse Reranking with Dependency Grammar Features

Sunghwan Mac Kim[1] *Dominick Ng*[2]
Mark Johnson[1] *James R. Curran*[2]
(1) Department of Computing, Macquarie University, Sydney, NSW, Australia 2109
(2) School of Information Technologies, University of Sydney, Sydney, NSW, Australia 2006
sunghwan.kim@students.mq.edu.au, dominick.ng@sydney.edu.au,
mark.johnson@mq.edu.au, james.r.curran@sydney.edu.au

ABSTRACT
This paper presents a novel method of improving Combinatory Categorial Grammar (CCG) parsing using features generated from Dependency Grammar (DG) parses and combined using reranking. Different grammar formalisms have different strengths and different parsing models have consequently divergent views of the data. More specifically, dependency parsers are sensitive to linguistic generalisations that differ from the generalisations that the CCG parser is sensitive to, and which the reranker exploits to identify the parse most likely to be correct. We propose DG-derived reranking features, which are obtained by comparing dependencies from the CCG parser with DG dependencies, and demonstrate how they improve the performance of a CCG parser and reranker in a variety of settings. We record a final labeled F-score of 87.93% on section 23 of CCGbank, 0.5% and 0.35% improvements over the base parser (87.43%) and reranker (87.58%), respectively.

KEYWORDS: Combinatory Categorial Grammar (CCG), Dependency Grammar (DG), Reranking, Dependency Grammar-derived features, parsing, syntax.

Proceedings of COLING 2012: Technical Papers, pages 1441–1458,
COLING 2012, Mumbai, December 2012.

1 Introduction

Reranking is the process of rescoring an *n*-best list with an external model, and it is an effective method for improving performance in NLP tasks. In parsing, rerankers are able to incorporate arbitrary global features from the entire parse tree that would be intractable in a base parser. More informative features can be considered in reranking as the entire parse tree is available, as opposed to the fragments considered in parsing.

In this paper, we propose a simple method for improving the performance of the C&C Combinatory Categorial Grammar (CCG) parser (Clark and Curran, 2007). We parse sentences using the C&C *n*-best parser and a 1-best dependency grammar (DG) parser, and generate DG-derived features by comparing the extracted dependencies from the C&C parser with the DG dependencies. We then incorporate the DG-derived features into the CCG reranker of Ng et al. (2010) to reorder the *n*-best CCG parses using the external parse information. We experiment with both the Maltparser (Nivre et al., 2007b) and the MSTparser (McDonald et al., 2005) as the DG parser. This is the first cross-formalism parser combination experiment for CCG parsing that we are aware of, combining the features and strengths of two different formalisms together.

Previous work has shown that dependency parsers such as the Maltparser perform better on short-range dependencies (McDonald and Nivre, 2007), whereas the C&C parser deals with long-range dependencies more reliably (Clark et al., 2002; Rimell et al., 2009). Short-range dependency information has also been shown to improve parser accuracy (Chen et al., 2009). We show how our new DG-derived features substantially improve parser performance by 0.35% to 87.93%, and improve the accuracy of the C&C parser on both short and long-range dependencies. These results demonstrate how rerankers can successfully combine diverse features from different formalisms for better parsing accuracy.

2 Background

2.1 Reranking and Cross-formalism Parser Combination

Collins (2000) describes reranking for the Collins (Model 2) parser and defined the general approach that has been used for the task since. Reranker training data is produced by parsing 36,000 sentences from sections 02-21 of the Penn Treebank WSJ data (Marcus et al., 1993) using an *n*-best version of the base parser. The parser model used for this process is trained using cross-validation to ensure that overly optimistic parses are not produced. Global features calculated over the whole tree such as context-free rules, *n*-gram ancestors, parent and grandparent relationships, and lexical heads and the distances between them are extracted from the parses and fed into a boosting-based reranker. Collins reports a final PARSEVAL F-score of 89.75%, a 1.55% improvement compared to the baseline parser.

The oracle F-score (given a perfect reranker that always chooses the best *n*-best parse for a sentence) is used to measure the quality of *n*-best parses. Huang and Chiang (2005) describe efficient *n*-best parsing algorithms that have become widely used in the field, including in the Charniak and Johnson (2005) reranker. This system uses a similar setup to the Collins reranker, but adopts a maximum-entropy model along with additional features, including features for subject-verb agreement, *n*-gram local trees, and right-branching factors. In 50-best mode the parser has an oracle F-score of 96.80%, and the reranker produces a final F-score of 91.40% compared to an 89.70% baseline.

Farkas et al. (2011) rerank BitPar, an unlexicalised generative PCFG parser for German (Schmid,

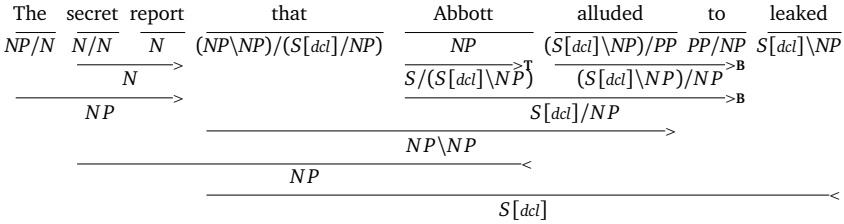


Figure 1: An example CCG derivation using application, composition, type-raising, and unary rules. A long-range dependency is created between *report* and *to*, mediated by *that*.

2004) using dependency grammar features and forest-based rerankers. Bohnet (2010), which is a second order dependency parser, was used to generate parses for feature extraction. Their experimental results show a 0.8% F-score improvement. However, their work was only concerned with extracting additional features from the dependency parses, and does not generate features based on a comparison between the extracted dependency parses and the constituency parses that are being reranked.

Øvrelid et al. (2009) describe a two-stage system where the output of an LFG parser is used to provide features for the Maltparser in English and German. They observe a 0.15% improvement in Maltparser unlabeled attachment scores for English, and 1.81% improvement in German. This work exploits analyses from different formalisms, but it completely retrains the Maltparser with additional features based on a conversion of the LFG analyses to a dependency representation. It also targets improved dependency parsing rather than improved grammar-driven parsing. Sagae et al. (2007) used the output of dependency parser to disambiguate Head-driven Phrase Structure Grammar (HPSG) parses. This work used a single penalty parameter for each mismatch between the DG and HPSG parses that was set via a parameter sweep on held-out data.

2.2 Combinatory Categorial Grammar

Combinatory Categorial Grammar (CCG; Steedman, 2000) is a lexicalised grammar formalism based on combinatory logic. Lexical categories govern the syntactic behaviour of each word, and generic combinatory rules combine categories together to form an analysis of a sentence.

Atomic categories such as noun phrases (*NP*) and sentences (*S*) represent syntactically complete constituents. Complex categories are binary functors of the form A/B or $A\backslash B$, and subcategorize for an argument category B to the right or left respectively to form an A. For example, transitive verbs ($(S\backslash NP)/NP$) subcategorize for an object *NP* to the right to form a verb phrase $S\backslash NP$, which in turn expects a subject *NP* to the left to form a sentence *S*.

The simplest combinatory rules are forward and backward application, where complex categories acquire their outermost argument and return their result. Additional combinators are based on composition and unary category type-changing, increasing the generative power of the formalism and enabling the analysis of phenomena such as *wh*-movement and right-node raising. Figure 1 gives an example CCG derivation using these combinators.

We will use the CCG dependency representation of CCGbank (Hockenmaier and Steedman,

2007) in this work. Each dependency expresses a word-word relationship between a head and a dependent, generated when the assigned categories are combined. Additionally, CCG allows for the production of long-range dependencies mediated by intermediate words in the sentence. This allows a clear representation of function and trace information that would require co-indexation in phrase-structure parses. These dependencies have the following form: $\langle to, PP/NP_1, 1, report, (NP\backslash NP)/(S[dcl]/NP)\rangle$, which includes the head word, its category, the argument slot, argument word, and the mediating category for long-range dependencies.

2.3 CCG parsing

The C&C parser is a fast and accurate wide-coverage CCG parser. It is a two-stage system, where a supertagger assigns probable categories to words in a sentence and the parser combines them using the CKY algorithm. The parser has been found to be particularly accurate at recovering long-range dependencies (Clark et al., 2002; Rimell et al., 2009).

C&C is trained on CCGbank, a conversion of the Penn Treebank WSJ data to CCG derivations and dependencies (Hockenmaier and Steedman, 2007). We use the normal-form model described in Clark and Curran (2007), which models the probability of derivations. We also follow the convention of using section 00 of CCGbank as development data, sections 02-21 as training data, and section 23 for final testing. The standard evaluation metric is labeled dependency recovery, as described by Clark and Hockenmaier (2002).

Clark and Curran (2007) develop a conversion from CCG dependencies to Briscoe and Carroll-style grammatical relations (GRS) (King et al., 2003; Briscoe and Carroll, 2006). GRS provide a useful abstraction as they allow the conflation of many CCG dependencies that are semantically similar but structurally different. For example, since subcategorization information is fully specified in categories, the verb-subject relationship is expressed in many different forms in CCG depending on the transitivity of the verb. In the GR scheme, they map to a general `ncsubj` dependency, echoing the underlying similarity between the CCG dependencies.

Rimell and Clark (2009) adapt the C&C parsing for the biomedical domain, and in the process they developed a mapping from CCG dependencies to Stanford dependencies based on the GR conversion. We generate DG-derived features based on this Stanford dependency output of the C&C parser to maximise the potential overlaps between the representations; this differs from the existing C&C reranking features, which use the CCG dependency format (Ng et al., 2010).

Figure 2 shows the CCG derivation and corresponding Stanford dependencies for the example sentence, *We are about to see if advertising works*, taken from WSJ section 22.

2.4 Dependency parsing

Dependency Grammars (DG) describe the syntactic structure of a sentence in terms of head-dependent relations between words. The set of dependency relations for a sentence forms a dependency tree with a special root head word. Unlike CCG, DG directly model relationships between pairs of words, and do not easily account for mediated long-range dependencies. CCG categories encoded detailed subcategorization information that is not present in DG labels, and the restrictions in combinator application constrain the way CCG derivations can be built, whereas dependency arcs may appear between any pair of words under a DG. Thus, we expect that CCG and DG analyses will provide markedly different insights, despite both producing dependency-style output.

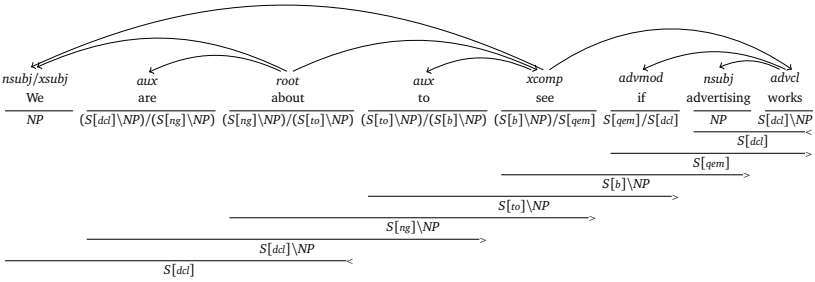


Figure 2: A CCG derivation and the Stanford dependencies produced by applying the Rimell and Clark (2009) conversion on the parse.

2.5 DG Representation Schemes

We experimented with four different dependency schemes; for each scheme, we retrained the Maltparser and the MSTparser over the extracted dependencies from the Penn Treebank WSJ data. 20-fold cross-validation was used to generate the parses corresponding to the reranker training data (sections 02-21); sections 02-21 was used to create a model for use at test time.

CoNLL: The CoNLL DG was used in the CoNLL 2007 dependency parsing shared task (Nivre et al., 2007a). Penn2Malt, a publicly available conversion utility[1], was used to generate CoNLL dependencies for our experiments. In contrast to other grammars used in this paper, this dependency scheme contains only unlabeled word-word arcs.

Stanford: de Marneffe and Manning (2008) introduced the dependency scheme used in the Stanford parser[2]. We used the Stanford parser's built-in converter to transform Penn Treebank trees into dependencies. The Stanford scheme has different variants; for this work we use the basic projective tree schema.

LTH: The LTH dependency scheme was developed with the aim of making better use of the linguistic information present in the Penn Treebank from version II onwards (Johansson and Nugues, 2007). We generated these dependencies using the LTH converter[3] over the NP-bracketed version of the Penn Treebank described by Vadas and Curran (2007). The converter was configured to produce a functional rather than lexical DG.

Fanse: Another conversion of the Penn Treebank with more fine-grained labels was presented in Tratz and Hovy (2011). The Fanse scheme is linguistically rich, featuring both non-projective dependencies and shallow semantic interpretation in its analyses. We used the freely available converter[4], which also requires the Vadas and Curran (2007) NP-bracketed Penn Treebank.

Figures 3 and 4 demonstrate some of the differences between the four dependency schemes. For instance, auxiliaries take the lexical verb as a dependent in all schemes except for Stanford,

[1] http://w3.msi.vxu.se/~nivre/research/Penn2Malt.html
[2] http://nlp.stanford.edu/software/lex-parser.shtml
[3] http://nlp.cs.lth.se/software/treebank_converter/
[4] http://www.isi.edu/publications/licensed-sw/fanseparser/

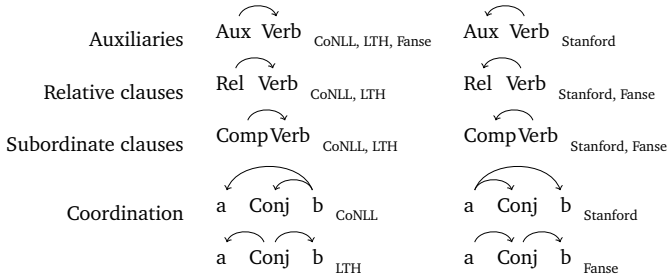


Figure 3: Analyses of auxiliaries, relative/subordinate clauses and coordination in the DG schemes.

where the lexical verb is the head of a *VP*. The characteristics of each scheme mean that each one produces an analysis that is quite different to the others as well as to CCG; by investigating a variety of schemes we hope to identify characteristics which are useful in our cross-formalism experiment.

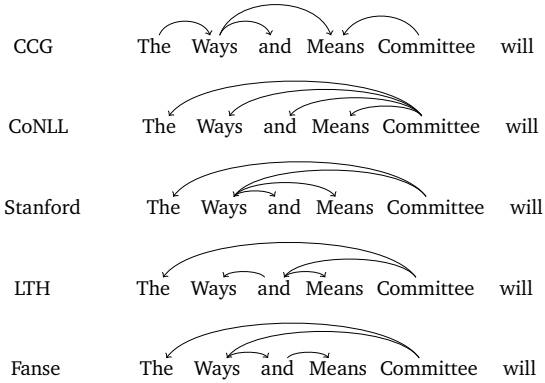


Figure 4: Example of divergence on the interpretation of the coordination by each scheme.

2.6 Maltparser and MSTparser

DG schemes	Maltparser	MSTparser
CoNLL	**90.46**	88.77
Stanford	89.82	87.27
LTH	84.54	86.67
Fanse	89.96	**89.61**

Table 1: Unlabeled Attachment Scores for each scheme over WSJ section 22.

In this work we use the Maltparser, a transition-based dependency parser (Nivre et al., 2007b),

and the MSTparser, a graph-based dependency parser (McDonald et al., 2005). The Maltparser uses an incremental shift-reduce algorithm, with actions guided by a classifier trained over parse history information. In contrast, the MSTparser builds a weighted graph for sentences, and finds the parse corresponding to the maximum spanning tree of the graph.

Table 1 shows the Unlabeled Attachment Scores (UASs) over WSJ section 22 for the Maltparser and the MSTparser with respect to four DG schemes[5]. The Maltparser has the highest UAS (90.46%) with the CoNLL DG, while the MSTparser performs best (89.61%) using the Fanse DG. Both parsers perform the worst with the LTH DG and by a substantial margin: 6% F-score for the Maltparser and 3% for the MSTparser compared with the best result. These results lead us to expect that features generated from Maltparser output will perform better than those from the MSTparser, LTH scheme notwithstanding.

3 CCG reranking

We follow the CCG reranker implementation described in Ng et al. (2010) and use the *n*-best C&C parser described in Brennan (2008); Ng and Curran (2012). This reranker is inspired by Charniak and Johnson (2005), with many new features designed to address specifics of the CCG formalism and evaluation process. For each *n*-best parse, the reranker uses a regression model to predict its expected F-score, and chooses the model with the highest predicted score. The log score and rank assigned to each derivation by the parser were encoded as core features in the reranker, and here we briefly summarise the other feature groups:

Tree Topology features describe the overall shape of the parse tree, to capture the fact that English generally favours right-branching parse trees, with heavy constituents generally occurring in the sentence-final position.

Local Context features represent fragments of the tree as well as layers of vertical and horizontal context that are difficult to encode in the parser model.

Argument-Adjunct features represent different attachment points for arguments and adjuncts and their wider context. Incorrect argument-adjunct distinctions can cause multiple CCG dependency errors due to the subcategorization information encoded in CCG categories.

Grammar-based features encode combinator sequences or combinations that may indicate an overly complicated or undesirable derivation. Additionally, these features encode the actual dependencies as these are the target of the evaluation.

C&C features from the parser are also incorporated as described in Clark and Curran (2007). These features encode combinations of word-category, word-POS, root-word, CCG rule, distance, and dependency information.

4 DG-derived Features

This section describes the DG-derived features. For convenience, we refer to the converted Stanford dependency output of an *n*-best CCG parse as the CCG *Dependency Parse* (CDP), as opposed to the *Malt/MST Dependency Parse* (MDP).

Our DG features are designed to capture the desirable and undesirable characteristics of the CDP and MDP, as well as the ways in which dependencies match and mismatch between the two. The dependencies from each *n*-best CDP are compared pairwise with the dependencies from the

[5] 20-fold cross-validation on sections 02-21 is used to create reranker training data.

1-best MDP for the corresponding sentence. *Matching* dependencies are those where the head and dependent are the same in the CDP and MDP. These dependencies also have a directionality component: whether the match occurs in the *same* direction (head and dependent are the same in CDP and MDP), or in the *reverse* direction (head and dependent are in opposition in the CDP and MDP). *Mismatching* dependencies are classified as being a head-dependent pair existing only in the CDP, or only in the MDP.

Our intuition is that the reranker should learn to prefer or disprefer particular properties of matching and mismatching dependencies. For example, it may learn that a particular CCG dependency is usually expressed in the same or opposite direction in the dependency parse. It may learn that a particular dependency is always expected to be mismatching, or that particular heads tend to take the same pairs of dependents in both parses.

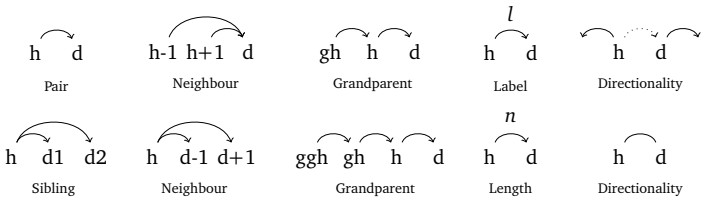


Figure 5: Dependency representations for each feature group. The letters ggh, gh, h, d refer to great grandparent head, grandparent head, head and dependent respectively.

For each matching dependency, binary indicator features were generated based on our feature templates (depicted in Figure 5). These features represent fragments of one or more dependency arcs that the reranker learns to favour or disprefer. Each feature includes components specified by the template, a directionality marker, and all four combinations of the word and POS tag for the head and dependent. Some of our templates also generate additional features for each mismatching dependency, conjoined with a label indicating whether the dependency existed only in the CDP or the MDP.

This approach differs from that of Sagae et al. (2007) since our reranker learns a separate penalty parameter for each combination of DG and CCG constructions as a feature of our regularised MaxEnt reranker model; these weights are learnt as part of the reranker training procedure. This enables our reranker to learn which DG constructions are most reliable and informative for CCG parsing, and which DG constructions should be ignored.

Following are descriptions of our DG-derived feature templates, which correspond to various dependency relations shown in Figure 5:

Pair-dependency features encode the head-dependent pair and a flag indicating a matching or mismatching dependency.

Sibling-dependency features encode matching sets of heads and pairs of dependents between the CDP and MDP. Multiple features are generated for each additional pair of matching dependents per head.

Neighbour-dependency features encode the linear context of heads and dependents in a sentence. For each matching dependency, the head is encoded with the word and POS tag in

turn of words immediately to the left and right of its dependent. This procedure is repeated for the dependent with the neighbours of its head.

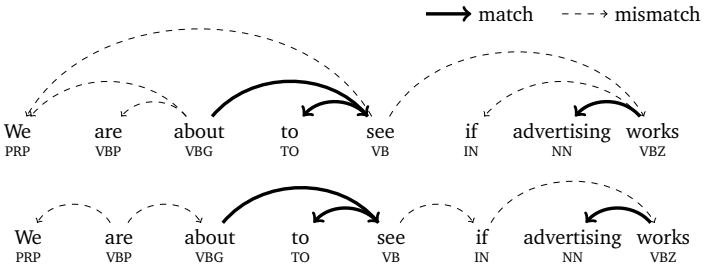
Grandparent-dependency features encode matching relationships of a great-grandparent head, grandparent head, head, and dependent between the CDP and MDP.

We also developed features that included further arc-level information from the dependency parses. These included:

Label-dependency features mimic the pair dependency features, but also include the label assigned by the dependency parser to the arc. This feature was not used for the CoNLL scheme (as this scheme does not include labels), but was active for each of the others.

Length-dependency features encode matching dependencies conjoined with the bucketed length of the dependency with respect to intervening tokens between head and dependent. The bucket intervals were set at 1, 2, 5, and 8 based on an analysis of typical lengths.

Directionality-dependency features consider, given a matching dependency, the additional matching dependencies that the head or dependent are part of. We term the dependencies in this set that are headed by the head or dependent as *out arcs*, and all others as *in arcs*. This feature template encodes the number of out arcs for each matching dependency, as well as a real value feature encoding the ratio of out arcs to in arcs.



Match: (about, see) (see, to) (works, advertising)
Mismatch^{CDP}: (about, are) (about, We) (see, We) (see, works) (works, if)
Mismatch^{MDP}: (are, We) (are, about) (see, if) (if, works)

Figure 6: Matching and mismatching pair dependencies between a CCG parse (top) and CoNLL parse (bottom) for the sentence *We are about to see if advertising works*.

Figure 6 illustrates matching and mismatching pair dependencies for the sentence shown in Figure 2, with dependencies involving punctuation ignored. There are three matching dependencies, five mismatching dependencies from to the CCG parse, and four mismatching dependencies from the DG parse. Table 2 lists examples of the pair-dependency features generated from these parses.

5 Results

We ran two classes of reranker experiments for each dependency scheme: one using gold-standard DG dependencies that were directly converted from the treebank data, and one using

Dependency	Features
(about, see)	`match:about:see, match:about:VB, match:VBG:see, match:VBG:VB`
(see, We)	`nomatch-ccg:see:We, nomatch-ccg:VB:We, nomatch-ccg:see:PRP, nomatch-ccg:VB:PRP`
(see, if)	`nomatch-malt:see:if, nomatch-malt:VB:if, nomatch-malt:see:IN, nomatch-malt:VB:IN`

Table 2: The generation process for pair-dependency features. Each feature template follows a similar pattern.

parser-predicted DG dependencies. The gold experiment allowed us to investigate the upper performance bound of our reranking technique and of our DG-derived features. We evaluate using the standard CCG dependency recovery metric over section 00 of CCGbank.

We use the reranker settings that Ng et al. (2010) found to provide best performance: regression learning, 10-best mode, and no feature pruning. We use the same experimental settings reported in Nivre et al. (2010) and McDonald et al. (2005) for the Maltparser and MSTparser respectively. This means that both parsers will produce a projective dependency tree for each scheme that we experimented with.

5.1 Overall Comparison

section 00 (dev)		LP	LR	LF
Baselines	C&C normal '07	87.27	86.41	86.84
	Reranker '10	87.57	86.69	87.13
Gold	CoNLL features	89.17	88.21	88.69
	Stanford features	88.97	88.06	88.51
	LTH features	88.95	88.01	88.48
	Fanse features	**89.61**	**88.72**	**89.16**
Malt Predicted	CoNLL features	87.74	86.85	87.29
	Stanford features	87.80	86.90	87.35
	LTH features	87.43	86.50	86.96
	Fanse features	**87.82**	**86.93**	**87.37**
MST Predicted	CoNLL features	**87.65**	**86.77**	**87.21**
	Stanford features	87.60	86.71	87.15
	LTH features	87.58	86.69	87.14
	Fanse features	87.61	86.72	87.17

Table 3: Parsing performance for the four DG schemes in gold and predicted configurations over section 00 of CCGbank

Table 3 records the labeled precision (LP), recall (LR) and F-score (LF) per system over section 00 of CCGbank. We compare our results to that of the C&C'07 baseline and the Reranker '10 baseline of Ng et al. (2010); the latter uses only the features defined in Ng et al. (2010), whereas our work includes the DG-derived features previous described. Each dependency scheme was tested in turn with the same feature set.

The gold results show that performance improvements of over 2% F-score are possible with

perfect DG-derived features. Each of the DG schemes performs similarly, with the exception of the Fanse scheme, which provides roughly 0.5% higher F-score than the others. The Fanse scheme is generated with the NP-bracketed version of the Penn Treebank, and also incorporates deeper linguistic analysis than the other schemes. However, it is interesting that the LTH dependencies generated from the same enriched corpus have the lowest upper bound.

The Maltparser-predicted results show that, with the exception of the LTH DG, the DG-derived features perform roughly on par with each other and generally better than the Reranker '10. However, the LTH features perform substantially worse than the others and also worse than Reranker '10; this corresponds with the Maltparser performing worst with respect to unlabeled attachment on that scheme. In contrast, the MSTparser feature results are each indistinguishable from one another, despite the very different performance of the baseline with respect to each scheme. The MSTparser results are also indistinguishable from the Reranker '10 system; this shows that the MSTparser does not produce enough useful variations compared with CCG for our procedure to work.

5.2 Isolation experiments

We took our best performing systems, which used the Fanse and CoNLL-based features for the Maltparser and MSTparser respectively, and investigated the individual impact of the DG-derived features. We did this by running the reranker with DG-derived features only over section 00 of CCGbank, and comparing our results with those of Reranker '10 (CCG features only) and the full system. Table 4 summarises the results; DG denotes our new features, and CCG denotes the CCG features.

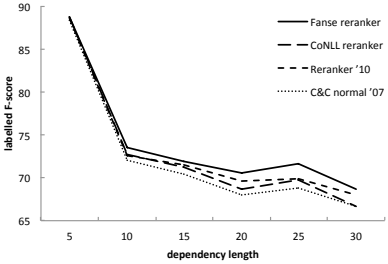
section 00 (dev)		LP	LR	LF
Reranker '10	CCG	87.57	86.69	87.13
Gold	Fanse DG	89.43	88.44	88.93
	CCG +Fanse DG	**89.61**	**88.72**	**89.16**
Malt Predicted	Fanse DG	86.79	85.79	86.29
	CCG +Fanse DG	**87.82**	**86.93**	**87.37**
MST Predicted	CoNLL DG	85.94	84.80	85.36
	CCG +CoNLL DG	**87.65**	**86.77**	**87.21**

Table 4: A comparison of DG-derived features in isolation and in combination with CCG reranker features over section 00 of CCGbank.

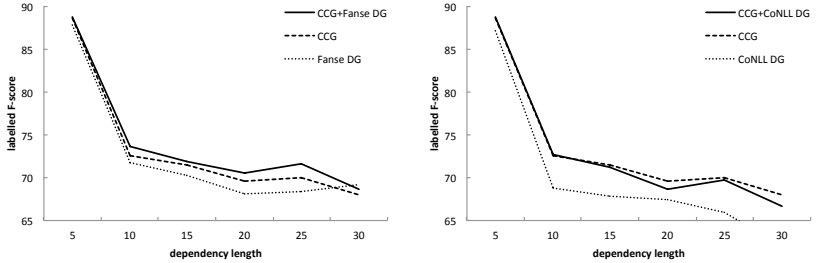
The gold results show that the DG-derived features perform strongly in isolation and outperform Reranker '10. However, performance is much worse using automatic DG parses compared to Reranker '10; F-score drops from 88.93% to 86.29% for the Maltparser-predicted experiment, and to 85.35% for MSTparser-predicted. These experiments are simply the result of removing the CCG features from the best performing system, and show that automatically produced DG features in isolation are harmful for reranking. However, the abstraction provided by automatic DG features prove useful in combination with CCG features and lead to a performance improvement.

5.3 Dependency lengths

Figure 7-(a) plots the labeled F-scores with respect to dependency length (number of words between the head and dependent) on section 00 for the best Fanse and CoNLL features. Our



(a) System comparison



(b) Feature comparison

Figure 7: Labeled F-score performance by bucketed dependency lengths for the Maltparser-predicted Fanse and MSTparser-predicted CoNLL experiments. Test conducted over CCGbank section 00.

new DG-derived features improve the Fanse performance across all dependency lengths compared to the C&C baseline and Reranker '10. In particular, we have improvements of 2.86% and 1.67% over the C&C baseline and Reranker '10 respectively when the dependency length is between 21 and 25, though the number of these dependencies is relatively small. In contrast, the MSTparser-predicted CoNLL features perform worse than the Reranker '10 as the dependency length grows though even though the CoNLL features perform better than the C&C'07 baseline for all dependency lengths.

We repeated the dependency length analysis with DG-derived features in isolation, as per the previous section. As can be seen from Figure 7-(b), Fanse DG-derived features in isolation perform poorly, while combining CCG and DG features gives an overall performance improvement over CCG features alone for all dependency lengths. Our CCG +Fanse DG reranker performs well for dependency lengths 21-25, with an F-score difference of 1.67% compared to Reranker '10. However, the F-score of CCG +CoNLL DG starts falling below the performance of CCG features beyond the dependency length 10.

These results all show a similar pattern, where F-score is very high for short dependencies, drops sharply up to length 6-10 words, and levels out for longer lengths. Interestingly, we

notice that the performance using only Fanse DG-derived features slightly increases as the dependency length grows beyond 15. For dependencies of length 25 and longer, the Fanse features in isolation actually outperform CCG features in isolation. Additionally, the performance improvement for the CCG +Fanse features over Reranker '10 is more substantial at longer dependency lengths – contrary to our initial expectations given the relative strengths of the parsers we used. One possible reason may be due to the generation of the Fanse scheme over the NP-enriched Penn Treebank. In contrast, the C&C parser was trained over a non-enriched version of CCGbank where all the noun phrases are right-branching. More errors may have crept into the noun phrase results as they typically contain short dependencies.

5.4 Subtractive feature analysis

section 00 (dev)		LP	LR	LF
Gold	CCG +Fanse DG	89.61	88.72	89.16
	-Pair/Neighbour	89.49	88.61	89.05
	-Sibling	89.64	88.76	89.20
	-Grandparent	89.58	88.71	89.14
	-Label	89.53	88.63	89.07
	-Length	**89.13**	**88.25**	**88.69**
	-Directionality	**89.18**	**88.31**	**88.74**
Malt Predicted	CCG +Fanse DG	87.82	86.93	87.37
	-Pair/Neighbour	87.66	86.78	87.22
	-Sibling	87.69	86.80	87.24
	-Grandparent	87.72	86.85	87.28
	-Label	87.67	86.79	87.23
	-Length	87.68	86.82	87.25
	-Directionality	87.69	86.82	87.25
MST Predicted	CCG +CoNLL DG	87.65	86.77	87.21
	-Pair/Neighbour	87.56	86.67	87.12
	-Sibling	87.53	86.65	87.09
	-Grandparent	87.58	86.70	87.14
	-Length	87.56	86.69	87.12
	-Directionality	87.76	86.84	87.30

Table 5: Subtractive analysis of each DG feature for the Fanse and CoNLL schemes on CCGbank section 00. Bolded rows indicate the most substantial performance drops.

We performed a subtractive analysis to investigate the individual contribution of each feature type. We can see in Table 5 that different features are important for the gold and predicted experiments. Removing the length or directionality-dependency features causes a substantial performance decrease in the gold-standard experiment. This seems to suggest that certain dependencies between words will only exist with a certain length between head and dependent, and that words also have a relatively predictable number of incoming and outgoing arcs in the gold standard.

In contrast, the removal of no individual feature causes a significant change in F-score in either the Maltparser-predicted or MSTparser-predicted experiments. The introduction of parse errors has reduced the reliance of the reranker on length and directionality features, and caused it to smooth out the weights to other features. In this setting it is the combination of small

contributions from many features that is important.

5.5 Final results

We used the Fanse and CoNLL features, as these performed best on the development data for the Maltparser and MSTparser respectively, for a single run on the final test set of section 23 of CCGbank. Table 7 shows that the Fanse features with the Maltparser improve F-score relative to the C&C baseline and Reranker '10 systems. In particular, our final result of 87.93% F-score is a 0.5% improvement over the baseline parser, and a 0.35% gain over Reranker '10. These results are significant at $p < 0.05$, as tested using Bikel's approximate randomisation procedure [6].

section 23 (test)	LP	LR	LF
C&C normal '07	87.81	87.06	87.43
Reranker '10 (CCG)	87.98	87.18	87.58
Fanse CCG +DG features (Maltparser)	**88.32**	**87.54**	**87.93**
CoNLL CCG +DG features (MSTparser)	88.02	87.20	87.61

Table 7: Final test results for the best DG parser-predicted features over section 23 of CCGbank.

Conclusion

In this paper we proposed new DG-derived features, generated by a dependency parser and incorporated into a CCG parser using reranking. We observe significant performance improvements using the DG-derived features from the Maltparser, and also show that there remains substantial potential in the DG-derived features with improved dependency parsing.

The LTH and Fanse dependency schemes were created from the NP-bracketed Penn Treebank of Vadas and Curran (2007). While the Fanse features performed the best in our experiments with the Maltparser, the LTH features performed poorly, further work should experiment with the standard Penn Treebank as the basis for conversion, as well as generating the CoNLL and Stanford schemes with the enriched corpus.

We plan to develop features to address more specific linguistic phenomena such as coordination and prepositional phrase attachment. The dependency schemes each represent these phenomena differently (see Section 2.5), and they are a particular issue in parsing. New features that compare the way different schemes represent these phenomena may allow us to better represent and reproduce them in parsing.

Our work focused on English parsing and we used the Maltparser and MSTparser independently of one another to generate features. It would be interesting to examine the effect of our features on parsing and reranking in different languages, as well as developing new features that compare the output of both dependency parsers. Additionally, we did not use higher order features for the Maltparser and MSTparser. Since these features are not present in the C&C parser, further work should explore whether enabling these features helps reranking.

We have developed a technique for incorporating parse information from one formalism as features for reranking another. This allows us to exploit the strengths of each formalism in a flexible framework with arbitrarily complex features.

[6] based on Dan Bikel's script at `http://www.cis.upenn.edu/~dbikel/software.html#comparator`

Acknowledgments

We would like to thank Matthew Honnibal for his valuable feedback on this work and thank the anonymous reviewers for their helpful comments. This research has been supported by a Macquarie University Research Excellence Scholarship, a Faculty of Science Postgraduate Research Fund and an Australian Research Council DP110102593. The second author was supported by an Australian Postgraduate Award and a Vice-Chancellor's Research Scholarship.

References

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

Brennan, F. (2008). k-best Parsing Algorithms for a Natural Language Parser. Master's thesis, University of Oxford.

Briscoe, T. and Carroll, J. (2006). Evaluating the accuracy of an unlexicalized statistical parser on the PARC DepBank. In *Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions*, pages 41–48, Sydney, Australia.

Charniak, E. and Johnson, M. (2005). Coarse-to-fine *n*-best parsing and MaxEnt discriminative reranking. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics*, pages 173–180, Ann Arbor, Michigan.

Chen, W., Kawahara, D., Uchimoto, K., Zhang, Y., and Isahara, H. (2009). Using short dependency relations from auto-parsed data for chinese dependency parsing. *ACM Transactions on Asian Language Information Processing (TALIP)*, 8(3):10.

Clark, S. and Curran, J. R. (2007). Wide-coverage efficient statistical parsing with CCG and log-linear models. *Computational Linguistics*, 33(4):493–552.

Clark, S. and Hockenmaier, J. (2002). Evaluating a Wide-Coverage CCG Parser. In *Proceedings of the Beyond PARSEVAL Workshop at the 3rd International Conference on Language Resources and Evaluation (LREC-02)*, pages 60–66, Las Palmas, Canary Islands, Spain.

Clark, S., Hockenmaier, J., and Steedman, M. (2002). Building deep dependency structures using a wide-coverage CCG parser. In *Proceedings of 40th Annual Meeting of the Association for Computational Linguistics*, pages 327–334, Philadelphia, Pennsylvania, USA.

Collins, M. (2000). Discriminative reranking for natural language parsing. In *Machine Learning: Proceedings of the Seventeenth International Conference (ICML 2000)*, pages 175–182, Stanford, California.

de Marneffe, M.-C. and Manning, C. D. (2008). The Stanford typed dependencies representation. In *Coling 2008: Proceedings of the workshop on Cross-Framework and Cross-Domain Parser Evaluation*, pages 1–8, Manchester, UK.

Farkas, R., Bohnet, B., and Schmid, H. (2011). Features for phrase-structure reranking from dependency parses. In *Proceedings of the 12th International Conference on Parsing Technologies (IWPT-11)*, pages 209–214.

Hockenmaier, J. and Steedman, M. (2007). CCGbank: a corpus of CCG derivations and dependency structures extracted from the penn treebank. *Computational Linguistics*, pages 355–396.

Huang, L. and Chiang, D. (2005). Better k-best Parsing. In *Proceedings of the 9th International Workshop on Parsing Technology (IWPT-05)*, pages 53–64, Vancouver, British Columbia, Canada.

Johansson, R. and Nugues, P. (2007). Extended constituent-to-dependency conversion for english. In *Proc. of the 16th Nordic Conference on Computational Linguistics (NODALIDA)*.

King, T. H., Crouch, R., Riezler, S., Dalrymple, M., and Kaplan, R. M. (2003). The PARC 700 Dependency Bank. In *Proceedings of the 4th International Workshop on Linguistically Interpreted Corpora*, pages 1–8, Budapest, Hungary.

Marcus, M. P., Santorini, B., and Marcinkiewicz, M. A. (1993). Building a large annotated corpus of English: The Penn Treebank. *Computational Linguistics*, 19(2):313–330.

McDonald, R., Crammer, K., and Pereira, F. (2005). Online large-margin training of dependency parsers. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 91–98, Ann Arbor, Michigan. Association for Computational Linguistics.

McDonald, R. and Nivre, J. (2007). Characterizing the errors of data-driven dependency parsing models. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 122–131.

Ng, D. and Curran, J. R. (2012). Dependency Hashing for n-best CCG Parsing. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (ACL-12)*, pages 497–505, Jeju Island, Korea.

Ng, D., Honnibal, M., and Curran, J. R. (2010). Reranking a wide-coverage CCG parser. In *Proceedings of the Australasian Language Technology Association Workshop 2010*, pages 90–98, Melbourne, Australia.

Nivre, J., Hall, J., Kübler, S., McDonald, R., Nilsson, J., Riedel, S., and Yuret, D. (2007a). The CoNLL 2007 shared task on dependency parsing. In *Proceedings of the CoNLL Shared Task Session of EMNLP-CoNLL 2007*, pages 915–932, Prague, Czech Republic.

Nivre, J., Hall, J., Nilsson, J., Chanev, A., Eryigit, G., Kübler, S., Marinov, S., and Marsi, E. (2007b). Maltparser: A language-independent system for data-driven dependency parsing. *Natural Language Engineering*, 13(02):95–135.

Nivre, J., Rimell, L., McDonald, R., and Gómez Rodríguez, C. (2010). Evaluation of dependency parsers on unbounded dependencies. In *Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010)*, pages 833–841, Beijing, China.

Øvrelid, L., Kuhn, J., and Spreyer, K. (2009). Cross-framework parser stacking for data-driven dependency parsing. *Traitement Automatique des Langues (TAL) Special Issue on Machine Learning for NLP*, 50(3):109–138.

Rimell, L. and Clark, S. (2009). Porting a lexicalized-grammar parser to the biomedical domain. *Journal of Biomedical Informatics*, 42(5):852–865.

Rimell, L., Clark, S., and Steedman, M. (2009). Unbounded dependency recovery for parser evaluation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 813–821, Singapore.

Sagae, K., Miyao, Y., and Tsujii, J. (2007). Hpsg parsing with shallow dependency constraints. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 624–631, Prague, Czech Republic. Association for Computational Linguistics.

Schmid, H. (2004). Efficient parsing of highly ambiguous context-free grammars with bit vectors. In *Proceedings of Coling 2004*, pages 162–168, Geneva, Switzerland.

Steedman, M. (2000). *The Syntactic Process*. MIT Press, Cambridge, MA.

Tratz, S. and Hovy, E. (2011). A fast, accurate, non-projective, semantically-enriched parser. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 1257–1268, Edinburgh, Scotland, UK.

Vadas, D. and Curran, J. (2007). Adding noun phrase structure to the Penn Treebank. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 240–247, Prague, Czech Republic.

Inducing Crosslingual Distributed Representations of Words

Alexandre Klementiev *Ivan Titov* *Binod Bhattarai*
Saarland University, Saarbrücken, Germany
{aklement, titov, bhattara}@mmci.uni-saarland.de

ABSTRACT
Distributed representations of words have proven extremely useful in numerous natural language processing tasks. Their appeal is that they can help alleviate data sparsity problems common to supervised learning. Methods for inducing these representations require only unlabeled language data, which are plentiful for many natural languages. In this work, we induce distributed representations for a pair of languages jointly. We treat it as a multitask learning problem where each task corresponds to a single word, and task relatedness is derived from co-occurrence statistics in bilingual parallel data. These representations can be used for a number of crosslingual learning tasks, where a learner can be trained on annotations present in one language and applied to test data in another. We show that our representations are informative by using them for crosslingual document classification, where classifiers trained on these representations substantially outperform strong baselines (e.g. machine translation) when applied to a new language.

KEYWORDS: distributed representations, multilingual learning, direct transfer of annotation.

Proceedings of COLING 2012: Technical Papers, pages 1459–1474,
COLING 2012, Mumbai, December 2012.

1 Introduction

Word representations induced to capture syntactic and semantic properties of words have been extremely useful for numerous natural language processing applications (Collobert and Weston, 2007; Turian et al., 2010). Their primary appeal is that they can be induced using abundant unsupervised data and then used directly or as additional features to alleviate the data sparsity problem common in the supervised learning scenario.

Most of the prior work on inducing these representations has focused on a single language, English, which enjoys the largest repository of available annotated resources. In this work, we focus on a single representation for a pair of languages such that semantically similar words are closer to one another in the induced representation irrespective of the language. Learning with these representations for a task where annotation is available for one language would induce a classifier which could be used in another language lacking sufficient resources for this task. We pick one example of such a task, document classification, to show that a classifier trained using these representations in one language achieves high accuracy in another language where no annotation is available (the set-up called *direct transfer of annotation*).

Our main contribution is a general technique for inducing crosslingual distributed representations. We use an existing model for learning distributed representations in individual languages; however, motivated by the multitask learning (MTL) setting of Cavallanti et al. (2010), we propose a method to jointly induce and align these representations. We use word co-occurrence statistics from parallel data to define a signal for aligning the latent representations in both languages as we induce them. In MTL terminology, we treat words as individual tasks; words that are likely to be translations of one another (based on bitext statistics) are treated as related tasks and effectively help to align representations in both languages during learning.

We use a variant of a neural network language model of Bengio et al. (2003) to induce the latent representations in individual languages. These models learn a lower-dimensional embedding of words arguably capturing their syntactic and semantic properties (Socher et al., 2011a).

In sum, the contributions of this work are:

- we frame induction of crosslingual distributed word representations as joint induction and alignment of distributed representations in individual languages;
- we apply our framework to the neural network language modeling approach of Bengio et al. (2003);
- although our goal is not to beat the state of the art in crosslingual document classification, we use this task to show that the crosslingual embeddings we induce enable us to transfer a classifier trained on one language to another without any adaptation.

The crosslingual representation induction set-up we propose is motivated by the multitask learning (MTL) setting of Cavallanti et al. (2010), so we begin with a brief overview in Section 2, in part to introduce terminology and notation. In our set-up, we do not commit to a particular technique for learning representations in individual languages, but rather propose a general technique for jointly inducing and aligning representations in multiple languages. However, since we apply the setup to a neural probabilistic language model in this work, we also give a short overview of a variant of the method from Bengio et al. (2003) in Section 3. In Section 4,

we define the crosslingual distributed representation induction as the joint task of learning distributed representations in two languages. Finally, Section 5 gives experimental evaluation of the induced crosslingual representations on the crosslingual document classification task.

2 Multitask Learning

The goal of multitask learning (MTL) is to improve generalization performance across a set of related tasks by learning them jointly. MTL is particularly relevant when sufficient annotation is not available for (some of) these tasks.

In the multitask set-up of Cavallanti et al. (2010), at time t a multitask learner receives an example relevant to one of K tasks it is learning. Along with the example $x_t \in \mathbb{R}^m$, and the correct binary label $y_t \in \{-1,+1\}$, the learner is supplied with the task index $i_t \in [1,K]$. It then considers a compound multitask instance $\phi_{x_t} \in \mathbb{R}^{mK}$:

$$\phi_{x_t} = (\underbrace{0,\ldots,0}_{(i_t-1)m}, x_t^\top, \underbrace{0,\ldots,0}_{(K-i_t)m})^\top$$

A multitask version of the perceptron algorithm they propose keeps a weight vector for each task. Assuming that at time t the algorithm has made s mistakes, the compound weight vector at t is $v_s = (v_{1,s}^\top, \ldots, v_{K,s}^\top)^\top$, where $v_{j,s} \in \mathbb{R}^m$ is the weight vector for task j. When a mistake is made at time t, updates are performed not only for the weight vector of task i_t, but also for the remaining $K-1$ tasks. The rate of the update for each task is defined by a $K \times K$ *interaction matrix* A, which, intuitively, encodes relatedness between the tasks. When a learner makes a mistake, the compound weight vector update rule applied is $v_s \leftarrow v_{s-1} + (A \otimes I_m)^{-1}\phi_{x_t}$, where $\otimes$ is the Kronecker product and I_m is the identity matrix of size m. This update can be rewritten as separate updates for each task:

$$v_{j,s} \leftarrow v_{j,s-1} + y_t A^{-1}_{j,i_t} x_t \qquad \forall j \in [1,K]$$

This learning algorithm directly corresponds to the minimization of the following objective:

$$L(v) = \sum_t L^{(t)}(v) + \frac{1}{2} v^\top (A \otimes I_m) v \tag{1}$$

where $L^{(t)}(v) = \left[1 - y_t v^\top \phi_{x_t}\right]_+$ is the hinge loss on the example at time t. Consequently, this setup can be naturally extended to other loss function and to non-linear models. We will use it to formalize the crosslingual representation induction task in Section 4.

2.1 Encoding Prior Knowledge in Interaction Matrix A

Let us consider the following simple interaction matrix with the corresponding inverse:

$$A = \begin{pmatrix} K & -1 & \cdots & -1 \\ -1 & K & \cdots & -1 \\ \vdots & \vdots & \ddots & \vdots \\ -1 & -1 & \cdots & K \end{pmatrix} \qquad A^{-1} = \frac{1}{K+1} \begin{pmatrix} 2 & 1 & \cdots & 1 \\ 1 & 2 & \cdots & 1 \\ \vdots & \vdots & \ddots & \vdots \\ 1 & 1 & \cdots & 2 \end{pmatrix}$$

That is, when a mistake is made at time t, the rate of update is $2/(K+1)$ for task i_t and half as large for the other $K-1$ tasks. In other words, A defines all tasks as "equally related" to any other task.

Cavallanti et al. (2010) propose an elegant way of encoding richer prior knowledge in the interaction matrix. Relatedness between tasks can be naturally represented by an undirected graph $G=(R,E)$. The vertices R of the graph are tasks, and a pair of vertices are connected by an edge in E only if we believe that the corresponding tasks are related. The interaction matrix can then be defined as:

$$A = I + L \tag{2}$$

where I is the identity matrix and L is the Laplacian of graph G, defined as a $K \times K$ matrix:

$$L_{i,j}(G) = \begin{cases} \deg(i) & \text{if } i = j \\ -1 & \text{if } (i,j) \in E \\ 0 & \text{otherwise} \end{cases}$$

where $\deg(i)$ is the number of edges involving the vertex i.

This definition of the task interaction matrix A naturally generalizes to weighted graphs $H=(R,E,S)$, where S are weights associated with edges E. The graph Laplacian becomes:

$$L_{i,j}(H) = \begin{cases} \sum_{(i,k)\in E} s(i,k) & \text{if } i = j \\ -s(i,j) & \text{if } (i,j) \in E \\ 0 & \text{otherwise} \end{cases}$$

where $s(i,j)$ is the weight of $(i,j) \in E$. We will use these extended definitions in this work to include prior knowledge about the *degree* of relatedness between tasks. Note that the matrix A is invertible: a graph Laplacian is always positive semi-definite and consequently adding an identity matrix is guaranteed to yield a positive-definite matrix.

3 Neural Language Models

The goal of statistical language modeling methods is to estimate the joint probability distribution of word sequences occurring in a natural language. Neural probabilistic models learn a latent multi-dimensional representation of words and use them to estimate the probability distribution of word sequences. An important side-effect of training neural language models is the fact that the learned latent representations capture syntactic and semantic properties of context words, because these properties are predictive of a possible next word.

Lets us assume that a word sequence is a string of words $w_1,\ldots,w_T$, and $w_i \in V, i \in (1,\ldots,T)$ for some vocabulary V. For notational convenience, we will assume that the $|V|$ types are indexed, and w_i could refer to either the i-th token in the sequence or the corresponding index, depending on the context in which it is used.

When building a statistical n-gram language model, the aim is to estimate a conditional distribution of the next word given the preceding $n-1$ words, i.e. $P(w_t|w_{t-n+1:t-1})$, where

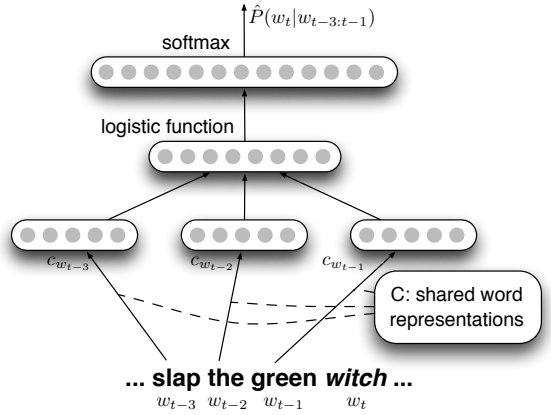


Figure 1: Neural architecture (3-gram language model) for inducing word representations in a single language.

$w_{t-n+1:t} = (w_{t-n+1}, \ldots, w_{t-1}, w_t)$ is a subsequence of n words. The language model of Bengio et al. (2003) estimates the distribution over the next word $w_t \in V_{out}$ in the sequence (see Figure 1) as follows[1]:

1. Uses a shared representation vector $c \in \mathbb{R}^{|V_{in}|d}$, a concatenation of representations of all vocabulary words $c = (c_1^\top; \ldots; c_{|V_{in}|}^\top)^\top$, to map each of the context words $w_i, i \in [t-1, \ldots, t-n+1]$ to its distributed representation c_{w_i}.

2. Concatenates all of the word representations of context $w_{t-n+1:t-1}$ preserving the order, $(c_{w_{t-n+1}}^\top; \ldots c_{w_{t-2}}^\top; c_{w_{t-1}}^\top)$.

3. The hidden layer applies a linear transformation followed by the logistic function on the concatenated embeddings.

4. Finally, the output layer separates out the classes (words in V_{out}) and applies the softmax function to ensure that the network outputs can be interpreted as a probability distribution. We will call W all network weights other than the embedding c.

The key component of the architecture is the *shared* embedding c, which is learned along with the rest of the network parameters using backpropagation. The model captures local context, so that the induced d-dimensional distributed vectors for words in the vocabulary V_{out} are "closer" for more semantically "similar" words. Thus, the induced representation can help alleviate sparsity issues in a supervised learning setup (Turian et al., 2010).

[1]Note that we make a distinction between the input (V_{in}) and output (V_{out}) vocabularies. It will be relevant in the experimental section to speed up learning, but in the rest of the paper they can be assumed the same,$V_{in} = V_{out} = V$.

Learning maximizes the data likelihood objective with respect to model parameters $\theta = (W, c)$:

$$L(\theta) = \sum_{t=1}^{T} \log \hat{P}_\theta(w_t | w_{t-n+1:t-1}) \tag{3}$$

The training procedure uses stochastic gradient descent: it iteratively updates parameters using a gradient at each training subsequence $w_{t-n+1:t}$. Specifically, for the word representations c, the updates have the form:

$$c_w \leftarrow c_w + \eta \frac{\partial L^{(t)}(\theta)}{\partial c_w}, \tag{4}$$

where η is the learning rate and $L^{(t)}(\theta) = \log \hat{P}_\theta(w_t | w_{t-n+1:t-1})$ is the contribution of the example to the data likelihood objective. Note that only the representations of words in the contextual window (i.e. their corresponding parts of c) are modified during each step.

4 Crosslingual Representation Induction

The neural language model we described in Section 3 induces an embedding c, so that words which are semantically similar are close to one another in c. In this work, our goal is to have the same property hold across two languages.[2] We train neural language models jointly for both languages and induce a common embedding.

We cast crosslingual distributed representation induction as a multitask learning problem by treating each word w in our languages' vocabularies as a separate task. The set of related tasks for each w are then the possible translations of the word in the other language. When encoding relatedness and defining an interaction matrix A, we make use of parallel data (a set of sentences and their translations). These resources are available for many language pairs and include large volumes of multilingual parliamentary proceedings, book translations, etc. Standard Machine Translation tools (e.g. GIZA++ (Och and Ney, 2003)) can be used to induce alignments between words on both sides of the bitext.

Assuming that word alignments are available, we first define a complete undirected bipartite weighted graph H with two disjoint sets of vertices corresponding to the input vocabularies V_{in} of the two languages, and edges labeled with the number of alignments between each pair of words in the two sets. The edge weights indicate the fit of a pair of words as translations, and thus encode the degree of relatedness between the two corresponding tasks. We can now directly apply the definition of the interaction matrix from Section 2, defining $s(w, \tilde{w})$ as the number of alignments between words w and $\tilde{w}$.

We use a separate neural language model for each language l, parameterized by $\theta^{(l)} = (W^{(l)}, c)$. Although the notation might suggest that embedding c is shared across languages, this is not the case, as we distinguish between word types of the two languages: for example, the word *handy* in English and the word *Handy* in German (meaning a mobile phone) would be treated as two different word types. Given an interaction matrix A, we can extend the MTL formalization (1)

[2] Our methods can be trivially extended to more than two languages.

and reformulate the monolingual learning objective (3) as:

$$L(\theta) = \sum_{l=1}^{2} \sum_{t=1}^{T^{(l)}} \log \hat{P}_{\theta^{(l)}}(w_t^{(l)} | w_{t-n+1:t-1}^{(l)}) + \frac{1}{2} c^\top (A \otimes I_d) c \tag{5}$$

where $T^{(l)}$ and $w_t^{(l)}$ are the number of words in the data set for language l and a word at position t in this corpus, respectively. As before, $\otimes$ denotes the Kronecker product and I_d is the identity matrix of size d.

Intuitively, the first (language modeling) part of the learning objective (5) captures the syntactic and semantic similarities between words in each of the two languages, while the second (MTL regularization term) ensures that the learned representations are aligned across the two languages. Note that additional information such as WordNet synsets could in principle be used to also encode relatedness between words within each language into A. However, these resources are unavailable for most languages. Also, similar type of information is already induced by the neural language model.

The stochastic gradient descent procedure would now iteratively update parameters using a gradient at each training subsequence $w_{t-n+1:t}^{(l)}$ in both languages. The monolingual update formula (4) now becomes:

$$c_w \leftarrow c_w + \eta \sum_{w'} A^{-1}_{w',w} \frac{\partial L^{(l,t)}(\theta)}{\partial c_{w'}}, \tag{6}$$

where η is the learning rate and $L^{(l,t)}(\theta) = \log \hat{P}_{\theta^{(l)}}(w_t^{(l)} | w_{t-n+1:t-1}^{(l)})$ is the contribution of the training example. In this formulation, both representations of the words in the contextual window $c_{t-n+1:t-1}$ and words w' "related" to them (i.e. those w' for which $A^{-1}_{w,w'} \neq 0$ for any contextual word w) are modified on each training step. These updates can be computed efficiently as long as A^{-1} is sufficiently sparse.

Computing these learning updates requires the inverse of the interaction matrix A. However, the dimensionality of the matrix is equal to the total number of word types in both languages, so the standard cubic-time Gaussian elimination is infeasible even for moderately sized datasets. Direct computation of A^{-1} can be made more efficient if we compute it separately for each of the connected components in our graph. Still, because of large sizes of the input vocabularies and the noise in word alignments, this computation remains impractical. A future direction could be to explore faster algorithms which could take advantage of our particular setup (i.e., sparse matrices corresponding to bipartite graphs (Li, 2009)), or to use approximate iterative algorithms (Fouss et al., 2007). In our experiments, we approximate A^{-1} directly with the following heuristic:

$$\hat{A}^{-1}_{w,w} = \frac{m_w + 1}{m_w + 1 + \sum_{\tilde{w}} s(w, \tilde{w})}$$

$$\hat{A}^{-1}_{w,w'} = \frac{s(w, w')}{m_w + 1 + \sum_{\tilde{w}} s(w, \tilde{w})}$$

where $m_w = \max_{\tilde{w}} s(w, \tilde{w})$. Intuitively, the effective update rate for a word w' "related" to a contextual word w is proportional to their alignment count. The rate applied to a context word itself is only slightly larger than the rate used for the word w' most frequently aligned to it if the corresponding alignment frequency $m_w = s(w, w')$ is high. However, if m_w were 1 and w were not aligned to other words, the self update rate $\hat{A}^{-1}_{w,w}$ would be twice as large. Consequently, the $+1$ term reduces the effect of potentially noisy counts. While this heuristic does not quite correspond to the exact computation of the inverse of the interaction matrix A as we defined it for weighted graphs, it plays a similar role, has a similar form (compare with the example in section Section 2), and is easy to compute.[3]

5 Experiments

The technique we propose induces crosslingual representations capturing relatedness of words in a pair of languages. We use a particular supervised learning task, crosslingual document classification, and show that a classifier trained using these representations in one language achieves high accuracy in another language where no annotation is available. Note that our goal is not to induce a state-of-the-art classifier, but rather to examine the informativeness of the induced representations.[4] Thus, we keep the classification experiments simple: we chose a learning algorithm requiring no parameter tuning and used simple features.

5.1 Data

In our experiments, we induce crosslingual embeddings and use them for multilinigual document classification for the English-German language pair. We use the following resources:

- English (**en**) and German (**de**) sections of the Europarl v7 parallel corpus (Koehn, 2005) to induce our baseline systems and to compute the interaction matrix A (see Section 4). We used GIZA++ (Och and Ney, 2003) to induce word alignments, keeping only bidirectional alignments. In the context of our model, parallel data is only used to estimate the interaction matrix A. When constructing A, we discard word pairs aligned only once in order to reduce the number of effective updates during gradient descent (see equation (6)).

- A subset of the English and German sections of the Reuters RCV1/RCV2 corpora (Lewis et al., 2004) to induce crosslingual embeddings and for the crosslingual document classification experiments. The corpus contains documents (news stories) in several languages which are assigned topics capturing the major subjects of the story. In the English dataset, there are four topics (each with hierarchy of sub-topics): `CCAT` (Corporate/Industrial), `ECAT` (Economics), `GCAT` (Government/Social), and `MCAT` (Markets). Note that these documents are not parallel.

  Each document can be labeled with multiple topics, however, since we do not want to consider multi-label classification in our experiments, we only select documents assigned to single topics. Of those, we sampled 34,000 **en** and 42,753 **de** documents (they were selected with the goal of keeping roughly 8 million tokens for each language). which we used for unsupervised induction of crosslingual representations.

[3] In preliminary small scale experiments we did not observe a significant advantage from using the true inverse matrix, and therefore we chose not to resort to more accurate approximations.

[4] An embedding specifically learned for the classification task would require modifications of the learning objective. While it is likely to improve the performance on this specific task, it is not the aim of this work.

For our classification experiments, we randomly selected 15,000 documents from our sampled dataset and used a third of them as a test set, with the remainder used to construct training sets of sizes between 100 and 10,000 documents. We repeated this procedure for both **en** and **de**; for both languages, the majority class was `MCAT` with roughly 46.8% of the documents.

All datasets were normalized with the tools distributed by the 2012 SMT workshop (Callison-Burch et al., 2012). The list of RCV1/RCV2 document names we used in our experiments along with the crosslingual word representations we induced are available at `http://www.ml4nlp.de/code-and-data`.

5.2 Our Model and Baselines

Our neural language model architecture (Section 3) was the same for both languages with 25 hidden units, and the context size of 4. We induced representations of $d = 40$ dimensions for input vocabularies of $|V_{in}^{\mathbf{en}}| = 43,614$ and $|V_{in}^{\mathbf{de}}| = 50,110$ words (filtering out words which occur fewer than five times in our dataset). However, to speed up training,[5] we learn on a subset of training sequences choosing the 3,000 most frequent words in **en** and **de** for their output vocabularies $V_{out}^{\mathbf{en}}$ and $V_{out}^{\mathbf{de}}$, respectively. The representations were induced from our subset of RCV1/RCV2 dataset using word alignments from Europarl v7 (see Section 5.1). We ran the learning procedure for 40 iterations, which took about 10 CPU days and is linearly parallelizable. Learning rate was set to 0.005 and was reduced when the training data likelihood went up, as is common when training neural networks.

We used the averaged version of the perceptron algorithm (Collins, 2002) to train a multiclass document classifier, so that we do not need to tune any parameters, with the exception of the the number of epochs, which we set to 10 in all experiments (the results were not sensitive to this parameter). Our goal is to train a classifier in one language and test it on data in another, so we compared the following classifiers:

- A classifier which used features based on the crosslingual representations we induced (*DistribReps*) and was trained on supervised training data in one language and *directly* tested on documents in the other. We represent each document as an average of d-dimensional representations of all of its tokens weighted by their *idf* score (Huang et al., 2012).
- A classifier with word count features which was trained and tested on the second language documents translated into the original language. Translations are done by replacing each word in a test document by the word most frequently aligned to it in the parallel data (*Glossed*). Unaligned words were left as is.
- Using a machine translation system instead of simple glossing would provide a natural baseline (Fortuna and Shawe-Taylor, 2005; Shi et al., 2010). So, another baseline (*MT*) is similar to the previous with the exception that the second language documents were translated by the standard phrase-based machine translation model (Koehn et al., 2007) using default parameters and a 5-gram language model trained on Europarl v7 data.
- For reference, we also include majority class predictions (*Majority Class*).

[5] In particular, computing the normalization in the softmax function, is linear in $|V_{out}|$.

january		*president*		*said*	
en	**de**	**en**	**de**	**en**	**de**
january	januar	president	präsident	said	sagte
february	februar	king	präsidenten	reported	erklärte
november	november	hun	minister	stated	sagten
april	april	areas	staatspräsident	told	meldete
august	august	saddam	hun	declared	berichtete
march	märz	minister	vorsitzenden	stressed	sagt
june	juni	advisers	us-präsident	informed	ergänzte
december	dezember	prince	könig	announced	erklärten
july	juli	representative	berichteten	explained	teilt
september	september	institutional	außenminister	warned	berichteten
oil		*microsoft*		*market*	
en	**de**	**en**	**de**	**en**	**de**
oil	baumwolle	microsoft	microsoft	market	markt
car	kaffee	intel	intel	papers	marktes
energy	telekommunikation	instrument	chemikalien	side	fonds
air	tabak	chapman	endesa	economy	sektor
tobacco	rindfleisch	endesa	kabel	duration	laufzeit
steel	öl	distillates	hewlett-packard	sector	montreal
housing	benzin	pty	guinness	tobacco	verkäufer
cotton	stahl	hewlett-packard	dienste	montreal	papiere
insurance	strom	guinness	thomson	house	fracht
technology	milch	potash	exxon	pay	hersteller

Table 1: Example English words along with 10 closest words both in English (**en**) and German (**de**), using the Euclidean distance in the induced joint distributed representation.

5.3 Classification Results

Before looking at the classification results, let us examine the distributed representations we induce with a small experiment. Table 1 shows six English words, each along with ten words in English and German ranked by the Euclidean distance in the induced embedding. With few exceptions, all six end up being near semantically similar words in both languages. Identical ranking of months in both languages in the first example suggest that aligned data brought translations very close to one another in the induced embedding.

We ran crosslingual classification experiments training on English and testing on German documents, varying the training data size from 100 to 10,000 documents, then repeated the same experiments going from German to English. Classification results are summarized on Figure 2 and a single point is detailed in Table 2. Classifiers based on distributed representations substantially outperform all baselines. They are especially beneficial when the amount of training data is small, effectively taking advantage of plentiful unsupervised data used for inducing crosslingual word representations. While their performance is high relative to the baselines, it does not change significantly with the training data size. We believe that is likely due to relatively low embedding dimensionality ($d = 40$); 100 examples were sufficient to learn a good classifier for this representation. Increasing the size of the hidden representation is likely to improve the results. Note that these embeddings were not induced specifically for this task. It is likely that these results would improve if we reformulate the objective with the classification task in mind (see e.g. (Titov, 2011; Glorot et al., 2011)).

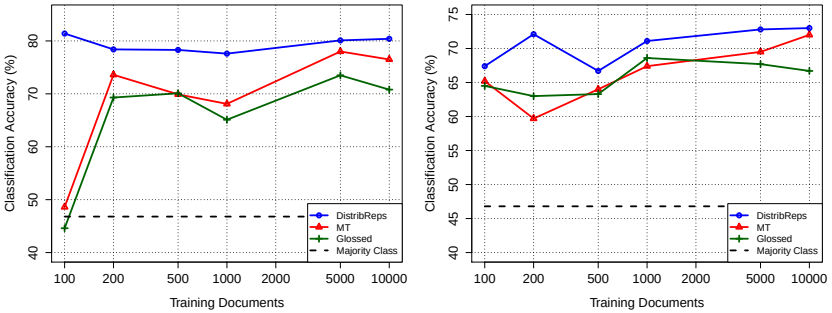


Figure 2: Classification accuracy with three types of features: crosslingual distributed representations (*DistribReps*), translated (*MT*), and glossed (*Glossed*) words, and the majority class baseline (*Majority Class*). The results are for training on English and testing on German documents (left) and vice versa (right).

6 Additional Related Work

In the last decade crosslingual methods have attracted a lot of attention both in NLP and closely related communities such as information retrieval (Lavrenko et al., 2002) and information management (Frederking et al., 2001). Much of this work has focused on techniques for porting methods and resources from one language to another (see, e.g., crosslingual document classification (Fortuna and Shawe-Taylor, 2005; Shi et al., 2010)). Development of crosslingual models (e.g., topic models (Zhang et al., 2010; Mimno et al., 2009)) has also attracted some attention. However, these approaches either do not induce representations of individual words and, as such, may not be very useful for methods dealing with richer linguistic structures (such as syntactic parsing or semantic role labeling) or they focus on porting a specific method (e.g., named entity recognizer (Steinberger and Pouliquen, 2007)). This contrasts significantly with our objective: inducing fine-grain distributed word representation useful in virtually arbitrary NLP problems.

Possibly the most related work to ours is the method for inducing crosslingual Brown clusterings (Täckström et al., 2012). They also use multi-lingual parallel data to enforce a form of crosslingual agreement in the induced representations. However, atomic cluster labels arguably are not capable of encoding multiple factors or views on the syntactic and semantic properties of words, and, consequently, may be less informative for many applications. For a detailed comparison of properties of distributed representations and Brown clustering we refer the reader to Turian et al. (2010).

Construction of crosslingual representations and similarity functions has also been considered in the related area of distributional semantics (van der Plas and Tiedemann, 2006; Agirre et al., 2009) where a word is represented as a vector and each of its components encodes the strength of co-occurrence with a specific lexical or syntactic context (Rapp, 1995). These representations again have very different properties from the ones considered here: for example, they are typically very highly dimensional and, consequently, may be less useful as features in

	en → de	de → en
DistribReps	77.6	71.1
MT	68.1	67.4
Glossed	65.1	68.6
Majority-Class	46.8	46.8

Table 2: Classification accuracy for training on English and German with 1000 labeled examples.

classifiers. Also they generally cannot be created with a specific application in mind, whereas word representations can be learned to be useful for a specific problem (Collobert and Weston, 2007).

7 Conclusions and Future Work

In this work, we propose a general method for inducing crosslingual distributed representations for a pair of languages. We treat it as a multitask learning problem, where each task corresponds to a word in the vocabularies of the two languages, and relatedness information between them is estimated from word alignments in parallel data. Intuitively, task relatedness information encoded in the interaction matrix A is used to align the representations in both languages as we learn them. Words in either language that are similar to each other end up being "close" in the joint representation. However, since aligned resources may not be available for a given language pair, an investigation of robustness of our setup to the amount of parallel data as well as using alternative resources to define A is an interesting future direction.

Distributed representations of multi-word expressions (phrases) have recently been shown very useful for sentiment analysis (Socher et al., 2011b). Inducing these representations in multiple languages is likely to benefit tasks like low-resource machine translation (Klementiev et al., 2012) where it could potentially be used to both induce phrase tables and score them with little parallel data.

We showed that crosslingual representations are very informative for crosslingual document classification, where they can be used to directly apply a classifier trained on data in one language to test data in another. Classification accuracy is likely to improve if we were to learn these representations specifically for the task, which would require a small change to the learning objective. Applying our framework with the specific goal of building a state-of-the-art classifier is also an interesting future direction.

Acknowledgements

The work was supported by the MMCI Cluster of Excellence and a Google research award.

References

Agirre, E., Alfonseca, E., Hall, K., Kravalova, J., Pasca, M., and Soroa, A. (2009). A study on similarity and relatedness using distributional and wordnet-based approaches. In *Proc. of the Annual Meeting of the North American Association of Computational Linguistics (NAACL)*, pages 19–27, Boulder, Colorado. Association for Computational Linguistics.

Bengio, Y., Ducharme, R., Vincent, P., and Janvin, C. (2003). A neural probabilistic language model. *Journal of Machine Learning Research*, 3:1137–1155.

Callison-Burch, C., Koehn, P., Monz, C., Post, M., Soricut, R., and Specia, L. (2012). Findings of the 2012 workshop on statistical machine translation. In *Proc. of the Workshop on Statistical Machine Translation (WMT)*, pages 10–51, Montréal, Canada. Association for Computational Linguistics.

Cavallanti, G., Cesa-bianchi, N., and Gentile, C. (2010). Linear algorithms for online multitask classification. *Journal of Machine Learning Research*, 11:2901–2934.

Collins, M. (2002). Discriminative training methods for hidden markov models: Theory and experiments with perceptron algorithms. In *Proc. of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Collobert, R. and Weston, J. (2007). Fast semantic extraction using a novel neural network architecture. In *Proc. of the Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 560–567.

Fortuna, B. and Shawe-Taylor, J. (2005). The use of machine translation tools for cross-lingual text mining. In *Proc. of the Workshop on Learning with Multiple Views, 22nd ICML*.

Fouss, F., Pirotte, A., Renders, J., and Saerens, M. (2007). Random-walk computation of similarities between nodes of a graph with application to collaborative recommendation. *Knowledge and Data Engineering, IEEE Transactions on*, 19(3):355–369.

Frederking, R., Hovy, E., and Ide, N. (2001). Special issue on multi-lingual information management. *Computer and the Humanities*, 35.

Glorot, X., Bordes, A., and Bengio, Y. (2011). Domain adaptation for large-scale sentiment classification: A deep learning approach. In Getoor, L. and Scheffer, T., editors, *Proc. of the International Conference on Machine Learning (ICML)*, pages 513–520, New York, NY, USA. ACM.

Huang, E., Socher, R., Manning, C., and Ng, A. (2012). Improving word representations via global context and multiple word prototypes. In *Proc. of the Annual Meeting of the Association for Computational Linguistics (ACL)*.

Klementiev, A., Irvine, A., Callison-Burch, C., and Yarowsky, D. (2012). Toward statistical machine translation without parallel corpora. In *Proc. of the Meeting of the European Association of Computational Linguistics (EACL)*.

Koehn, P. (2005). Europarl: A parallel corpus for statistical machine translation. In *Proc. of the Machine Translation Summit*.

Koehn, P., Hoang, H., Birch, A., Callison-Burch, C., Federico, M., Bertoldi, N., Cowan, B., Shen, W., Moran, C., Zens, R., Dyer, C., Bojar, O., Constantin, A., and Herbst, E. (2007). Moses: Open source toolkit for statistical machine translation. In *Proc. of the ACL-2007 Demo and Poster Sessions*.

Lavrenko, V., Choquette, M., and Croft, W. B. (2002). Cross-lingual relevance models. In *Proc. of International Conference on Research and Development in Information Retrieval (SIGIR)*, pages 175–182.

Lewis, D., Yang, Y., Rose, T., and Li, F. (2004). RCV1: A new benchmark co . in: Sekine, s. and ranchhod, e. named entities: Recognition, classification and use. special issue of lingvisticæ investigationes. 30(1) pp.135-162. ollection for text categorization research. *Journal of Machine Learning Research*, 5:361–397.

Li, S. (2009). *Fast Algorithms for Sparse Matrix Inversion*. PhD thesis, Stanford University.

Mimno, D., Wallach, H. M., Naradowsky, J., Smith, D. A., and McCallum, A. (2009). Polylingual topic models. In *Proc. of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 880–889, Singapore. Association for Computational Linguistics.

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

Rapp, R. (1995). Identifying word translations in non-parallel texts. In *Proc. of the Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 320–322.

Shi, L., Mihalcea, R., and Tian, M. (2010). Cross language text classification by model translation and semi-supervised learning. In *Proc. of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1057–1067, Cambridge, MA. Association for Computational Linguistics.

Socher, R., Huang, E. H., Pennin, J., Ng, A. Y., and Manning, C. D. (2011a). Dynamic pooling and unfolding recursive autoencoders for paraphrase detection. In *Proc. of the Conference on Advances in Neural Information Processing Systems (NIPS)*, pages 801–809.

Socher, R., Pennington, J., Huang, E. H., Ng, A. Y., and Manning, C. D. (2011b). Semi-supervised recursive autoencoders for predicting sentiment distributions. In *Proc. of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 151–161, Stroudsburg, PA, USA. Association for Computational Linguistics.

Steinberger, R. and Pouliquen, B. (2007). Cross-lingual named entity recognition. *Lingvistica Investigationes*, 30:135–162.

Täckström, O., McDonald, R., and Uszkoreit, J. (2012). Cross-lingual word clusters for direct transfer of linguistic structure. In *Proc. of the Annual Meeting of the North American Association of Computational Linguistics (NAACL)*, pages 477–487, Montréal, Canada.

Titov, I. (2011). Domain adaptation by constraining inter-domain variability of latent feature representation. In *Proc. of the Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 62–71, Portland, Oregon, USA. Association for Computational Linguistics.

Turian, J., Ratinov, L., and Bengio, Y. (2010). Word representations: a simple and general method for semi-supervised learning. In *Proc. of the Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 384–394. Association for Computational Linguistics.

van der Plas, L. and Tiedemann, J. (2006). Finding synonyms using automatic word alignment and measures of distributional similarity. In *Proc. of the COLING/ACL Main Conference Poster Sessions*, pages 866–873, Sydney, Australia. Association for Computational Linguistics.

Zhang, D., Mei, Q., and Zhai, C. (2010). Cross-lingual latent topic extraction. In *Proc. of the Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 1128–1137, Uppsala, Sweden. Association for Computational Linguistics.

Exploring Local and Global Semantic Information for Event Pronoun Resolution

Fang Kong[1,2] *Guodong Zhou*[1*]
(1) School of Computer Science and Technology, Soochow University of China, 1 Shizi Street, Suzhou, China 215006
(2) Department of Computer Science, National University of Singapore, 13 Computing Drive, Singapore 117417
kongfang@suda.edu.cn, gdzhou@suda.edu.cn

ABSTRACT

Event anaphora resolution plays a critical role in discourse analysis. This paper focuses on improving event pronoun resolution using both local and global semantic information. In particular, a predicate-argument structure is proposed to represent the local semantic information about an event while the global semantic information is represented by the entity coreference chains related with various arguments in the predicate-argument structure to complement its locality. Evaluation on the OntoNotes English corpus shows the effectiveness of local and global semantic information for event pronoun resolution.

KEYWORDS: event pronoun, semantic information, corefernce resolution, predicate-argument structure

*Corresponding author

Proceedings of COLING 2012: Technical Papers, pages 1475–1488,
COLING 2012, Mumbai, December 2012.

1 Introduction

As one of the most important techniques in discourse analysis, anaphora resolution aims to resolve a given mention to its referred expression in a text and has been a focus of research in Natural Language Processing (NLP) for decades. According to the nature of the referred expression, anaphora resolution can be categorized into entity anaphora resolution and event anaphora resolution. While most studies focus on entity anaphora resolution and have achieved much success recently (e.g. Soon et al. 2001; Ng and Cardie 2002; Ng 2007, 2009; Yang et al.2004, 2006, 2008; Kong et al. 2009,2010), there are only a few studies on event anaphora resolution (Byron, 2002; Pradhan et al., 2007; Chen et al. 2010a, 2010b; Kong and Zhou, 2011).

In this paper, we address event pronoun resolution, the most difficult type of event anaphora resolution due to the least discriminative information that an event pronoun can provide. Here, an event pronoun is a pronoun whose antecedent refers to an event. In particular, we focus on improving event pronoun resolution using both local and global semantic information. For local semantic information, we employ a shallow semantic parser to extract the predicate-argument structure in a sentence to represent an involved event. In order to complement the locality of the predicate-argument structure, we consider the global semantic information via the entity coreference chains related with various arguments in the predicate-argument structure.

The rest of this paper is organized as follows. Section 2 describes the event anaphora resolution task. Section 3 briefly introduces the related work on event anaphora resolution. Section 4 presents our baseline framework which combines various kinds of flat features and a structured parse tree for event pronoun resolution. Section 5 explores both local and global semantic information for event pronoun resolution. Section 6 reports the experimental results. Finally, we conclude our work in Section 7.

2 Task Description

While entity anaphora focuses on mentions of an entity, event anaphora looks into mentions that refer to an event. In this paper, we consider event anaphora resolution. Consider the following examples:

a) In Yemen today, where the ship was ***[attacked]***$_1$, the deliberate, well-organized familiar effort to find out who did *[it]*$_2$, and how ***[it]***$_3$ happened.

b) Two F Tomcats ***[struck]***$_1$ the targets. After ***[today's air strikes]***$_2$, 13 Iraqi soldiers abandoned their posts and surrendered to Kurdish fighters.

c) Yes, it took a while last night to sort out precisely what the court had ***[decided]***$_1$ by such a narrow margin. ***[This]***$_2$ was a stabilizing decision that restored order to a very chaotic situation.

Example (a) shows the importance of event anaphora resolution in understanding the discourse. In example (a), the three mentions in italic and bold font form a chain of event "the ship was attacked". While entity anaphora resolution is capable of linking up mention 2 and mention 3, e.g. using the default proximity preference rule as widely adopted in the literature (Soon et al. 2001), this chain will only contain two pronouns without linking mention 2 to actual event mention 1. Event anaphora resolution provides an essential role to bridge the understanding gap

in such a discourse by linking mention 2 to actual event mention 1. Here, we employ the predicate of a clause to represent an event mention.

Similarly, in example (b), the anaphor of NP "today's air strikes" refers to event "Two F Tomcats struck the targets" while in example (c), the anaphor of pronoun "this" refers to event "what the court had decided by such a narrow margin". Obviously, compared with noun phrases, pronouns carry little information of their own. This indicates the difficulty of event pronoun resolution in event anaphora resolution. Besides, our statistics on the OntoNotes English corpus (Release 3.0) shows that event pronouns occupy about 40% of event anaphors. This indicates the importance of event pronoun resolution in event anaphora resolution.

For better understanding the paper, here we give some related terminologies:

- ***Entity***: an object or a set of objects in one of the semantic categories of interest, referred by one or more coreferential entity mentions in the document.
- ***Entity mention***: a reference to an entity (typically, a noun phrase).
- ***Event trigger***: the key word that most clearly expresses the occurrence of an event. In this paper, as mostly adopted in the literature, we take the main predicate (either verbal or nominal) of a clause as the event trigger to represent the corresponding event.
- ***Event arguments***: the entity mentions involved in an event.
- ***Event mention***: a clause within which an event is described, including event trigger and event arguments. Although some event pronouns can actually refer to a paragraph or larger chunks of texts, in this paper we only consider the cases taking clauses as antecedents.

3 Related Work

In comparison with entity anaphora resolution, there are few linguistic studies on event anaphora (e.g. Asher, 1993) and very initial explorations on event anaphora resolution. It was only until recently, with the increasing interest in discourse analysis, event anaphora resolution has begun to draw more and more attention for the natural language processing community. While some of them focus on hand-crafted constraints to resolve event anaphora of normally limited kinds of predicates (e.g. Byron, 2002), most of previous studies adopt a learning-based framework (e.g. Muller, 2007; Pradhan et al., 2007; Chen et al. 2010a, 2010b; Kong and Zhou, 2011).

As a representative to linguistic studies on event anaphora, Asher (1993) proposed a discourse representation theory to resolve the references to events. However, no computational system was proposed in his work.

As a representative of using hand-crafted knowledge to resolve specific kinds of predicates, Byron (2002) proposed a semantic filter as a complement to salience calculation in resolving event pronouns. However, since the semantic filter was constructed by using a set of hand-crafted constraints on some specific domains, this approach is not suitable for general event pronoun resolution.

Among learning-based methods to event anaphora resolution, Chen et al. (2010a) explored various kinds of positional, lexical and syntactic features for event pronoun resolution, which turned out quite different from entity pronoun resolution. Besides, they studied the importance of structured syntactic information by incorporating it into event pronoun resolution via a composite kernel. Finally, they explored the incorporation of negative instances from non-event anaphoric

pronouns, the fine-tuning of the SVM model and the employment of the twin-candidate model (Yang et al. 2003) in event pronoun resolution. Chen et al. (2010b) extended their previous work from event pronoun resolution to general event anaphora resolution by considering other types of event anaphors. Kong and Zhou (2011) proposed a new tree expansion scheme to automatically determine a proper parse tree structure for event pronoun resolution by considering various kinds of competitive information related with the anaphor and the antecedent candidate and achieved a much better performance on the OntoNotes English corpus than Chen et al (2010a).

Besides, there are some studies which integrate event anaphora resolution with entity anaphora resolution. For example, Pradhan et al. (2007) proposed a unified event and entity anaphora resolution framework based on a set of widely-used features which have been proven to be effective for entity anaphora resolution. Evaluation on the OntoNotes English corpus shows that their unified framework achieved the performance of 51.2 in F1-measure on overall entity and event anaphora resolution. However, they did not report the performance of their unified framework on event anaphora resolution. Alternatively, Muller (2007) constructed a logistic regression model to resolve event and entity pronouns together. For event pronoun resolution, he achieved 11.94 in F1-measure and found that the types of information effective for event pronoun resolution were very different from those for entity pronoun resolution. From this aspect, it seems better to independently explore event anaphora resolution first and then explore its possible integration (e.g. joint learning) with entity anaphora resolution.

In this paper, we focus on improving event pronoun resolution by exploring both local and global semantic information.

4 Baseline System

Our event pronoun resolution framework adopts the common learning-based one for entity anaphora resolution, as described by Soon et al. (2001). Specially, the way generating instances during training and testing procedures of event pronoun resolution is similar to Kong and Zhou (2011).

4.1 Flat Features

For entity pronoun resolution, Yang et al. (2004, 2005, 2006) explored various kinds of syntactic and semantic features to describe the information related with the antecedent candidate and the anaphor from their own and the relationship between them. However, few of these features can be adopted in event pronoun resolution. On one hand, since the antecedent candidate is an event trigger and the anaphor is a pronoun, both carry little obvious information about their own. On the other hand, the event anaphor and candidate pair in event pronoun resolution consists of a predicate and a pronoun. The difference in syntactic categories introduces extra difficulties. The features, such as number agreement, gender agreement, name alias, string matching and head matching, which have been proven to be effective for entity pronoun resolution, will no longer function here. Instead, we employ a list of flat features as shown in Table 1, inspired by Chen et al. (2010a).

In principle, the features in Table 1 can be grouped into three categories:

1) *Positional features*: the intuition is that the antecedent of an event pronoun should be close to each other. In particular, different kinds of distances between the anaphor and

the antecedent candidate are explored, i.e. over sentence, word, pronoun, predicate and main predicate.

2) *Grammatical features*: mainly used to describe the grammatical roles of the anaphor and the antecedent candidate.
3) *Similarity feature*: There is no doubt that semantic information is important for event pronoun resolution. However, since both event pronouns and event triggers carry little obvious semantics of their own, the context similarity is employed to measure the semantic compatibility between the anaphor and its antecedent candidate. In our baseline system, the similarity is calculated based on a list of nearby 10 contextual words (including previous 5 words and following 5 words) with proper stemming using the Porter stemmer and stop words (such as "in", "the" and etc.) filtered out.

	Features	Description
Positional Features	SenDist	Sentence distance between event anaphor and antecedent candidate
	WordDist	Word distance between event anaphor and antecedent candidate
	PredDist	Number of predicates between event anaphor and antecedent candidate
	PronDist	Number of pronouns between event anaphor and antecedent candidate
	MPredDist	Number of main predicate between event anaphor and antecedent candidate
Grammatical Features	isAnaInMC	Whether event anaphor occurs in main clause
	isAnaSub	Whether event anaphor is at subject position
	isPredInMC	Whether antecedent candidate occurs in main clause
	isMPred	Whether antecedent candidate is main predicate
Similarity Feature	ContextSim	Similarity between event anaphor's context and antecedent candidate's context

TABLE 1 – FLAT FEATURES FOR EVENT PRONOUN RESOLUTION

4.2 Structured Parse Tree

Besides various kinds of flat features described above, we also explore structured syntactic information via a parse tree structure. The commonly-used syntactic knowledge for anaphora resolution, such as the governing relations, can be directly described by the parse tree structure. Other syntactic knowledge that may be helpful for anaphora resolution could also be implicitly represented in the parse tree structure. Furthermore, tree kernel-based methods have been explored in entity anaphora resolution and achieved comparable performance with the dominated feature-based methods (Yang et al. 2006; Zhou et al. 2008). For structured syntactic information, we adopt the Dynamic Competitive Tree, as proposed in Kong and Zhou (2011), which takes the related competitive information, such as the event pronoun predicate (i.e. the predicate of the event pronoun), event antecedent competitors and event pronoun competitors between the anaphor and the considered antecedent candidate into consideration. For more details, please refer to Kong and Zhou (2011).

4.3 Polynomial Kernel, Tree Kernel and Composite Kernel

For two vectors of flat features, we compute their similarity using a polynomial kernel, while given any parse tree structure, e.g. the one as described above, the similarity between two parse trees is calculated using a convolution tree kernel. For more details about this kernel, please refer to Collins and Duffy (2002) and Moschitti (2004).

In order to combine flat features and structured parse trees, a linear-interpolated composite kernel is adopted in this paper:

$$K_{comp}(x_1, x_2) = \frac{K_{tree}(x1, x2)}{|K_{tree}(x1, x2)|} + \frac{K_{flat}(x1, x2)}{|K_{flat}(x1, x2)|}$$

For simplicity, this paper equally weights the convolution tree kernel K_{tree} for the parse tree structure and the polynomial kernel K_{flat} (d=2) for flat features with proper normalization.

5 Incorporating Semantic Information

It is well proven that the semantic compatibility between the anaphor and the antecedent candidate is important for entity anaphora resolution. For event pronoun resolution, since the anaphor is a pronoun and the antecedent candidate is an event trigger, both carry little obvious semantic information about themselves. Therefore, it is more difficult to measure the semantic compatibility between the anaphor and the antecedent candidate in event pronoun resolution. A possible way to measure it is to explore the contexts where the event pronoun and the antecedent candidate occur.

In our baseline system, we use a bag-of-words method to represent the contexts of the anaphor and the candidate. However, such a bag-of-words method suffers from the dilemma between the noise and the necessary information covered in a context window. In order to resolve this problem, we propose a predicate-argument structure representation to capture the local semantic information related with an event. Besides, we explore the global semantic information via entity coreference chains related with various arguments of the predicate-argument structure to complement its locality.

5.1 Local Semantic Information

Obviously, various arguments closely related with an event trigger contain necessary semantic information for event pronoun resolution. Therefore, it is reasonable to represent an event mention using the event trigger and corresponding event arguments. Since the event trigger is normally the predicate of a clause and event arguments correspond to the arguments driven by the predicate of a clause, we employ a shallow semantic parser to extract the predicate-argument structure of a clause as the representation of the local semantic information of the event. Especially for an event pronoun, we retrieve its governing predicate and related arguments as its local representation.

Figure1 shows the algorithm for computing the semantic compatibility between an event pronoun and an antecedent candidate. In particular, only those core arguments as defined in the Propbank (Palmer et al. 2005) are included in the computation. Consider following example:

a) ***Both President Gore and President Bush$_1$*** have held a flurry of news conferences in recent weeks and with each one ***they$_1$*** have ***[increased]*** *the* number of Stars and Stripes ***they$_1$*** use as a backdrop. I think ***they$_1$*** think ***[it]*** makes ***them$_1$*** look patriotic and presidential.

which includes an inter-sentence event anaphora with pronoun "it" referring to event "they have increased the number of Stars and Stripes". For this pair of event pronoun and antecedent, the algorithm in Figure1 returns {makes, it, them} and {increased, they, the number of Stars and Stripes} as AnaSet and CandSet, respectively.

Algorithm:

computing the semantic compatibility between an anaphor and an antecedent candidate

Input:

an anaphor: current event anaphor, a pronoun
an antecedent candidate: an event trigger, a predicate

Steps:

1) Initialize Score, CandSet and AnaSet
2) Use a semantic role labeling (SRL) toolkit to get all the core arguments of the antecedent candidate (i.e., event arguments) and add the antecedent candidate (i.e., the event trigger) and all the core arguments to CandSet (except pronouns).
3) Get the governing predicate of the anaphor and use a SRL toolkit to get all the core arguments of the predicate. Add the governing predicate of the anaphor and all the core arguments to AnaSet (except pronouns).
4) For every element pair from CandSet (candi) and AnaSet (anaj), compute the similarity between them using WordNet as described in Satanjeev and Ted (2002). If the similarity is larger than a threshold (0.5 in this paper), increase Score by 1.
5) Return Score/sqrt(|CandSet|*|AnaSet|) as the semantic compatibility between the given anaphor and the given antecedent candidate

FIGURE1– ALGORITHM FOR COMPUTING THE SEMANTIC SIMILARITY BETWEEN AN ANAPHOR AND AN ANTECEDENT CANDIDATE

5.2 Global Semantic Information

Obviously, the algorithm in Figure1 can only retrieve local descriptions of an event mention. It may be difficult to correctly measure the global semantic compatibility between the anaphor and the antecedent candidate using the predicate-argument structure due to its locality, e.g. considering example (d) due to the occurrence of various pronouns.

In order to complement the locality of the predicate-argument structure in representing the anaphor and the antecedent candidate, we further explore the global semantic information via entity coreference chains related with various arguments for event pronoun resolution.

The basic idea is that, when measuring the semantic compatibility between the anaphor and the antecedent candidate, we not only consider the event trigger and involved arguments, but also include different entity mentions (except pronouns) related with those involved arguments. Consider CandSet={increased, they, the number of Stars and Stripes} in the above example. Although we can retrieve little semantic information from argument "they" itself, we can find that it actually refers to "Both President Gore and President Bush" via entity anaphora resolution.

Similarly, we can find the referred expressions of other pronouns in CandSet and AnaSet. In this way, CandSet and AnaSet can be better represented.

6 Evaluation and Discussion

This section systematically evaluates the performance of our event pronoun resolution framework on the OntoNotes English corpus (Release 3.0).

6.1 Experimental Setting

The OntoNotes English corpus (Release 3.0) contains 300K words of English newswire data (from the Wall Street Journal) and 200K words of English broadcast news data (from ABC, CNN, NBC, Public Radio International and Voice of America). Table 2 shows the statistics of the corpus. From Table 2 we can find that about 9% of coreference chains are event related. Among 3550 event pronoun candidates (i.e. all the occurrences of "this", "that" and "it", which function as pronoun), 504 are event pronouns, accounting for about 14%. This indicates the difficulty of identifying event pronouns.

Category	Num
Coreference chains	8154
Coreference chains related with events	737
Event pronoun candidates	3550
Event pronouns	504

TABLE 2 – STATISTICS ON ONTONOTES 3.0 ENGLISH CORPUS

System	P(%)	R(%)	F
Polynomial kernel(flat features)	34.84	53.08	42.07
Tree kernel(structured)	47.06	65.71	54.84
Composite kernel(flat+structured)	49.78	69.17	57.89

TABLE 3 – PERFORMANCE OF THE BASELINE SYSTEM

For preparation, all the documents in the corpus are pre-processed automatically using a pipeline of NLP components. In addition, the corpus is parsed using the Charniak parser, and a state-of-the-art semantic role labelling (SRL) toolkit as proposed by Li et al. (2009) is employed to extract the predicate-argument structure (i.e. various arguments of a predicate)[1]. Finally, we use the SVM-light toolkit (Joakim, 1998)[2] with the convolution tree kernel SVMlight–TK (Moschitti, 2004)[3] for computing the similarity between two parse trees, and the polynomial kernel (d=2) for computing the similarity between two vectors of flat features, with learning parameters same as Chen et al. (2010a). For performance evaluation, we report the performance of event pronoun resolution with 10-fold cross validation in terms of recall, precision, and F1-measure.

[1] In this paper, we only consider verbal predicates, since 97.3% of events in the OntoNotes English corpus (Release 3.0) are triggered by verbal predicates. Besides, only those core arguments as defined in the Propbank are explored in this paper. For reference, the toolkit developed by Li et al. (2009) achieved the performance of 81.8 in F1-measure on the CoNLL'2005 version of the Propbank.

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

[3] http://ai-nlp.info.uniroma2.it/moschitti/

6.2 Experimental Results

6.2.1 Performance of baseline system

Table 3 shows the performance of our baseline system. It shows that the polynomial kernel with the flat features yields 42.07 in F1-measure while the convolution tree kernel with the parse tree structure achieves a much better performance of 54.84 in F1-measure due to direct modeling of commonly used syntactic knowledge and implicit including of other knowledge helpful for event pronoun resolution. It also shows that the flat features and the parse tree structure are quite complementary that their combination via a simple composite kernel achieves 57.89 in F1-measure. Although the employed dynamic competitive tree may lose some important contextual information, it prunes out potential noise as much as possible. At the same time, the flat features used in our baseline system mainly describe positional and grammatical information. This suggests the complementary nature of the flat features and the parse tree structure, which is justified by the effective use of the composite kernel. In all the following experiments, we only report the performance employing the composite kernel.

6.2.2 Contribution of local semantic information

Table 4 shows the contribution of local semantic information on the composite kernel. It shows that the local semantic information via the predicate-argument structure can significantly improve the performance on the composite kernel by 1.7 in F1-measure. This justifies the usefulness of the predicate-argument structure in representing the local semantic information of the anaphor and the antecedent candidate. It also shows that the inclusion of A0 (i.e. agent) improves the performance by 1.29 in F1-measure and further inclusion of A1 (i.e. recipient) contributes 0.36 in F1-measure while the effectiveness of other kinds of arguments is very limited due to their less number and their less possibility of being referred in a text.

System	P(%)	R(%)	F
Flat+Structured	49.78	69.17	57.89
+A0	53.04	66.92	59.18(+1.29)
++A1	53.59	66.98	59.54(+0.36)
+++Others	53.64	67.02	59.59(+0.05)

TABLE 4 – INCREMENTAL CONTRIBUTION OF DIFFERENT KINDS OF LOCAL SEMANTIC INFORMATION ON THE COMPOSITE KERNEL [4]

6.2.3 Contribution of global semantic information

While the global semantic information via entity coreference chains can complement the locality of the predicate-argument structure, entity anaphora resolution itself is a difficult task. Obviously, the effectiveness of the global semantic information will largely depend on the performance of entity anaphora resolution.

[4] Significance tests are conducted between each of them and the previous one. The *p*-values are all smaller than 0.01.

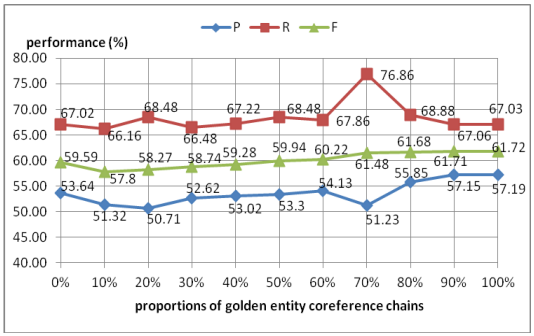


FIGURE 2– CONTRIBUTION OF GLOBAL SEMANTIC INFORMATION BY RANDOMLY CHOOSING DIFFERENT PROPORTIONS OF GOLDEN ENTITY COREFERENCE CHAINS

Figure 2 shows the contribution of global semantic information by randomly choosing different proportions of golden entity coreference chains. From Figure 2, we can find that:

1) Only considering a small proportion of even golden entity coreference chains (<50%) may harm the performance of event pronoun resolution. That is to say, only a small proportion of even golden entity coreference chains cannot complement the locality of the predicate-argument structure. In contrast, it may introduce too much uneven distribution across different events and thus harm the performance.
2) For golden entity coreference chains, including at least 50% begins to contribute.
3) When the used proportion of golden entity chains reaches a threshold (about 80%), the performance of event pronoun resolution stabilizes.

Systems	P(%)	R(%)	F
Soon et al. (2001) (duplicated)	61.5	45.9	52.57
Kong et al. (2009) (duplicated)	73.5	54.2	62.39

TABLE 5 – PERFORMANCE OF ENTITY ANAPHORA RESOLUTION

System	P(%)	R(%)	F
Flat+Structured+Local Semantic	53.64	67.02	59.59
+Global Semantic (Soon et al 2001)	53.6	67.36	59.70
+Global Semantic (Kong et al. 2009)	54.65	68.6	60.84

TABLE 6 – CONTRIBUTION OF AUTO ENTITY COREFERENCE CHAINS

In order to measure the effectiveness of global semantic information in practical environment, i.e. using automatic entity anaphora resolution, we duplicate two entity anaphora resolution systems with different levels of performance: the one proposed by Soon et al. (2001) and the other proposed by Kong et al. (2009). Table 5 shows the performance of these two duplicated systems on the OntoNote English corpus. Table 6 shows the contribution of global semantic information via entity coreference chains returned by the two automatic anaphora resolution systems. It shows

that the effectiveness of global semantic information largely depends on the performance of an entity anaphora resolution system. Using the duplicated system proposed by Soon et al.(2001), we can only get a performance improvement of only 0.11 in F1-measure for event pronoun resolution while applying the duplicated system proposed by Kong et al. (2009) improve the performance by 1.25 in F1-measure. That is to say, the state-of-the-art entity anaphora resolution system can improve the performance of event pronoun resolution by filling the gap by about 50% (60.84-59.59 vs. 61.72-59.59). While most of the contribution of golden entity chains comes from gain in precision, the contribution of automatic entity chains comes from gain in both precision and recall.

6.2.4 Comparison with the State-of-the-Art

For comparison, Table 7 illustrates the performance of the state-of-the-art event pronoun resolution system developed by Chen et al. (2010a) using different schemes. From Table 7, we can find that Chen et al (2010a) achieved the performance of 40.6 in F1-measure via a feature-based method, and the best performance of 44.4 in F1-measure using the min-expansion tree via a tree kernel-based method. They further studied different ways of combining flat features and a parse tree structure to improve the performance and achieved the best performance of 47.2 in F1-measure when combining flat features with the simple-expansion tree structure. In our study, our feature-based system achieves 42.07 in F1-measure, and our tree kernel-based method with the dynamic competitive tree achieves the performance of 54.84 in F1-measure. By combining the flat features with the structured parse tree via a composite kernel, our system achieved the performance of 57.89 in F1-measure. The much better performance of our baseline system is mainly due to two reasons: 1) our better preprocessing in filtering out unnecessary negative instances by employing a set of constraints as described in Byron (2002). In Chen et al. (2010a), each event pronoun will generate 6.93 candidates while the number in our system is reduced to about 3; 2) employing the more effective structured tree span.

System	P(%)	R(%)	F
Flat	40.6	40.6	40.6
Min-Expansion	35.5	59.6	44.4
Simple-Expansion +Flat	42.3	53.4	47.2
+Negative Instances w/ Sampling	59.9	50.6	54.9
++SVM Fine-tuning	65.2	49.2	56.1
+++Twin-Candidate Modelling	62.6	54.0	57.9

TABLE 7 – PERFORMANCE OF CHEN ET AL. (2010A) ON EVENT PRONOUN RESOLUTION

Besides, Chen et al. (2010a) looked into the incorporation of negative instances from non-event anaphoric pronouns and achieved the best performance of 54.9 in F1-measure. They further improved the performance by keeping certain training data as the development data to help SVM select a more accurate hyper plane and achieved the performance of 56.1 in F1-measure. Finally, they proceeded to apply the twin-candidate model as proposed in Yang et al. (2003) to event pronoun resolution and achieved the performance of 57.9 in F1-measure. After employing so many strategies, Chen et al. (2010a) achieved the comparable performance with our baseline system. This justifies the strength of our baseline system.

We further improve the performance of event pronoun resolution by combining local and global semantic information via the predicate-argument structure and entity coreference chains and achieve the outstanding performance of 60.84 in F1-measure using automatic semantic role labelling and entity anaphora resolution.

Conclusion and Further Work

This paper studies the impact of both local and global semantic information for event pronoun resolution. In particular, a predicate-argument structure is proposed to represent the local semantic information related with an event while the global semantic information via entity coreference chains is further incorporated to complement the locality of the predicate-argument structure. Experimental results show that both the local and global semantic information are very effective for event pronoun resolution. We also study the influence of the performance of entity anaphora resolution on event pronoun resolution.

For further work, we will explore more structured syntactic information and semantic information in event anaphora resolution. In addition, we will study joint learning of entity anaphora resolution and event anaphora resolution.

Acknowledgment

This research was supported by Projects 61003153, 61273320 and 61272257 under the National Natural Science Foundation of China, Project 2012AA011102 under the National 863 Program of China, Project 11KJA520003 under the Natural Science Major Fundamental Research Program of the Jiangsu Higher Education Institutions, Project SYG201112 under the Applied Basic Research of Suzhou.

References

Asher N. 1993. Reference to Abstract Objects in Discourse. *Kluwer Academic Publisher*.

Byron D. 2002. Resolving Pronominal Reference to Abstract Entities. *ACL'2002*

Banerjee S. and Pedersen T. 2002. An Adapted Lesk Algorithm for Word Sense Disambiguation Using WordNet .*CICLing'2002*.

Chen B., Su J. and Tan C.L. 2010(a). A Twin-Candidate Based Approach for Event Pronoun Resolution using Composite Kernel. *COLING'2010*

Chen B., Su J. and Tan C.L. 2010(b). Resolving Event Noun Phrases to Their Verbal Mentions. *EMNLP'2010*

Collins M. and Duffy N. 2001. Covolution kernels for natural language. *NIPS' 2001*

Kong F., Zhou G.D. and Zhu Q.M. 2009 Employing the Centering Theory in Pronoun Resolution from the Semantic Perspective. *EMNLP'2009*

Kong F., Li Y.C., Zhou G.D. and Zhu Q.M. 2009. Exploring Syntactic Features for Pronoun Resolution Using Context-Sensitive Convolution Tree Kernel. *IALP'2009*

Kong F., Zhou G.D., Qian L.H. and Zhu Q.M. 2010. Dependency-driven Anaphoricity

Determination for Coreference Resolution. *COLING'2010*

Kong F. and Zhou G.D. 2010. A Tree Kernel-based Unified Framework for Chinese Zero Anaphora Resolution. *EMNLP'2010*

Kong F. and Zhou G.D. 2011. Improve Tree Kernel-Based Event Pronoun Resolution with Competitive Information. *IJCAI'2011*

Laura H. and Constantin O. 2009. Do coreferential arguments make event mentions coreferential?. *DAARC'2009*

Li J.H, Zhou G.D., Zhao H., Zhu Q.M. and Qian P.D. 2009. Improving nominal SRL in Chinese language with verbal SRL information and automatic predicate recognition. *EMNLP '2009*

Moschitti A. 2004. A Study on Convolution Kernels for Shallow Semantic Parsing, *ACL'2004*

Muller C. 2007. Resoving it, this, and that in unrestricted multi-party dialog. *ACL'2007*

Ng V. and Cardie C. 2002. Improving machine learning approaches to coreference resolution. *ACL'2002*

Ng V. 2009. Graph-cut based anaphoricity determination for coreference resolution. *NAACL'2009*

Palmer M., Gildea D. and Kingsbury P. 2005. The Proposition Bank: An Annotated Corpus of Semantic Roles. *Computational Linguistics 31(1):71-105*

Pradhan S., Ramshaw L., Weischedel R., Macbride J. and Micciulla L. 2007. Unrestricted Coreference: Identifying Entities and Events in OntoNotes. *ICSC'2007*

Satanjeev B. and Ted P. 2002. An Adapted Lesk Algorithm for Word Sense Disambiguation Using WordNet .*CICLing'2002*

Soon W.M., Ng H.T. and Lim D. 2001. A machine learning approach to coreference resolution of noun phrase. *Computational Linguistics, 27(4):521-544.*

Yang X.F., Su J., Zhou G.D. and Tan C.L. 2004. Improving pronoun resolution by incorporating coreferential information of candidates. *ACL'2004*

Yang X.F., Su J. and Chew C.L., 2006. Kernel-based pronoun resolution with structured syntactic knowledge. *COLING-ACL'2006*

Yang X.F., Su J. and Tan C.L. 2008. A Twin-Candidate Model for Learning-Based Anaphora Resolution. *Computational Linguistics 34(3):327-356*

Zhou G.D., Kong F. and Zhu Q.M. 2008. Context-sensitive convolution tree kernel for pronoun resolution. *IJCNLP'2008*

Zhou G.D. and Kong F. 2009. Global Learning of Noun Phrase Anaphoricity in Coreference Resolution via Label Propagetion. *EMNLP'2009*

Semantic Processing of Compounds in Indian Languages

Amba Kulkarni[1] Soma Paul[2] Malhar Kulkarni[3]
Anil Kumar[1] Nitesh Surtani[2]

1. Department of Sanskrit Studies, University of Hyderabad
2. International Institute of Information Technology, Hyderabad
3. School of Humanities and Social Sciences, I.I.T. Bombay

apksh@uohyd.ernet.in, soma@iiit.ac.in, malharku@gmail.com,
anil.lalit22@gmail.com, nitesh.surtani0606@gmail.com

ABSTRACT
Compounds occur very frequently in Indian Languages. There are no strict orthographic conventions for compounds in modern Indian Languages. In this paper, Sanskrit compounding system is examined thoroughly and the insight gained from the Sanskrit grammar is applied for the analysis of compounds in Hindi and Marathi. It is interesting to note that compounding in Hindi deviates from that in Sanskrit in two aspects. The data analysed for Hindi does not contain any instance of Bahuvrīhi (exo-centric) compound. Second, Hindi data presents many cases where quite a lot of compounds require a verb as well as vibhakti(a case marker) for its paraphrasing. Compounds requiring a verb for paraphrasing are termed as madhyama-pada-lopī in Sanskrit, and they are found to be rare in Sanskrit.

KEYWORDS: Multi Word Expression, Compounds, Pāṇini, Aṣṭādhyāyī, semantics.

Proceedings of COLING 2012: Technical Papers, pages 1489–1502,
COLING 2012, Mumbai, December 2012.

1 Introduction

Noun compounds represent a linguistic device of encrypting information that makes their analysis a challenging NLP task. For example, consider a compound 'ballpoint pen'. It is made up of two nouns: a modifier 'ballpoint' and a head 'pen'. The modifier is again a compound that is composed of two nouns 'ball' and 'point'. The relation between 'ball' and 'point' and in turn between 'ballpoint' and 'pen' is not encoded anywhere on the surface although a user of the language can decodify the meaning correctly. The study of compounds involves two major tasks: 1) automatic identification and extraction of compounds from natural language texts and 2) syntactic and semantic analysis of compounds. The task of identification of compounds becomes significant because of orthographic vagaries. Orthographic conventions for writing compounds may vary from language to language and even within the same language. For example, in Sanskrit a compound is a single word, while in English (as exemplified above) as well as in modern Indian Languages (ILs) we find the following conventions of writing: components written with or without a space and components separated by a hyphen. When compounds are written without a space, the adjoining phonemes undergo euphonic changes as in *gaṅgā-udaka* changing to *gaṅgodaka*. Analysis of compounds primarily involves the expansion of these syntactically condensed constructs with an aim to unfold the meaning of the constructions (Butnariu and Veale, 2008; Girju et al., 2007; Kim and Baldwin, 2006; Kumar, 2012; Nakov, 2008; Nastase and Szpakowicz, 2009; Séaghdha and Copestake, 2007). Semantic analysis of compound is significant for various NLP applications including machine translation, information extraction and so on. Here we discuss one example to show how semantic analysis helps in machine translation. For example, let us consider the English compounds *cancer death* and *room temperature*. Lexical substitution of components of these compounds into Hindi would produce the following result: *kainsara mauta* and *kamarā tāpamāna* which are not legitimate constructions in Hindi. However, the semantic paraphrase of the two compounds, namely, *death from cancer* and *temperature of room* will be helpful for achieving the correct translations of the compounds: *kainsara se* (or *ke kāraṇa*) *mauta* and *kamare kā tāpamāna*. Currently there exist two different approaches in Computational Linguistics to deal with this phenomenon (Paul et al., 2010). They are (a) Labeling the semantics of compound with a set of abstract relations (Girju et al., 2003) and (b) Paraphrasing the compound in terms of syntactic constructs. Paraphrasing is again done in three ways: (i) with prepositions (*war story* vs *story about war*) (Lauer, 1995) (ii) with verb+preposition nexus (*war story* vs *story pertaining to war*, *noise pollution* vs *pollution caused by noise* (Finin, 1980) (iii) with Copula (*tuna fish* vs *fish that is tuna*) (Vanderwende, 1995).

Detailed study of Sanskrit compound processing had been taken up recently (Kumar, 2012), and the insights gained there were found useful for processing the compounds in ILs. After looking at the features of Sanskrit as described in the Sanskrit grammar, in the third section we describe the automatic Sanskrit compound processor, followed by the insights we gained from this processor to identify the compound tags and also the semantic categories necessary to carry out the compound analysis automatically. The fourth section discusses as a case study, use of these tags for Hindi and Marathi compound types. Conclusion follows in the fifth section.

2 Sanskrit Compounds

Pāṇini (500 BCE approximately) has described the process of compound formation in Sanskrit in his grammar called Aṣṭādhyāyī. He used the term Samāsa for Compound. The word Samāsa literally means "Throwing out together" which in the context means "throwing out the words

together". This concept implies that words are thrown out of mouth by human beings in syntactically related structures. There are three main features of a Compound: 1. One word (*aikapadya*), 2. One meaning (*aikārthya*) and 3. One accent (*aikasvarya*). According to Pāṇini, two words within a sentence can form a compound if and only if there is syntactic compatibility (sāmarthya) amongst them. Mere adjacency of words does not allow them to get compounded. The words should be first syntactically related and further should possess the quality of being used by the native speakers as one unit. Further, what distinguishes a compound from a non compounded word group within a sentence are various morphological as well as syntactical features such as a) loss of case, b) absence of intervention of other words, c) no possibility of relation of a non-head word within a compound with another word in the sentence, d) absence of expression of number of the first component.

There are broadly speaking four types of compounds in Sanskrit: 1. Avyayībhāva, 2. Tatpuruṣa, 3. Dvandva and 4. Bahuvrīhi. Semantically, Avyayībhāva and Tatpuruṣa are endocentric compounds with the head typically to the left and right respectively. Dvandva is a copulative compound while Bahuvrīhi is an exocentric compound. Many of the compounds are compositional and hence can be generated with the help of a rule base. However, there are some compounds which are non-compositional and they are treated separately in Pāṇini's grammar. These compounds are called nitya samāsa (obligatory compounds), that is they are always used in compounding form. Such compounds either can not be paraphrased at all (*Avigraha*) or involve extra words other than the components for their paraphrase (*Asvapadavigraha*). Following two examples will illustrate this point:

i) *aśvakarṇa* (name of a medicinal plant) is a compound made up of two words, *aśva* 'a horse' and *karṇa* 'an ear'. When both these words are joined together, they result into a compound, *aśvakarṇa*; but, this compound has no traces of the meanings of it's components. Therefore, this compound can not be paraphrased in terms of its components.

ii) *Kākapeyā* (meaning: a river which contains water potable only by crows). This compound is made up of two components, namely, *kāka* 'a crow' and *peya* 'potable'. As we see, there is an additional semantic element of censure which goes to make this compound. This additional meaning is obviously not a part of the component meanings.

The meaning of compounds may or may not be compositional. Based on the discussions in traditional Sanskrit grammar sources, we see a spectrum of compositionality as illustrated below (Shastri, 2006).
i) purely compositional (*sambaddhārtha*)
ex: *rāja-puruṣaḥ*
gloss: King - man
meaning: King's man

ii) compositionality with fixity of expression (*samprekṣita*)
ex: *khaṭvā-āruḍhaḥ*
gloss: Bed - one who climbs
meaning: One who climbs the bed without completing the education

iii) Non compositionality (with some predictability) (*samgatārtha*)
ex: *citra-gu*
gloss: colorful - cow
meaning: One who has a colorful cow

iv) Non compositionality (*samsṛṣṭārtha*)
ex: *aṣva-karṇa*
gloss: horse - ear
meaning: Name of a medicinal plant.

In Sanskrit compounds are always written without any space in between. But modern Indian languages do allow space in between. Therefore, this spectrum of meanings makes it difficult to decide whether a group of words written with space in between is a compound or not, making the identification of a compound a challenging task for these languages.

3 Sanskrit Compound Processing

Sanskrit is rich in compound formation. Almost every fifth or sixth word in a randomly chosen Sanskrit text is a compound. Compound formation being very productive, we can not list all the compounds in a dictionary. An automatic compound processor was developed (Kumar, 2012) as a part of Computational Toolkit for Sanskrit. This compound processor provides a general architecture for processing compounds.

Analysing a Sanskrit compound involves
i) Segmentation,
ii) Deciding the constituency structure,
iii) Identification of relations between the constituents, and
iv) paraphrasing it.

3.1 Segmentation

In Sanskrit a compound is always written without any space. Moreover, the phonemes of the adjoining components necessarily undergo euphonic changes. Splitting involves reversing these euphonic changes. For instance, the compound *gaṅgodaka* is segmented as *gaṅgā-udaka*. It is possible that a word is ambiguous leading to multiple possible splitting. In Sanskrit the authors have taken advantage of this ambiguity which resulted in many texts with pun. The splitter should be able to produce all possible splits and also rank them if possible. A splitter needs sandhi rules and also a morphological analyser to validate the splits. Two different methods have been followed for building a Sanskrit splitter (Mittal, 2010). In the first approach FST built for morphological analyser is augmented with the sandhi rules (Huet, 2009). In the second approach a given string is split in all possible ways following the sandhi rules, and then the splits are validated through the morphological analyser. This is closer to the GENerate-CONstrain-EVALuate model of Optimality theory (Prince and Smolensky, 1993). The sandhi rules split the given string into all possible ways, then constraints, such as every component of the split should be a valid morph, are applied, and finally the possible splits are ranked using the language and split model. The results of this splitter are quite good, with 93% of cases the first split is correct. This method, though sounds good, practically ends up generating thousands of splits 90% of which are not validated morphologically. Thus this method is computationally less efficient. On the other hand a splitter built by augmenting the FST with sandhi rules is computationally very efficient, since it splits the string only if it is morphologically valid thereby avoiding unnecessary splits. If this FST is further augmented with a proper model for sandhi rules and the lexicon, better results are expected.

3.2 Constituency Parser

Constituency parser takes an output of the segmenter and produces a binary tree showing the syntactic composition of a compound corresponding to each of the possible segmentations. Each of these compositions shows the possible ways various segments can be grouped. To illustrate various possible parses that result from a single segmentation, consider the segmentation a-b-c of a compound. A compound being binary[1], the three components a-b-c may be grouped in two ways as <*a*-< *b*-*c*>> or <<*a*-*b*>-*c*>. Only one of the ways of grouping may be correct in a given context (unless the text has intended pun) as illustrated by the following two examples. Parse of *three-meter-wide* is <<*three-meter*>-*wide*>, and that of *iron water pump* is <*iron*-<*water-pump*>>. The number of possible parses increase exponentially as the number of components increase. The problem of constituency parsing is similar to the problem of completely parenthesizing *n+1* factors in all possible ways. Thus the total possible ways of parsing a compound with $n+1$ constituents is equal to a *Catalan number,* C_n (Huet, 2009). In the absence of any morpheme marking the relation between the components, the constituency structure is governed by the compatibility of meanings of the components involved. Hence to decide the constituent structure, a semantically rich lexicon is needed. In the absence of any such lexicon, the statistical properties of the manually tagged corpus were used (Kulkarni and Kumar, 2011) to decide the constituency structure. For compounds with 3 components, it had a F-measure of 93.66, in case of compounds with 4 components, the F-measure dropped to 65.4. The corpus had compounds with as many as 10 components. On average in 86.5% of cases, the machine could produce the correct parse. Pāṇini has also provided certain rules with morphological constraints on the final component[2], or certain group of words as an initial component[3]. These rules help in prioritising the grouping. These rules, though are written for Sanskrit, hold good across languages. Here is an example of a rule which deals with three components. The sūtra *(diksaṅkhye) taddhitārthottarapadasamāhāre ca (2.1.50)* says that in case of a compound with three components with number or direction indicating word as the first component, the first two components combine first. This holds good for other languages as well. For example, *one day cricket match* is <<*one-day*>-<*cricket-match* >> and *South Indian Association* is <<*South-Indian*>-*Association*>. However, since the surface forms for adjectival usage and compounding forms in English being the same, one may have ambiguous expressions such as *South sea route.* But in Marathi which distinguishes between the adjectival and compounding usage, *dakṣiṇa sumudrī mārga* (<<south-sea>-route>) is different from *dakṣiṇī sumudrī mārga* (<south-<sea-route>>).

3.3 Type Identifier

The semantic classification of compounds given by Pāṇini is not only restricted to Sanskrit language per se, but is more general. For example, the Cambridge Grammar of the English language (Huddleston and Pullum, 2002) uses this classification to describe compounds in English. *Water pump* is an endocentric compound. An endo-centric compound typically shows a hyponymic relation with the head noun. An *egghead* is a bahuvrīhi compound meaning a person whose head resembles the shape of an egg, i.e. a high forehead, and hence intellectual. An example of a coordinative compound is *Hewlett-Packard* meaning a company whose owners are both Hewlett and Packard, or *secretary-treasurer* which stands for a person who is both a

[1] With a possible exceptions of coordinative and certain other rare compounds.
[2] for example, non-finite verbs ending in 'kta' suffix.
[3] for example sūtras corresponding to the 'avyayībhāva' type.

secretary as well as a treasurer. The type of a compound thus is useful in deciding the meaning of a compound. In order to decide the type of a compound, an access to the semantic content of its constituents, and possibly even to the wider context is needed. Now the immediate question is whether this classification of compounds into four classes is sufficient, or do we need further sub-classification. The grammatical texts sub-divide the Tatpuruṣa compounds further into sub-classes based on the case marker the first component takes when the compound is paraphrased. As an illustration, a compound *vidyānipuṇa* 'one who is sharp in the studies', when paraphrased, the first component takes a locative case marker, another compound *Daśarathaputraḥ* 'son of Dasharatha' takes a genitive marker, while the compound *vyāghrabhaya* 'fear of a tiger' takes an ablative case marker in Sanskrit. Based on the paraphrase, a set of 56 fine-grain tags was identified (Kumar et al., 2009) for Sanskrit.

It is important to note the level of semantics the compound tags deal with. Consider the compounds *rājapuruṣaḥ* 'King's servant', *Daśarathaputraḥ* 'son of Dasharatha', and *vṛkṣaśākhā* 'branch of a tree'. In the first case the relation between *rājan* 'king' and *puruṣa* 'man' is that of servant-master (*sevya-sevaka*), in the second the relation between *Daśaratha* and *putraḥ* 'son' is of father-son (*pitā-putra*) and in the third case the relation between *vṛkṣa* 'tree' and *śākhā* 'branch' is part-of (*avayava-avayavi*). However, in all the three cases instead of specifying these deeper relations, relation between the components is expressed through the genitive case suffix in the paraphrase of these compounds as *rājñaḥ puruṣaḥ* 'King's servant', *Daśarathasya putraḥ* 'son of Dasharatha', and *vṛkṣasya śākhā* 'branch of a tree', and thus these are classified as *Ṣaṣṭhī-Tatpuruṣa* 'genitive endocentric compounds with head to the right'. In other words, the classification is not guided by the deeper semantics, but by the paraphrase of a given compound, or by what the language expresses. Thus, on the one hand, to decide the meaning of a compound, we need a fine-grain tagset, at the same time, it should not be as fine-grained as to distinguish between the meaning of genitive cases in *rājñaḥ puruṣaḥ, Daṣarathasya putraḥ* and *vṛkṣasya śākhā*.

Assuming that we follow the fine-grained classification of compounds as dictated by the paraphrase, the question is, to what extent is it possible to decide the relation between the words only on the basis of components involved? For classification, a manually tagged corpus was used as a training data. A corpus of size 800K is tagged manually for the compounds in context by the Sanskrit Consortium. This had 92K instances of compound words. The distribution of frequent compounds is given in Table 1.

Type	Percentage
Endocentric	58.70
Karmadhāraya (IS_A)	18.11
Exocentric	11.04
copulative	5.67

Table 1: Distribution of Sanskrit Compounds

The endocentric compounds were further classified on the basis of missing case marker in the paraphrase. It was found that 55% of these compounds required genitive case marker. (Kumar et al., 2010) reported that the precision of a statistical classifier on this data considering only the major classification, is 72.7%. Allowing sub-divisions resulting into fine-grained tagset

lowers the performance to 63%. Statistical taggers perform well provided the training data is sufficient. So their performance goes down on compounds of rare type. Pāṇini has provided several sūtras in Aṣṭādhyāyī which deal with rare compound types. These sūtras provide various kind of semantic conditions under which a particular type of compound formation takes place. After going through the relevant sūtras, we observe that the conditions stated by Pāṇini fall under the following categories.

1. A restricted list of allowed components in certain type of compounds is provided.
2. A restriction in terms of special inflectional suffix / derivational suffix / category is mentioned.
3. A restriction is stated in terms of special technical terms, which are theory internal.
4. A restriction in terms of semantic relations between the components is mentioned.
5. Semantic property of the component is stated as a condition.

Out of these, the fourth and fifth category are important. The fourth category provides us clues for the important types of relations. Efforts such as Sanskrit WordNet (Kulkarni et al., 2010) or on marking semantic information in various kośas such as Amarakośa (Nair and Kulkarni, 2010) are concerned about lexical as well as semantic relations. In the sūtras related to compounds, we found the mention of following semantic relations.
i) *viśeṣaṇa-viśeṣya-bhāva* 'modifier-modified relation'
ii) *upamāna-upameya-bhāva* 'analogy or comparison'
iii) *avayava-avayavī-bhāva* 'part-whole relation'
iv) instrument-action relation.

The fifth category of conditions puts certain restrictions on the component in terms of semantic properties such as the component should be either a number or a direction or a color or a class indicating word or an adjective. This provides us a clue that the lexicon should have these semantic properties marked to enable automatic compound processing.

3.4 Paraphrase

The tagset for Sanskrit is dictated by the paraphrase. So except for some rare compound types with irregular paraphrases, typically each tag correspond to a well defined paraphrase (Kumar et al., 2009), distinct from the other, and one can then generate the paraphrase automatically. For example, the paraphrase rule for a *Ṣaṣṭhī Tatpuruśa* (T6) 'genitive compound' is given as below.

<x-y>T6 = x{6} y

where x{6} stands for the nominal form of x in genitive case. The paraphrase rules for the complete tagset along with the paraphrase generation is discussed in (Kumar et al., 2009). The problematic cases were those with an elision of certain terms called *madhyampadalopī*. For example, the paraphrase of *Śākapārthivaḥ* 'vegetable - human' is *Śākapriyaḥ pārthivaḥ* 'a human who likes vegetables'. In order to get this paraphrase, we need to insert appropriate content word. Such compounds are rare in Sanskrit, and are listed as exceptions.

3.5 Insights Gained

This detailed study of Sanskrit compounds has helped us in getting a good insight into the compound processing. To understand the meaning of a compound, or to translate a compound into another language, first one needs to understand the underlying constituency structure. For example,
South Indian Cricket Association = <<*South-Indian*>-<*Cricket-Association*>>,
South Indian Food Plaza = <<<*South-Indian*>-*Food*>-*Plaza*>, and
Colon Cancer Tumor Suppressor Protein = <<<<*Colon-Cancer*>-*Tumor*>-*Suppressor*>-*Protein*>.

The second important step in understanding of a compound is to identify the relation between the component pairs. These relations are classified broadly into four categories depending upon the position of the head in the compound. The four major types of compounds were further sub-divided taking into account the differences in their paraphrases. Manual tagging of the Sanskrit compounds revealed that only few of the compounds are very frequent. They include *Tatpuruṣa*, *Bahuvrīhi* and the *Karmadhāraya*. In case of *Tatpuruṣa* compounds, the paraphrase requires appropriate case marker which shows the relation between the components. Among the *Tatpuruṣa* compounds *Ṣaṣṭhī Tatpuruṣa* 'genitive' were most frequent. The *Karmadhāraya* marking the relation of co-referentiality was also frequent. The Sanskrit grammar also provided us certain clues for identifying a compound and its types based on its components.

Finally an important observation from Pāṇini's treatment of Sanskrit grammar was the following. Pāṇini has strived hard to make his grammar as exhaustive as possible by providing rules to handle very rare compounds. So we could take the advantage of both the statistical techniques which perform better with frequent cases and the rule based approach to cover the rare cases.

4 Nominal Compounds in Marathi and Hindi

We used these insights for processing nominal compounds in two major Indian Languages, namely, Hindi and Marathi. Both languages do not have any specific convention for writing the compounds. We find instances of compounds written with components joined together, with a hyphen in between and also with a space in between. Compounds when written as a single word need a segmenter to split it into valid components.

It was observed that a special type of compounds termed as 'Avyayībhāva' are always written as a single word in these languages. Examples of such compounds are *yathāśakti* 'as per the capability', *anurūpa* 'in accordance with'. These are statistically found to be rare in Sanskrit Corpus. We also observe that in Hindi as well as Marathi also such compounds are rare. There could be two ways of accounting for such rare compounds:
1. To have them stored in the lexicon.
2. To have rules with the help of which such compounds can be analysed.
Pāṇini has accounted for such rare compounds with rules which we may use for Hindi and Marathi as well.

In what follows we concentrated only on the compounds written with a space in between. The study was undertaken only to decide the tagset for marking the relation between the components. The tagset for Sanskrit is very exhaustive, covering even the rare compounds. However the purpose of this study was to identify only those tags which are frequent in Hindi and Marathi.

The researchers working on the Cross Lingual Information Retrieval systems among Indian

Languages at IIT Bombay have developed a tool for automatic extraction of Multi Word Expressions from a corpus that uses minimum linguistic tools such as morphological analysers, and POS taggers. The candidates were ranked using Point-wise Mutual Information (PMI) method. Marathi corpus from Tourism domain consisting of 15,925 sentences with 0.325M words was chosen for the experiment. The Multi Word Expression extraction tool gave an initial set of Multi Word Expressions. From these Multi Word Expressions for Marathi, noun compounds were extracted manually, and a study was undertaken to identify the relations between the components. Table 2 lists the identified relations with examples from Marathi.

Dependence relation	Example	Gloss	Meaning
Tādarthya (Purpose)	Praveśa dvāra	Entry door	Door for entry
Karaṇa (Instrument)	Hasta shilpa	Hand Architechture	Architecture made by Hand
adhikaraṇa (Location)	Bhitti chitra	Wall painting	Painting on the wall
samānādhikaraṇa (co-referentiality)	Bauddha dharma	Buddhist religion	Buddhist religion
	sāhasika paryaṭana	adventurous Tourism	adventurous Tourism
	dara varṣī	Every in year	Every year
śeṣa (genitive)	samudra taṭa	Sea bank/shore	Shore of sea
	pāka kalā	Cooking art	Art of cooking
	paryaṭana sthala	tourism place	place of tourism
	upāhāra gṛha	little food house	restaurant

Table 2: Relations in Marathi Noun compounds

As observed in Sanskrit, in Marathi as well we found compounds with genitive case marker, compounds with co-referential components and compounds involving various kinds of dependency relation amongst the components were dominant.

In order to make sure that these relations are sufficient across other ILs, we repeated this study with Hindi. However, this time the compounds were extracted from a Hindi-Urdu dependency treebank being developed at IIIT Hyderabad. Pāṇinian grammar formalism is being followed for the annotation. The treebank has 10,799 sentences consisting of approximately 0.25M words. The compounds in this treebank form a chunk and are annotated with a special label. This made it easy to extract the sentences with compounds. We have examined around 827 sentences and identified 895 noun compounds with two components. Number of unique compounds is 597. Among them 20 are dvandva (copulative) compounds and 15 are cases of reduplication. We observe that compounds can be analysed with genitive (*ṣaṣṭhī sambandha*) for around 45% of times even though we understand that paraphrasing with genitives does not necessarily capture deep semantic relations (see section 3.3 for examples). Nevertheless for the purpose of machine translation genitive paraphrasing may be sufficient because a genitive construct in one language can be mostly translated into a genitive construct in another language whereas a source language compound need not remain a compound in the target language. For example, both the compound *room temperature* and the corresponding genitive construction *temperature of room* can only be translated into *kamare kā tāpamāna* in Hindi. In case of

English-Hindi language pairs, it was observed that in 59% of cases an English Noun compound can be translated into genitive construction in Hindi (Paul et al., 2010). However, for other NLP tasks such as information extraction, question answering etc., genitive relation will not be sufficient and one needs to look for deeper semantic relation. In the present work, we have attempted annotation of deeper semantic relation only when the paraphrase with genitive is illegitimate. The paraphrases were of the following types.

a) with vibhakti (equivalent to post-positions)
b) verb + vibhakti (and not with vibhakti alone)
c) Subtype relation (hyponymy relation)
d) Other kinds of paraphrase

We will discuss each case with suitable examples:

(I) Paraphrasing with vibhakti alone
We come across six classes of vibhaktis[4] which are used for paraphrasing other than the genitive one. Table 3 provides the examples.

(II) Paraphrasing with verb + vibhakti
We find many cases where a meaningful paraphrase is not possible with post-position alone. We have used verbal form along with post-position for meaningful paraphrasing for such cases. For example:

- antarīkṣa yāna vs antarīkṣa meṁ jāne vālā (or ke lie) yāna
  gloss: space ship vs ship that goes into space
- rela saḍaka vs rela calane ke lie saḍaka
  gloss: railway track vs track for the running of train
- rājya sarakāra vs rājya ke lie cunī gayī sarakāra
  gloss: state government vs a government to run the state affairs
- nirvāṇa sthala vs nirvāṇa prāpti (or pāne) ke lie sthala (which according to some annotators can be tagged as nirvāṇa ke lie sthala)
  gloss: nirvāṇa place vs a place where nirvāṇa is attained
- garbha gṛha vs garbha meṁ sthita gṛha
  gloss: inside room vs a room situated inside

(III) Hyponymic Relation
This relation is quite common apart from vibhakti paraphrasing. Hyponymic relations can again be of different nature as exemplified below.

(i) A hyponymic relation which is similar to samānādhikarana (see section 3.3). For example,
a. kāngresa dala vs kāngresa IS_A (nāmaka) dala
gloss: Congress Party
b. taṭarakṣaka bala vs taṭarakṣaka IS_A (nāmka) bala
gloss: Post guard

[4] Vibhakti or post-position can be multi word in Hindi; for example: ke viṣaya meṁ, ke bAre meṁ etc. These multi word expressions are treated as one post-position.

Vibhakti	Meaning	Instances		
		Compound	Gloss	English Translation
se, ke dvārā	With	rimoṭ kanṭrola	Remote Control	Remote Control
		gaisa pīḍita	Gas Victim	Gas Victim
		phoṭo pahacāna	Photo Identity	Identity Card
		dhvani pradūṣaṇa	Noise Pollution	Noise Pollution
		dūrasaṁchāra sevā	Telecom Service	Telecom Service
se	From	sevā nivṛtta	Service Retired	Retired from Service
		karma nivṛtti	Work Retired	Retired from Work
ke lie	For	surakshā bala	Security Forces	Security Forces
		samanvaya samiti	Sensation Committee	Sensation Committee
		ṭurisṭa hāusa	Tourist House	Tourist House
		prajanana aṁga	Reproductive Organs	Reproductive Organs
		śoka saṁvedanā	Mourning Sensation	Condolence
meṁ, para	In, On, At	besa kāmpa	Base Camp	Base Camp
		bāla aparādha	Child Crime	Juvenile Delinquency
ke viṣaya meṁ	About	vidhi śikṣā	Legal Education	Legal Education
		śramika mudde	Labour Issues	Labour Issues
ke sambandha meṁ, sambandhita	About (with regard to)	videśa nīti	Foreign Policy	Foreign Policy
		anusandhāna vibhāga	Research Department	Research Department

Table 3: Paraphrasing Hindi compounds with vibhakti alone

(ii) A hyponymic relation which refers the whole Multi Word Expression as a type of the entity that the head denotes; for example:
a. moṭara boṭa vs moṭara boṭa IS_A (Type of) boṭa (unlike (i), moṭara is not a boṭa)

gloss: Motor Boat
b. prashna cinha vs prashna cinha IS_A (Type of) cinha
gloss:question mark

(IV) Other kinds of paraphrases
We come across some cases where the genitive paraphrase is possible if and only if the modifier can be pluralized. For example,

(a) film abhinetā vs filmoṁ kā abhinetā
gloss: film actor vs actor of film

(b) pujārī samudāya vs pujārīyoṁ kā samudāya
gloss: priest group vs group of priests

(c) cālaka dala vs cālakoṁ kā dala
gloss: driver group vs group of drivers

The heads of (b) and (c) are aggregate nouns and therefore the modifier acquires plural meaning. In case of (a), *film (sg.) kā abhinetā* would mean actor of a particular film; whereas *film abhinetā* as a compound conveys the meaning of 'profession' as in *amitābha eka film abhinetā haiṁ*. The other suitable paraphrase would be *film meṁ kāma karane vālā abhinetā*, where *film* remains singular.

There are institutionalized terms such as *pulisa āyukta*, *vikāsa saciva* and so on and also borrowed compounds such as *ḍāyala up* 'dial up', *kebal cār* 'cable car', which we have left out of the scope of paraphrasing. Table 4 presents number of occurrence of various paraphrases in the data that we have analysed.

Type	No of Instances	Percentage
Genitive	270	47.12
Paraphrasing with Vibhakti alone	80	13.96
Hyponymic Relation	68	11.86
Paraphrasing with verb + Vibhakti	40	6.98
Copulative	20	3.49
Reduplications	15	2.62
Other kinds of Paraphrases	14	2.44
Difficulty in annotation	66	11.51

Table 4: Analysis of Hindi data for various types of paraphrases

This table further supports our intuition from analysis of Marathi data that the tatpuruṣa 'endocentric with missing case markers' and the copulative compounds are more frequent, providing a very strong empirical support for the development of tagset for semantic annotation of noun compounds. In Hindi we also found a considerable number of compounds which require an additional verb and a post position marker for paraphrasing. It is necessary to study the last

category of compounds further in order to enable machines to carry out the analysis and 'guess' the missing verb automatically.

5 Conclusion

It is clear from the data analysed for Sanskrit and Hindi that the most dominating type of compounds is the Ṣaṣṭhī Tatpuruṣa (genitive), in both languages. There are cases of Tatpuruṣa which cannot be analysed with genitive and they are paraphrased with various vibhaktis (case markers). We come across quite a number of cases of Karmadhāraya 'hyponymic' compounds. It is interesting to note that compounding in Hindi deviates from that in Sanskrit in two aspects. Data analysed for Hindi does not contain any instance of Bahuvrīhi (exo-centric) compound. Second, Hindi data presents many cases where quite a lot of compounds requires a verb as well as vibhakti for its paraphrasing. Such compounds are termed as madhyama-pada-lopī in Sanskrit, and they are found to be rare in Sanskrit. The compounds which were found to be difficult for the annotators should form the part of a lexicon.

References

Butnariu, C. and Veale, T. (2008). A concept-centered approach to noun compound interpretation. In *Proceedings of the 22nd International Conference on Computational Linguistics (COLING-08)*, Manchester, UK.

Finin, T. W. (1980). The semantic interpretation of nominal compounds. In *In the Proceedings of the 1st Conference on Artificial Intelligence (AAAI-80)*.

Girju, R., Badulescu, A., and Moldovan, D. (2003). Learning semantic constraints for the automatic discovery of part-whole relations. In *In the proceedings of the Human Language Technology Conference (HLT)*.

Girju, R., Nakov, P., Nastase, V., Szpakowicz, S., Turney, P., and Yuret, D. (2007). Classification of semantic relations between nominals. In *Proceedings of The Semantic Evaluation Workshop (SemEval) in Conjunction with ACL*, Prague.

Huddleston, R. and Pullum, G. K. (2002). *The Cambridge Grammar of the English Language*. Cambridge University Press.

Huet, G. (2009). Formal structure of Sanskrit text: Requirements analysis for a mechanical Sanskrit processor. In Huet, G., Kulkarni, A., and Scharf, P., editors, *Sanskrit Computational Linguistics 1 & 2*. Springer-Verlag LNAI 5402.

Kim, S. N. and Baldwin, T. (2006). Interpreting semantic relation in noun compound via verb semantics. In *Proceedings of ACL/COLING-2006*.

Kulkarni, A. and Kumar, A. (2011). Statistical constituency parser for Sanskrit compounds. In *Proceedings of ICON 2011*. Macmillan Advanced Research Series, Macmillan Publishers India Ltd.

Kulkarni, M., Dangarikar, C., Kulkarni, I., Nanda, A., and Bhattacharyya, P. (2010). Introducing sanskrit wordnet. In Pushpak Bhattacharyya, C. F. and Vossen, P., editors, *Principles, Construction and Application of Multilingual Wordnets, Proceedings of the Global Wordnet Conference, 2010*. Narosa Publishing House, New Delhi.

Kumar, A. (2012). *An automatic Sanskrit Compound Processing*. PhD thesis, University of Hyderabad, Hyderabad.

Kumar, A., Mittal, V., and Kulkarni, A. (2010). Sanskrit compound processor. In Jha, G. N., editor, *Proceedings of the International Sanskrit Computational Linguistics Symposium*. Springer-Verlag LNAI 6465.

Kumar, A., SheebaSudheer, V., and Kulkarni, A. (2009). Sanskrit compound paraphrase generator. In *Proceedings of ICON 2009*.

Lauer, M. (1995). *Designing Statistical Language Learners: Experiments on Noun compounds*. PhD thesis, Macquarie University, Australia.

Mittal, V. (2010). Automatic sanskrit segmentizer using finite state transducers. In *Proceedings of the ACL 2010 Student Research Workshop*, pages 85–90, Uppsala, Sweden. Association for Computational Linguistics.

Nair, S. and Kulkarni, A. (2010). The knowledge structure in amarakośa. In Jha, G. N., editor, *Proceedings of the International Sanskrit Computational Linguistics Symposium*. Springer-Verlag LNAI 6465.

Nakov, P. (2008). Noun compound interpretation using paraphrasing verbs: Feasibility study. In *Proceeding of 13th International Conference on Artificial Intelligence: Methodology, Systems and Applications (AIMSA-08)*, Varna, Bulgaria.

Nastase, V. and Szpakowicz, S. (2009). The same semantic relations link structurally different realizations of concept. In *Linguistic Issues in Language*.

Paul, S., Mathur, P., and Kishore, S. (2010). Syntactic construct: An aid for translating english nominal compound into hindi. In *Proceedings of the NAACL HLT Workshop on Extracting and Using Constructions in Computational Linguistics*, Los Angeles, California.

Prince, A. and Smolensky, P. (1993). Optimality theory: Constraint interaction in generative grammar. Technical report, Rutgers University, Piscataway.

Séaghdha, D. O. and Copestake, A. (2007). Co-occurrence contexts for noun-compound interpretation. In *Proceedings of the ACL-07 Workshop on a Broader Perspective on Multiword Expression (MWE-07)*, Prague, Czech Republic.

Shastri, G. (2006). *Patañjali's Vyākaraṇa Mahābhāṣya with Kaiyaṭa's Pradīpa and Nāgojibhaṭṭa's Uddyota with the Notes by Guruprasad Shastri (Adhyāya 2)*. Rashtriya Sanskrit Sansthan, New Delhi (reprint of 1938 edition).

Vanderwende, L. (1995). *The Analysis of Noun Sequences Using semantic Information Extracted from on-line Dictionaries*. PhD thesis, Georgetown University.

Unsupervised Japanese-Chinese Opinion Word Translation Using Dependency Distance and Feature-Opinion Association Weight

Guo-Hau Lai, *Ying-Mei Guo* and *Richard Tzong-Han Tsai*[*]
Department of Computer Science and Engineering, Yuan Ze University
135 Yuan-Tung Road, Chungli, Taoyuan, Taiwan, R.O.C.
[*]Corresponding Author
{s986005, s996148}@mail.yzu.edu.tw, thtsai@saturn.yzu.edu.tw

ABSTRACT

Online shoppers depend on customer reviews when evaluating products or services. However, in the international online marketplace, reviews in a user's language may not be available. Translation of online customer reviews is therefore an important service. A crucial aspect of this task is translating opinion words, key words that capture the reviewers' sentiments. This is challenging because opinion words often have multiple translations. We propose an unsupervised opinion word translation disambiguation scoring method using dependency distance and feature-opinion association as weighting factors. The scores of an opinion word's translation and its surrounding words' translations are estimated using Google snippets. We focus on Japanese-Chinese translation of hotel reviews from Rakutan Travel, using the 10 most common polysemous Japanese opinion words to evaluate system performance. Results show our weighting factors significantly improve translation accuracy compared to Google and Excite.

KEYWORDS : Opinion Word Translation Disambiguation, Dependency Distance, Feature-Opinion Association

Proceedings of COLING 2012: Technical Papers, pages 1503–1518,
COLING 2012, Mumbai, December 2012.

1 Introduction

The development of Web 2.0 has made it easier for internet users to post their reviews or comments about products or services on structured websites. Online shoppers are increasingly likely to look at these reviews before deciding on a purchase. In recent years, the research field of sentiment analysis has focused on analyzing this form of textual-information, particularly opinions or sentiments expressed by internet users.

However, given the international nature of the web and online shopping, opinions in a user's mother language may not be available. Translation of online customer reviews is therefore an important service sought after in many markets. A crucial aspect of customer review translation is translating opinion words, key words that capture the sentiments. At present, machine translation (MT) systems can translate whole sentences or even complete paragraphs. This not a trivial task, however, because opinion words usually have multiple possible translations and the MT systems have low accuracy on polysemous words (Carpuat & Wu, 2005).

This paper proposes an unsupervised method of selecting the most appropriate Chinese translation for an opinion word in a given Japanese sentence. Candidate translations are retrieved from a bilingual dictionary. Consider the following Japanese sentence:

綺麗な夜景とともに食事を楽しむことができました。

(I was able to enjoy a nice meal with a beautiful night view.)

The target opinion word 綺麗 has three candidate translations: 漂亮 (beautiful), 乾淨 (clean), and 清楚 (clear) in Chinese. In this example, the most appropriate translation is 漂亮 (beautiful). This disambiguation problem is known as Word Translation Disambiguation (WTD).

One way to solve the WTD problem is to calculate the sum of association scores of pairs among translation of the target word and all its surrounding words' translations, and then select the one with the highest score. However, since the different surrounding words have different amounts of influence on the target word, it is necessary to add some weighting factors (e.g., word distance). Since our goal is disambiguating opinion words in opinionated sentences, the product features (or aspect expressions) should have direct influence on translation selection. In the above example, 夜景 (night view) and 食事 (meal) are product features, the former having the greatest influence on the opinion word 綺麗 (beautiful).

Our proposed unsupervised opinion word translation disambiguation scoring method uses the dependency distance and feature-opinion association as weighting factors. The scores of an opinion word's translation and its surrounding words' translations are estimated using Google search snippets. In our experiments, we focused on opinion word translation of hotel reviews from Japanese to traditional Chinese. From a dataset of hotel reviews compiled from Rakuten Travel, we selected the top-10 most common polysemous Japanese opinion words to evaluate the performance of our system. The results show that our weighting factors have significantly improved translation accuracy. Compared to Google Translate and Excite translation system, our system can translate opinions more accurately, which could be a boon for Chinese online shoppers seeking accommodations in Japan.

The remainder of this paper is organized as follows: Section 2 introduces some related work while Section 3 describes our proposed method in detail. The experimental results are given in Section 4. The error analysis discussed in Section 5. Finally, conclusion gives in the last section.

2 Related Work

In this section, we introduce some previous works related to our method.

Word translation disambiguation (WTD) (Marsi, Lynum, Bungum, & Gambäck, 2011), also called cross-lingual word sense disambiguation (CL-WSD), which is the task of selecting the most appropriate translation of a polysemous word in a given context. This task can be seen as a special variant of WSD.

The best-known open task in this specialty is the Multilingual Lexical Sample/CL-WSD task, held by the Senseval/SemEval workshop (Chklovski, Mihalcea, Pedersen, & Purandare, 2004; Jin, Wu, & Yu, 2007; Lefever & Hoste, 2010). This task provides a framework for the evaluation of systems that perform machine translation, with a focus on the translation of ambiguous words. Unlike in other lexical sample tasks, the sense inventory for CL-WSD is the set of translations from a bilingual dictionary or a parallel corpus instead of human-defined sense labels.

There have been several studies that use cross-lingual evidence to deal with the WSD problem: (Chan & Ng, 2005; Chklovski et al., 2004; Ng, Wang, & Chan, 2003). These approaches rely on large parallel corpora to train a WSD classifier. However, for some language pairs (e.g., Japanese-Chinese), such corpora are not available. To overcome this problem, Dagan and Itai (1994) used a bilingual lexicon and statistical data from a monolingual corpus of the target language for disambiguation. Tsunakawa and Kaji (2010) proposed a method for using a bilingual dictionary with a correlation matrix to select an appropriate translation word. An item in the matrix is the correlation score between associated words and candidate translations.

The Web is increasingly being used as a data source in a wide range of natural language processing tasks including WSD. Liu and Zhao (2009) presented a fully unsupervised WTD method which selects the maximum sum of Web Bilingual Relatedness (WBR) between a translation and all context words. The WBR is calculated by four association measures based on mixed-language webpage counts from the Baidu search engine. Their WBR model outperformed the best unsupervised SemEval-2007 participant system in the Multilingual Chinese-English Lexical Sample Task. Another work using the Web as a knowledge resource is (Liu, Xue, Li, & Liu, 2010), which is based on minimum Normalized Google Distance and also outperformed the best unsupervised participant system in SemEval-2007.

Most of these studies use the corpus statistics to measure the association between a candidate translation and its context words to disambiguate polysemous words. They tend to consider that all context words have equal influence on a target word. However, since our work is a special case of WTD that focuses on translating polysemous opinion words for a given opinionated sentence, we need give more weight to words that are closely related to opinion words, such as the product features.

3 Opinion Word Translation Disambiguation

In this section, we introduce our method for selecting the translation of a given opinion word in a sentence. The procedure consists of six parts: (1) related word extraction, (2) related word translation, (3) translation corpus, (4) Japanese dependency analysis, (5) related product feature identification, and (6) word translation disambiguation scoring method. The procedures are described in detail in the following sections.

For related word extraction, we apply Japanese compound combination rules to extract nearby words related to the target word. Then, the extracted related words are translated from Japanese to Chinese by our dictionary-based system, which uses an online bilingual dictionary. The translation corpus is compiled from snippets returned by Google Search. To obtain dependency distances among the target word and related words, we feed the given sentence into a Japanese dependency analyzer. We also identify all product features appearing in the given sentence and estimate the association between the target word and each feature. Finally, the disambiguation scoring method calculates scores for each candidate translation to determine the appropriate one.

3.1 Related Word Extraction

We use the following procedure to extract nearby words related to the target word: First, the test sentences are analyzed by a Japanese part-of-speech (POS) and morphological analyzer, MeCab (Kudo, 2005). The MeCab output contains not only the segmented words with POS tags but also detailed information on the *katsuyou* form, root form, and pronunciation of each word. The *katsuyou* is the inflection of the *yougen* (a verb, an adjective, or an auxiliary verb). According to its tense and voice, the *yougen* may have different inflections. For example, 美味しい (delicious) should inflect to 美味しかった in the past tense and 美味しくない (not delicious) in the negative. However, dictionaries usually do not include all these inflections. So we use the root forms instead of the original segmented words in subsequent processing steps.

In addition, we found one difficulty using MeCab: due to the annotation standards of its training corpus, MeCab sometimes treats one compound or loanword as many morphemes. For example, 従業員 (staff) is separated into 従業 (work) and 員 (member), and the loanword エントランス (entrance) is separated into エン (dollar) and トランス (transformer). In both cases, the original meaning is lost. To solve this problem, we apply the Japanese compound combination rules shown in TABLE 1. Most of these morphological rules are based on POS tags. For example, a compound adjective (*comadj*) is composed of a verb (main) followed by an adjective (sub).

noun → noun-verbal | noun-common | noun-proper-misc
prefix → prefix-nominal
suffix → noun-suffix-misc
noun' → *noun'* + *noun*
| *noun* + *noun*
comnoun → *prefix* + *noun* + *suffix*
| *prefix* + *noun'*
| *noun* + *suffix*
| *noun'* + *suffix*
| *noun'*
comadj → verb-main + adjective-sub
location → noun-proper-place-misc + noun-suffix-place
comword → *comnoun* | *comadj* | *location*

comadj: compound adjective
comnoun: compound noun
comword: compound word

TABLE 1 – Japanese Compound Combination Rules

Finally, we filter out irrelevant words using a list of stop words. From the remaining words, we retain only adjectives, verbs and nouns with the following POS tags, as shown in TABLE 2.

Type	POS tags
adjective	adjective-main, adjective-suffix, adjective-sub
verb	verb-main
noun	noun-verbal, noun-common, noun-adjective-base, noun-proper-misc, noun-proper-organization

TABLE 2 – The list of POS tags for retained words

3.2 Related Word Translation

The extracted related words are translated by our dictionary-based system, which uses the Sanseido Japanese-Chinese dictionary[1]. Japanese words are input and Chinese translations are output. Given a Japanese word w_j, all translations of w_j in the dictionary are regarded as potential translations. The Sanseido dictionary includes a total of 28,000 entries and provides the majority of Chinese translations for this study. For few related words that do not appear in this dictionary, we use results from Google Translate.

Since the Sanseido dictionary's Chinese translations are in simplified Chinese, our system must convert the output to traditional Chinese characters. Direct conversion by table lookup may result in mistranslations since the usage of some terms is quite different in mainland China and Taiwan. We use a mapping table of common synonymous words provided by China Biz[2] to improve conversion accuracy.

One translation difficulty that is often encountered in informal-style online user reviews is varying use of common words or expressions. For example, the word すごい (very) can be expressed by the hiragana terms すごーい (terrible) or すっごい (terrific) and the katakana term スゴイ (amazing). Ikeda, Yanagihara, Matsumoto, and Takishima (2009) proposed a normalization algorithm to reduce the number of variant expressions. We apply some of their conversion rules to improve the recall of our translation system. For example, if words are written in all katakana (e.g., スゴイ, the above example), it may imply emphatic use. In this case, we convert the words to hiragana (e.g., スゴイ → すごい). In polite usage, the honorific prefixes (e.g., お, ご, and 御) are used in words such as お手洗い (toilet). In these cases, we remove the prefixes (e.g., お手洗い → 手洗い).

3.3 Translation Corpus

We compiled our translation corpus from snippets returned by Google Search for conjunctive queries of word pairs. These snippets provide useful clues related to the semantic relations that exist between two words. First, we take Chinese word pairs in which one word is a candidate translation of the target word and the other is a translation of a related word. Then, we submit each word pair joined by the Boolean operator "AND" to Google Search and collect the first 500 snippets for our text corpus.

[1] http://www.excite.co.jp/dictionary/japanese_chinese/
[2] http://www.chinabiz.org.tw/

3.4 Japanese Dependency Analysis

In order to obtain the dependency distance between the target word and a related word, we use CaboCha, a Japanese dependency structure analyzer based on Support Vector Machines (SVMs) and the most accurate publicly available system to date, with a reported accuracy of 89.29% (Kudo & Matsumoto, 2002). CaboCha uses a parsing algorithm based on the Cascaded Chunking Model.

FIGURE 1 – An example of Japanese dependency parsing

The basic syntactic unit used in Japanese parsing is the *bunsetsu*, which consists of one or more words followed by either nothing or function words such as particles and auxiliary verbs. FIGURE 1 shows an example of Japanese dependency parsing for a sentence お部屋の照明の照度は大変明るいです！ (The illumination of the lighting of the room is very bright!). In this example sentence, we first find the last node which contains the target word 明るい (bright). Then we calculate all dependency distances from other nodes which contain a related word. For example, the distance from the target word's node to the third node containing the related word 照度 (illumination) is one and the distance to the second node containing 照明 (lighting) is two.

3.5 Related Product Feature Identification

In opinionated sentences, opinion words often describe product attributes or features, such as cleanliness, staff attitude, food quality, etc. in the hotel domain. We believe that considering the product feature(s) related to the target opinion word is helpful for disambiguation of the opinion word. To implement this feature, we enumerated product features[3] for the hotel domain.

3.6 Word Translation Disambiguation Scoring Method

This section describes the word translation disambiguation scoring method. Assume the target opinion word is *o*. One way to select the appropriate translation of *o* is to first calculate the association scores for pairs of each candidate translation and each of its related words' translations, and then select the translation with highest sum of these scores. Consider the following formula (Assume *t* is a candidate translation for *o*):

$$Translation(t \mid o) = \sum_{s_i \in S} \sum_{t' \in translations(s_i)} \frac{Association_T(t, t') \cdot Association_S(o, s_i)}{|translations(s_i)|} \quad (1)$$

where $Translation(t \mid o)$ is the score of the candidate translation *t*, *S* is the set of all related words, and $translations(s_i)$ is the set of s_i's translations (for convenience, hereafter referred to as related translations). For instance, *o* has two candidate translations (t_1 and t_2) and two related words (s_1 and s_2). The translations of s_1 are $t'_{1,1}$, $t'_{1,2}$ and $t'_{1,3}$, and s_2 can be translated to $t'_{2,1}$. Using the above notation, $translations(s_1) = \{t'_{1,1}, t'_{1,2}, t'_{1,3}\}$ and $translations(s_2) = \{t'_{2,1}\}$. $Association_T(t, t')$

[3] The product feature list is available at http://iisr.cse.yzu.edu.tw/~guohau/coling/feature.list

is the association score between a candidate translation t and a related translation t'. $Association_S(o, s_i)$ is the association score between the target opinion word o and a related word s_i—this weighting factor is primarily designed to model their association in the Japanese context. Notice that the final score is divided by the size of $translations(s_i)$. The purpose of adding this term is to balance out the extra influence of s_i's multiple possible translations. In the next two parts, we will introduce the terms $Association_T(t, t')$ and $Association_S(o, s_i)$ in detail.

3.6.1 Association Score between Translations in the Target Language

In this section, we describe how we estimate $Association_T(t, t')$. There are many ways to measure the correlation between two words: one simple way is to calculate the mutual information score in the corpus. First, the snippets described in Section 3.3 are split into segments using punctuation. Their similarity is estimated by Pointwise Mutual Information (PMI) (Church & Hanks, 1990), which is defined as:

$$Association_T(t, t') = PMI(t, t') = \log_2 \frac{p(t, t')}{p(t) \cdot p(t')} \tag{2}$$

where $p(t)$ and $p(t')$ are the probability of word t and t' appearing separately in the corpus, and $p(t, t')$ is the probability of the co-occurrence of word t and t' in the corpus, which is estimated by the number of co-occurring segments for t and t' divided by the total number of segments. However, using PMI does not yield the expected $Association_T(t, t')$ measurements, because the corpus is compiled from the search snippet results of sending all candidate-/related-translation pairs to Google Search. The very low or zero co-occurrence frequencies of some pairs cause difficulty in calculating their values. For instance:

if $\tilde{p}(t, t') \to 0$, then $PMI(t, t') \to -\infty$.

In addition, according to Formula 1, each term in the summation is the product of two scores: $Association_T$ and $Association_S$. For two related words s_1 and s_2 and any of their translations $t'_{1,j}$ and $t'_{2,k}$, if $Association_T(t, t'_{1,j})$ is greater than $Association_T(t, t'_{2,k})$ and $Association_S(o, s_1)$ is greater than $Association_S(o, s_2)$, but $Association_T(t, t'_{1,j})$ and $Association_T(t, t'_{2,k})$ are both negative (PMI score), $Association_T(t, t'_{1,j}) * Association_S(o, s_1)$ is not guaranteed to be larger than $Association_T(t, t'_{2,k}) * Association_S(o, s_2)$. Such a result is not appropriate for our application here.

To solve this problem, PMI is mapped to an exponential function where the value is always positive, which is defined as:

$$PMI_{Exp}(t, t') = e^{PMI(t, t')} \tag{3}$$

3.6.2 Association Score between the Opinion Word and Related Words in the Source Language

We aim to determine the translation of an opinion word by scoring its association to its related words in the source language. Different related words have different influence on the target word, so added weighting factors are necessary. There are two factors that we consider: dependency distance and feature-opinion association.

Distance weighting has been used in several studies (Beeferman, Berger, & Lafferty, 1997; Brosseau-Villeneuve, Nie, & Kando, 2010; Gao, Zhou, Nie, He, & Chen, 2002) as a means of estimating the association between two words. The exponential model, in which association between two words decreases exponentially when the distance between them increases, is a commonly used approach. We employ Beeferman et al. (1997)'s distance weighting approach. Therefore, the association score of o and s_i is defined as:

$$Association_S(o, s_i) = \mu \cdot e^{-\mu \cdot distance(o, s_i)} \tag{4}$$

where $distance(o, s_i)$ is the dependency distance (see Section 3.4 for the details) between o and s_i; and μ is the parameter for decay rate determined by maximum likelihood estimate:

$$\mu = \log_2 \left(1 - \frac{1}{E_{\tilde{p}}[k]} \right) \approx E_{\tilde{p}}[k] - \sum_{k \geq 0} k \tilde{p}(k) \tag{5}$$

where $\tilde{p}(k)$ is the probability of the distance between o and s_i being k.

Since our goal is disambiguating opinion words in opinionated sentences, we should give product-feature words more influence on the translation selection than normal words. To do this automatically, we modify Formula 4 by introducing the feature-opinion association (FOA) score, which is defined as:

$$FOA(o, s_i) = \begin{cases} \min\left[J(o, s_i), \frac{1}{\lambda} \right] & \text{, if } s_i \text{ is a predefined product feature} \\ \frac{1}{\lambda} & \text{, if } s_i \text{ is not a predefined product feature} \end{cases} \tag{6}$$

where the constant value λ is determined empirically to be 1500. The purpose of introducing the minimal function is to set the lower bound of FOA to be $1 / \lambda$, which is a reasonably small value. $J(o, s_i)$ is the Jaccard coefficient, which is defined as:

$$J(o, s_i) = \frac{|o \cap s_i|}{|o \cup s_i|} = \frac{freq(o, s_i)}{freq(o) + freq(s_i) - freq(o, s_i)} \tag{7}$$

Then, we can modify $Association_S(o, s_i)$ as follows:

$$\begin{aligned} &Association_S(o, s_i) = \mu \cdot e^{-\mu \cdot IFOA(o, s_i) \cdot distance(o, s_i)}, \\ &IFOA(o, s_i) = \frac{1}{\lambda \cdot FOA(o, s_i)} \end{aligned} \tag{8}$$

where $IFOA$ is the abbreviation of the inversed FOA score.

In Formula 6, we mentioned that when s_i is not a predefined product feature, $FOA(o, s_i)$ is $1 / \lambda$, making the $IFOA(o, s_i)$ 1, which implies that $IFOA(o, s_i)$ does not have any effect. This increases the influence of related product-feature words while having no influence on regular related words.

3.7 Application of Our WTD Formulae

Let us consider the following example to demonstrate our proposed WTD scoring method:

お部屋もお風呂も綺麗に掃除がされている。

(The room and bathroom have been swept clean.)

The target opinion word **綺麗** has three candidate translations: 漂亮 (beautiful), 乾淨 (clean), and 清楚 (clear). For convenience, we consider only the first two: 漂亮 (beautiful) and 乾淨 (clean). The steps of our WTD method are as follows:

1. Related Word Extraction
   Three related words are extracted from the input sentence: **部屋** (room), **風呂** (bathroom), and **掃除** (sweep).

2. Related Word Translation
   The related words are translated into Chinese (called related translations): 房間 (room) and 屋子 (house) for **部屋**, 浴室 (bathroom) for **風呂**, 打掃 (sweep) for **掃除**. TABLE 3 lists Japanese words and their Chinese translations used in this application.

Japanese word	Chinese translations
綺麗	漂亮 (beautiful), 乾淨 (clean), 清楚 (clear)
部屋	房間 (room), 屋子 (house)
風呂	浴室 (bathroom)
掃除	打掃 (sweep)

TABLE 3 – Japanese word and its Chinese translations

3. Translation Corpus
   We look up the Chinese word pairs (e.g., "漂亮" AND "房間", "乾淨" AND "房間") in our Google Search translation corpus (Section 3.3) and collect the snippets.

4. Japanese Dependency Analysis
   After dependency analysis, the dependency distances between each target opinion word and the related word are acquired:
   {**綺麗-部屋**: 3, **綺麗-風呂**: 2, **綺麗-掃除**: 2}

5. Related Product Features Identification
   The product features are **部屋** (room) and **風呂** (bathroom).

6. Word Translation Disambiguation Scoring Method
   Consider Formula 1:

$$Translation(t \mid o) = \sum_{s_i \in S} \sum_{t' \in translations(s_i)} \frac{Association_T(t, t') \cdot Association_S(o, s_i)}{\left| translations(s_i) \right|}$$

   o: the target opinion word **綺麗**
   s_i: any related word in {**部屋**, **風呂**, **掃除**}
   t: any candidate translation in {漂亮, 乾淨}
   t': any related translation in {房間, 屋子, 浴室, 打掃}

   Consider Formula 2:
   Assume the $Association_T(t, t')$ for each candidate translation-related translation pair is:
   {漂亮-房間: 0.5, 漂亮-屋子: 0.55, 漂亮-浴室: 0.4, 漂亮-打掃: 0.35,
   乾淨-房間: 0.6, 乾淨-屋子: 0.45, 乾淨-浴室: 0.7, 乾淨-打掃: 0.75}

Consider Formula 5:

$$\mu = \log_2\left(1+\frac{3}{3+2+2}\right) = 0.51$$

Consider Formulae 6 to 8:
We assume the feature-opinion association scores [$FOA(o, s_i)$] are:
FOA(綺麗, 部屋) = 0.0025, $IFOA$(綺麗, 部屋) = 0.27
FOA(綺麗, 風呂) = 0.0032, $IFOA$(綺麗, 風呂) = 0.21

Then, the $Association_S(o, s_i)$ between the target opinion word and each related word is:
$Association_S$(綺麗, 部屋) = $0.51 * e^{-0.51 * 0.27 * 3} = 0.34$
$Association_S$(綺麗, 風呂) = $0.51 * e^{-0.51 * 0.21 * 2} = 0.41$
$Association_S$(綺麗, 掃除) = $0.51 * e^{-0.51 * 1 * 2} = 0.18$

Now, we can calculate the weighted score for each candidate translation:
Translation(漂亮 | 綺麗)
= $Association_S$(綺麗, 部屋) * $Association_T$(漂亮, 房間) / |*translations*(部屋)| +
$Association_S$(綺麗, 部屋) * $Association_T$(漂亮, 屋子) / |*translations*(部屋)| +
$Association_S$(綺麗, 風呂) * $Association_T$(漂亮, 浴室) / |*translations*(風呂)| +
$Association_S$(綺麗, 掃除) * $Association_T$(漂亮, 打掃) / |*translations*(掃除)|
= 0.34 * 0.5 / 2 + 0.34 * 0.55 / 2 + 0.41 * 0.4 / 1 + 0.18 * 0.35 / 1
= 0.41

Translation(乾淨 | 綺麗)
= $Association_S$(綺麗, 部屋) * $Association_T$(乾淨, 房間) / |*translations*(部屋)| +
$Association_S$(綺麗, 部屋) * $Association_T$(乾淨, 屋子) / |*translations*(部屋)| +
$Association_S$(綺麗, 風呂) * $Association_T$(乾淨, 浴室) / |*translations*(風呂)| +
$Association_S$(綺麗, 掃除) * $Association_T$(乾淨, 打掃) / |*translations*(掃除)|
= 0.34 * 0.6 / 0.5 + 0.34 * 0.45 / 0.5 + 0.41 * 0.7 / 1 + 0.18 * 0.75 / 1
= **0.6**

So, the chosen translation is 乾淨 (clean).

4 Experiments

We conducted experiments with our Japanese hotel review corpus to empirically evaluate the translation accuracy of our WTD scoring method using different sets of weighting factors and the modified PMI formula. We also compared our system's performance to that of Google Translate and the Excite translation system.

4.1 Dataset

Our dataset consists of 956,892 reviews of 15,291 hotels from the Rakutan Travel website[4], the largest hotel-booking/review website in Japan. The sentences are segmented and duplicate content is removed. After processing, the dataset contains 4,341,266 sentences. We also use this data to create the product feature list for Section 3.5.

[4] http://travel.rakuten.co.jp/

4.2 Experiment Design

We selected the top-10 most common polysemous opinion words and annotated each of their occurrences in the dataset with their translations. For each of the ten opinion words, we constructed test examples by randomly selecting 1,200 sentences from the dataset. Each test example contains only one opinion word. The ground truth of each translation was assigned by two human annotators. Statistics for the gold standard dataset are presented in TABLE 4.

Word	#Sense	#Instance	Avg length	Min length	Max length
明るい (bright)	2	992	36.3	7	135
甘い (sweet)	2	808	40.3	7	145
暖かい (warm)	2	979	41.6	6	147
丁寧 (polite)	2	1,057	37.9	11	125
冷たい (cool)	2	957	44.0	6	174
薄い (thin)	2	1,041	38.3	6	141
綺麗 (beautiful)	3	736	35.2	9	113
きつい (tiring)	3	755	41.9	10	136
寂しい (lonely)	3	794	38.7	8	120
厳しい (strict)	4	506	45.0	7	141
Avg.	**2.5**	**862.5**	**39.9**	**7.7**	**137.7**

TABLE 4 – Statistics for the gold standard dataset

As shown in TABLE 4, our gold standard dataset contains a total of 8,625 test examples with an average of 2.5 senses per word. The average length of the test examples is 39.9 Japanese characters.

In order to measure the impact of different sets of weighting factors and modified PMI formula on translation accuracy, we ran a set of 30 experiments for each configuration. For each experiment, we randomly chose 85% of the test examples for each opinion word. After running all the experiments, we performed a two-tailed paired T-test on the average accuracies of different sets of weighting factors to prove our weighting approach significantly improved translation accuracy. We also compared our system's performance with that of Google Translate and the Excite translation system using the same test method. The online translation systems' performance was checked by two annotators (only for the correctness of the opinion word translation, but ignoring translation of surrounding words).

4.3 Evaluation Metrics

For each opinion word, the results are given in terms of translation accuracy, which is defined as:

$$accuracy = \frac{\#\ of\ correct\ translations}{\#\ of\ test\ sentences} \tag{9}$$

We also calculated macro and micro averages to measure the overall performance across all opinion words. The macro average is simply the average of the accuracies of all ten opinion words. In contrast, micro average is the sum of correct occurrences divided by the sum of all occurrences.

4.4 Results

TABLE 5 shows the experimental results for the different configurations of our WTD scoring method. The value in each cell indicates the average accuracy. For our baseline system, we used the most frequent sense (MFS) method:

$$MFS = \frac{\#most\ \ frequent\ sense}{\#test\ sentence} \tag{10}$$

In TABLE 5, A_T(PMI) stands for the configuration in which $Association_T$ (t, t') is estimated by the original PMI (Formula 2) and $Association_S$ (o, s_i) is set to 1. $A_T(PMI_{Exp})$ means that the modified PMI formula (Formula 3) is used to replace Formula 2. $A_T(PMI_{Exp})+A_S(D)$ means that $Association_S$ (o, s_i) is estimated by Formula 4, which considers the dependency distance. $A_T(PMI_{Exp})+A_S(D+F)$ means that Formula 8, which considers both dependency distance and feature-opinion association, is used to replace Formula 4. TABLE 5 shows that $A_T(PMI_{Exp})$ significantly improves overall accuracy by about 7.6% over A_T(PMI). $A_T(PMI_{Exp})+A_S(D)$ improves overall performance by about 9.6% from A_T(PMI), and $A_T(PMI_{Exp})+A_S(D+F)$ achieved the best result, improving overall accuracy by about 11% over A_T(PMI). It should be noted that $A_T(PMI_{Exp})+A_S(D+F)$ also has a positive impact on performance of every opinion word individually.

Compared with the online translation systems, TABLE 5 shows our system outperforms Excite and Google for opinion word translation. The two online translation systems perform even worse than MFS on average.

Word	MFS	Excite	Google	A_T(PMI)	$A_T(PMI_{Exp})$	$A_T(PMI_{Exp})$ $+A_S(D)$	$A_T(PMI_{Exp})$ $+A_S(D+F)$
明るい	0.7323	0.7311	0.6460	0.8656	0.9067	0.9198	0.9212
甘い	0.7679	0.7679	0.7291	0.8062	0.8431	0.8608	0.8876
暖かい	0.5234	0.4766	0.7103	0.6685	0.7988	0.8183	0.8600
丁寧	0.8284	0.0439	0.7854	0.5045	0.8287	0.8679	0.8711
冷たい	0.9098	0.9119	0.6790	0.9382	0.9119	0.9210	0.9386
薄い	0.9033	0.9033	0.8198	0.8734	0.9088	0.9293	0.9549
綺麗	0.4845	0.5313	0.5408	0.7341	0.7804	0.7875	0.7889
きつい	0.5436	0	0.0408	0.6563	0.7458	0.7649	0.7729
寂しい	0.5275	0.1157	0.0725	0.6263	0.6201	0.6352	0.6666
厳しい	0.4886	0.4846	0.2039	0.5571	0.5592	0.6026	0.6209
micro	0.6932	0.5102	0.5622	0.7327	0.8081	0.8280	0.8456
macro	0.6709	0.4966	0.5227	0.7230	0.7903	0.8107	0.8283

TABLE 5 – Comparison of different translation systems and configurations

In addition, we performed a two-tailed paired T-test on the average accuracies of different weighting factors. The T-test results in TABLE 6 show that different weighting factors can have a statistically significant impact (bold text results) on system performance.

Word	$A_T(PMI_{Exp})$	$A_T(PMI_{Exp})$ $+A_S(D)$	T-test p-value	$A_T(PMI_{Exp})$ $+A_S(D)$	$A_T(PMI_{Exp})$ $+A_S(D+F)$	T-test p-value
明るい	0.9067	**0.9198**	1.86E-24	0.9198	**0.9212**	0.00886
甘い	0.8431	**0.8608**	2.33E-25	0.8608	**0.8876**	7.05E-33
暖かい	0.7988	**0.8183**	3.15E-23	0.8183	**0.8600**	6.85E-34
丁寧	0.8287	**0.8679**	3.16E-34	0.8679	**0.8711**	2.51E-06
冷たい	0.9119	**0.9210**	7.07E-16	0.9210	**0.9386**	1.42E-26
薄い	0.9088	**0.9293**	2.03E-29	0.9293	**0.9549**	3.5E-36
綺麗	0.7804	**0.7875**	9.85E-10	0.7875	0.7889	0.120067
きつい	0.7458	**0.7649**	2.77E-20	0.7649	**0.7729**	1.16E-13
寂しい	0.6201	**0.6352**	7.97E-21	0.6352	**0.6666**	4.46E-35
厳しい	0.5592	**0.6026**	3.16E-30	0.6026	**0.6209**	1.02E-26
micro	0.8081	**0.8280**	1.63E-38	0.8280	**0.8456**	2.33E-40
macro	0.7903	**0.8107**	2.62E-38	0.8107	**0.8283**	2.31E-39

TABLE 6 – T-tests on the average accuracies of different weighting factors

5 Discussion

In this section, we discuss the causes of some common errors that our system made.

5.1 Errors caused by Japanese homonyms

Japanese has many homonyms—words that share the same pronunciation but have different meanings and kanji. For example, the words 蜜柑 (orange), 未完 (unfinished), and 未刊 (unpublished) all share the same hiragana spelling みかん (orange) but are represented by different kanji. When calculating the *Association*$_T$ of hiragana words, the co-occurrence frequency of opinion word/related word pairs may be overestimated.

5.2 Limitations of single word-pair association scores

In the sentence, 暑い時期は駅からの距離が**きつい**です (When it is hot, walking the distance from the station is tiring.), which describes the hotel location, the two most likely Chinese translations for the opinion word きつい are 累人 (tiring) and 擁擠 (crowd), the former being the correct choice. The *Association*$_T$ between 累人 (tiring) and 距離 (distance) is incorrectly calculated as being lower than that between 擁擠 (crowd) and 距離 (distance). To determine the correct translation in cases like this, we should calculate *Association*$_T$ between pairs of related words and the opinion word. In the above example, if we consider 暑い (hot) and きつい (tiring) as one entity and calculate the *Association*$_T$ between 距離 (distance) and 暑い-きつい (hot-tiring), we get the correct translation.

5.3 Target opinion word associated with multiple feature words

If the target opinion word can apply to multiple product feature words in a sentence, the incorrect pairing may end up with the highest *FOA*. For example, in the sentence エントランスが**明るく**て従業員の対応も素敵でした。 (The entrance was bright and the staff's attitude was also friendly.), the opinion word 明るい (bright) can describe both feature words エントランス

(entrance) and 従業員 (staff). In this case our system calculates a higher *FOA* for the 明るい-従業員 (staff-bright) pair.

Conclusion

In this paper, we propose an unsupervised opinion word translation disambiguation scoring method which uses dependency distance and feature-opinion association as weighting factors. The scores of an opinion word's translation and its surrounding words' translations are estimated using Google search snippets. In our experiments, we focused on translation of hotel reviews from Japanese to Chinese. From a dataset of hotel reviews compiled from Rakutan Travel, we selected the top-10 most common polysemous Japanese opinion words to evaluate the performance of our system. The results show that our scoring method for representing the influence of product features and dependency distance improves translation accuracy effectively. Compared to Google Translate and Excite translation system, our system can translate opinion words more accurately, which could be of benefit to Chinese online shoppers seeking accommodations in Japan.

Acknowledgments

We would like to thank the two anonymous reviewers for their valuable comments, which have greatly improved the presentation of this paper. This work was supported in part by the Taiwan National Science Council under Grants NSC 98-2221-E-155-047 and NSC 99-2628-E-155-004.

Reference

Beeferman, D., Berger, A. and Lafferty, J. (1997). *A model of lexical attraction and repulsion*. Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics and Eighth Conference of the European Chapter of the Association for Computational Linguistics, Madrid, Spain.

Brosseau-Villeneuve, B., Nie, J.-Y. and Kando, N. (2010). *Towards an optimal weighting of context words based on distance*. Proceedings of the 23rd International Conference on Computational Linguistics (COLING 2010), Beijing, China.

Carpuat, M. and Wu, D. (2005). *Evaluating the Word Sense Disambiguation Performance of Statistical Machine Translation*. Proceedings of the 2nd International Joint Conference on Natural Language Processing (IJCNLP 2005), Jeju Island, Korea.

Chan, Y. S. and Ng, H. T. (2005). *Scaling Up Word Sense Disambiguation via Parallel Texts*. Proceedings of the 20th National Conference on Artificial Intelligence (AAAI 2005), Pittsburgh, Pennsylvania.

Chklovski, T., Mihalcea, R., Pedersen, T. and Purandare, A. (2004). *The SENSEVAL-3 Multilingual English-Hindi Lexical Sample Task*. Proceedings of the third International Workshop on the Evaluation of Systems for the Semantic Analysis of Text (SENSEVAL-3), Barcelona, Spain.

Church, K. W. and Hanks, P. (1990). Word association norms, mutual information, and lexicography. *Computational Linguistics*, *16*(1), 22-29.

Dagan, I. and Itai, A. (1994). Word Sense Disambiguation Using a Second Language Monolingual Corpus. *Computational Linguistics*, *20*(4), 563-596.

Gao, J., Zhou, M., Nie, J.-Y., He, H. and Chen, W. (2002). *Resolving Query Translation Ambiguity using a Decaying Co-occurrence Model and Syntactic Dependence Relations*. Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2002), Tampere, Finland.

Ikeda, K., Yanagihara, T., Matsumoto, K. and Takishima, Y. (2009). *Unsupervised Text Normalization Approach for Morphological Analysis of Blog Documents*. Proceedings of the 22nd Australasian Joint Conference on Advances in Artificial Intelligence, Melbourne, Australia.

Jin, P., Wu, Y. and Yu, S. (2007). *SemEval-2007 Task 5: Multilingual Chinese-English Lexical Sample*. Proceedings of the 4th International Workshop on Semantic Evaluations (SemEval 2007), Prague, Czech Republic.

Kudo, T. (2005). MeCab: Yet Another Part-of-Speech and Morphological Analyzer. Retrieved May, 2010, from http://mecab.sourceforge.net/

Kudo, T. and Matsumoto, Y. (2002). *Japanese Dependency Analysis using Cascaded Chunking*. Proceedings of the 6th Conference on Natural Language Learning (CoNLL 2002), Taipei, Taiwan.

Lefever, E. and Hoste, V. (2010). *SemEval-2010 Task 3: Cross-lingual Word Sense Disambiguation*. Proceedings of the 5th International Workshop on Semantic Evaluation (SemEval 2010), Los Angeles, California.

Liu, P., Xue, Y., Li, S. and Liu, S. (2010). *Minimum Normalized Google Distance for Unsupervised Multilingual Chinese-English Word Sense Disambiguation*. Proceedings of the 4th International Conference on Genetic and Evolutionary Computing (ICGEC 2010), Shenzhen, China.

Liu, P. and Zhao, T. (2009). *Unsupervised Translation Disambiguation Based on Maximum Web Bilingual Relatedness: Web as Lexicon.* Proceedings of the 6th International Conference on Fuzzy Systems and Knowledge Discovery (FSKD 2009), Tianjin, China.

Marsi, E., Lynum, A., Bungum, L. and Gambäck, B. (2011). *Word Translation Disambiguation without Parallel Texts.* Proceedings of the International Workshop on Using Linguistic Information for Hybrid Machine Translation (LIHMT 2011), Barcelona, Spain.

Ng, H. T., Wang, B. and Chan, Y. S. (2003). *Exploiting Parallel Texts for Word Sense Disambiguation: An Empirical Study*. Proceedings of the 41st Annual Meeting on Association for Computational Linguistics (ACL 2003), Sapporo, Japan.

Tsunakawa, T. and Kaji, H. (2010). *Augmenting a Bilingual Lexicon with Information for Word Translation Disambiguation*. Proceedings of the 8th Workshop on Asian Language Resources (ALR 2010), Beijing, China.

On-line Trend Analysis with Topic Models: #twitter trends detection topic model online

JeyHan Lau[1,2] *Nigel Collier*[3] *Timothy Baldwin*[1,2]
(1) Dept of Computing and Information Systems, The University Melbourne, Australia
(2) NICTA Victoria Research Laboratory, Australia
(3) National Institute of Informatics, Japan
jhlau@csse.unimelb.edu.au, collier@nii.ac.jp, tb@ldwin.net

ABSTRACT
We present a novel topic modelling-based methodology to track emerging events in microblogs such as Twitter. Our topic model has an in-built update mechanism based on time slices and implements a dynamic vocabulary. We first show that the method is robust in detecting events using a range of datasets with injected novel events, and then demonstrate its application in identifying trending topics in Twitter.

KEYWORDS: Topic Model, Twitter, Trend Detection, Topic Evolution, Online Processing.

Proceedings of COLING 2012: Technical Papers, pages 1519–1534,
COLING 2012, Mumbai, December 2012.

1 Introduction

In recent years, microblogs such as Twitter (http://www.twitter.com/) have emerged as a highly popular form of social media. Microblogs provide a platform for users to post short messages (a.k.a. "tweets") for followers to read in an on- or off-line fashion. One of the major attractions of microblogs is their contextual immediacy, i.e. they provide "here and now" access to users, including the potential to geotag tweets for the location of the post. Twitter is used in a variety of ways, from posting about personal daily life events to finding jobs to keeping up to date with news events.[1]

As microblogging services have gained in popularity, a myriad of applications that perform analysis of what is "trending" at a given point in time have emerged, with examples including Trendsmap (http://trendsmap.com/), What the Trend (http://whatthetrend.com/), twinitor (http://www.twinitor.com/) and Twendr (http://twendr.com/). There are a number of reasons for tracking trends. At the end-user level, it provides a way for users to identify popular discussions to follow or participate in. From a social science perspective, it provides insights into how new trends emerge, the half-life of trends, and the types of topics commonly discussed in the Twittersphere, etc.

Applications that track trends commonly use a keyword-based approach, and provide output in the form of simple terms, hashtags or term *n*-grams. While these keywords are indicative of the subject of the trending topic, ultimately they fall short in providing users with a fine-grained insights into the nature of the event. This motivated us to look for an alternative means of analysing and presenting trends, and ultimately led us to look at topic models as a potential solution.

Our contributions in this work are as follows. We first describe a topic model that processes documents in an on-line fashion. The model has the important properties that it does not grow over time, and can cope with dynamic changes in vocabulary. We then describe a method to measure shifts in the topic model, in order to track emerging events. We demonstrate the robustness and accuracy of the model using a suite of synthetic datasets based on Twitter and data from the TREC Topic Detection and Tracking (TDT) corpus, and then apply it to a series of Twitter feeds to detect popular topics in particular locations, which we find closely track local and global news events. The associated topics, in the form of a multinomial distribution over terms, are also more descriptive than single hashtags or strings.

2 Background

In recent years, there has been a surge of interest in event detection, due to the ready accessibility of document streams from newswire sources and social media. It has seen applications in many areas, such as the tracking of influenza (Signorini et al., 2011) and harvesting of spatio-temporal information for forest fires (De Longueville et al., 2009). Event detection occurs in two forms: (1) retrospectively, assuming the full document collection as input; and (2) on-line, processing documents dynamically as they arrive.

Retrospective event detection in microblogs provides insights about events that occurred in static sets of historical data. Much of the early work on retrospective event detection took place in the context of the TREC Topic Detection and Tracking (TDT) task (Allan, 2002), e.g. using document clustering and anomaly detection methods. If we wish to detect events happening presently in

[1] http://webtrends.about.com/od/twitter/a/why_twitter_uses_for_twitter.htm

our time, however, we require on-line event detection models. An example application where real-time responsiveness is critical is earthquake detection (Sakaki et al., 2010), and trend analysis also clearly requires on-line processing in order to be of use (Mathioudakis and Koudas, 2010). Most on-line approaches, however, use a relatively simple keyword-based methodology over a pre-defined set of keywords (Culotta, 2010; Lampos and Cristianini, 2010; Weng and Lee, 2011; Zhao et al., 2012) rather than tackling the more challenging task of open-world event detection.

Real-time first story detection (Petrović et al., 2010; Osborne et al., 2012) is the task of detecting the mentions of a breaking story as close as possible in time to its first mention. Here, the system should ideally pick up on the breaking story within seconds or minutes of its first mention in order to have impact, e.g. as an alert system for a newswire agency or intelligence organisation. As such, the methods that are standardly applied to the task tend to be based on analysis of local "burstiness" in the data, e.g. through locality sensitive hashing. Our work differs from theirs in that we wish to identify trends or topics that occur with a significant proportion in the data—which is different from trying to detect the very first mention of these topics. In our case, timeliness of detection is not as critical.

Bursty term analysis has obvious limitations in identifying events, both in that it fails to capture the fact that multiple terms may be involved with the same event (Zanzotto et al., 2011), and requires that at least one term undergoes a sufficiently high jump in relative frequency that the event can be identified. Topic models have been proposed as a means of better capturing events, by way of learning *clusters* of terms that are associated with a given event, as well as modelling changes in term co-occurrence rather than just term frequency. Most work based on topic modelling has been in the form of retrospective event detection models, however (Kireyev et al., 2009; Diao et al., 2012).

Moving to the more general area of the machine learning, several online topic models have been proposed (Hoffman et al., 2010; AlSumait et al., 2008). Hoffman et al. (2010) introduced an online LDA variant that uses variational Bayes as the approximate posterior inference algorithm. The model that is closest in spirit to what we propose is On-Line LDA (OLDA) (AlSumait et al., 2008). Using collapsed Gibbs sampling for approximate inference, OLDA processes documents in an on-line fashion by resampling topic assignments for new documents using parameters from a previously learnt model. We return to compare OLDA with our proposed method in Section 3.3.

3 Methodology

We first provide background on LDA topic modelling in Section 3.1. Next we describe our proposed online variant of LDA in Section 3.2, and contrast it with Online LDA in Section 3.3. Lastly, we explain how our topic model can be used to detect emerging topics in Section 3.4.

3.1 LDA Topic Model

LDA is a generative model that learns a set of latent topics for a document collection (Blei et al., 2003). The input to LDA is a bag-of-words representation of the individual documents, and the output is a set of latent topics and an assignment of topics to every document in the collection. Formally, a topic is a multinomial distribution of words, and a document is associated with a multinomial distribution of topics. A summary of LDA variables is presented in Table 1.

In topic models, the generative process for a word is as follows: first choose a topic, then sample

Variable	Dimension and Type	Description
T	Integer	Number of topics
W	Integer	Number of unique words (vocabulary)
D	Integer	Number of documents
N	Integer	Number of tokens
θ	$D \times T$ of probabilities	Topic distribution in documents
ϕ	$T \times W$ of probabilities	Word distribution in topics
$\boldsymbol{\alpha}$	$D \times T$ of α priors	Dirichlet prior for θ
$\boldsymbol{\beta}$	$T \times W$ of β priors	Dirichlet prior for ϕ
$\boldsymbol{w}$	N-Vector of word identity w	Words in documents
$\boldsymbol{z}$	N-Vector of topic assignment z	Topic Assignment of Words

Table 1: A summary of variables used in LDA.

a word from the given topic. Blei et al. (2003) introduced Dirichlet priors to the generative model, and used variational Bayes to learn θ and ϕ by maximising the probability of words in the collection. Griffiths and Steyvers (2004) proposed using collapsed Gibbs sampling to do approximate inference by considering the posterior distribution over the assignments of words to topics ($P(\boldsymbol{z}|\boldsymbol{w})$). Summarising the derivation steps, the update function in the sampling process for a new topic assignment of a word can be expressed as follows:

$$P(z = t|\boldsymbol{z}, \boldsymbol{w}, \boldsymbol{\alpha}, \boldsymbol{\beta}) \propto \frac{n(d,t) + \alpha}{n(d,\cdot) + T\alpha} \frac{n(t,w) + \beta}{n(t,\cdot) + W\beta}$$

where $n(d,t)$ is the number of assignments of topic t in document d, and $n(t,w)$ is the number of assignments of word w to topic t; all counts exclude the current assignment z.

3.2 Online Processing Variant

LDA processes the data in a single batch to learn the topic assignments. To facilitate the processing of streamed text, we need a model that: (1) processes the input and updates the model periodically; (2) produces topics that are comparable for different periods so that topic shift/evolution is measurable; and (3) does not grow in size with time (to ensure that it stays sensitive to topic changes over time).

We first introduce a few concepts needed for the model. Time in the model is discretised into slices, and documents (i.e. the input data) are partitioned into time slices. For example, a time slice can be an hour, a day, or a year. Denoting each time slice as k_t, k_0 is the first time slice. L is a sliding window that keeps documents for a fixed number of time slices. As documents in the new time slice arrive, documents in older time slices are discarded, so that length of the window, $|L|$, remains constant. The rationale of this approach is that we require a model that is constant in size; storing the complete document stream history data would cause the model to grow indefinitely over time, and become increasingly insensitive to topic changes.

At the arrival of new documents for time slice k_{t+1}, we update the model by resampling the topic assignments $\boldsymbol{z}$ for all documents in window L (Equation 1), using θ and ϕ from the previous model in slice k_t to serve as Dirichlet priors $\boldsymbol{\alpha}'$ and $\boldsymbol{\beta}'$ in the new model in slice k_{t+1}. The contribution factor, c, determines the degree of contribution of learnt parameters to the priors of the new model. c ranges from 0 to 1: $c = 0$ means the model is run without using any previously learnt parameters. The introduction of c is key in enabling the model to have a set of constantly evolving topics. In other words, it dampens the rich-gets-richer dynamic of the Chinese Restaurant Process in LDA.

1. Initial step:
   (a) Set Dirichlet priors α_0 and β_0; topic number T; contribution factor c; time slice k; and window size $|L|$;
   (b) Given $|L| = l$, window L contains documents from slices $k_0, ..., k_{l-1}$;
   (c) Run LDA for documents in window L;
2. Iterative step for each k_{l+i}:
   (a) Add documents in slice k_{l+i} to window L;
   (b) Remove documents in slice k_i from window L, updating θ and ϕ from the previous model as necessary;
   (c) Re-generate vocabulary for documents in window L;
   (d) Calculate priors $\boldsymbol{\alpha}'$ and $\boldsymbol{\beta}'$ as per Section 3.2;
   (e) Resample $\boldsymbol{z}$ using $\boldsymbol{\alpha}'$ and $\boldsymbol{\beta}'$ for documents in window L as per Equation 1.

Table 2: Work flow of the online processing model.

As a true online model, it would not be appropriate to assume a fixed vocabulary across time. The importance of having a dynamic vocabulary is motivated by the fact that we are interested in detecting emerging events in the data stream, where new words are likely to appear and be associated with new events (e.g. in the form of names of key people or places). To accommodate this, at every update we process the documents in the time window to re-generate the vocabulary, removing words that fall below a pre-defined frequency threshold and adding new words that now satisfy it.[2]

The Dirichlet priors $\boldsymbol{\alpha}'$ and $\boldsymbol{\beta}'$ in the new model in slice k_{t+1} are calculated as follows:

For previously seen documents and words:

$$\alpha'_{dt} = \frac{n(d,t)}{N_{old}} \times D_{old} \times T \times \alpha_0; \qquad \beta'_{tw} = \beta_0 \times (1-c) + \frac{n(t,w)}{N_{old}} \times T \times W_{new} \times \beta_0 \times c \tag{1}$$

For new documents and words:

$$\alpha'_{dt} = \alpha_0; \qquad \beta'_{tw} = \beta_0$$

where α'_{dt} is the prior for topic t in document d; β'_{tw} is the prior for word w in topic t; $n(d,t)$ and $n(t,w)$ are counts from the previous model in slice k_t; α_0 and β_0 are the default uniform prior values for θ and ϕ; and D_{old}, N_{old} and W_{new} are the number of previously processed documents, number of tokens in those documents and number of vocabulary, respectively, in time window L. The rationale behind the normalisation approach is to maintain a constant sum of priors across different batches of processing, i.e. $\sum \boldsymbol{\alpha}' = \sum \boldsymbol{\alpha} = D \times T \times \alpha_0$, and $\sum \boldsymbol{\beta}' = \sum \boldsymbol{\beta} = T \times W \times \beta_0$.

The work flow of the model is presented in Table 2.

3.3 Comparison with On-line LDA

One key difference between our proposed model and On-line LDA (OLDA) is the transfer of parameters from a previously learnt model to the updated model. In OLDA, ϕ counts from the

[2] In all our experiments, we filter our all words that occur less than 10 times across all documents in the window.

previous model are used directly or normalised as priors in the new model. To avoid topics converging after a number of iterations, OLDA removes the the $\boldsymbol{\beta}$ prior counts in ϕ of the previous model before importing them into the new model. Our model handles the converging issue in a more elegant way, by introducing a parameter (contribution factor, c) to dampen the influence from the previous model.

The second difference is that OLDA assumes a fixed vocabulary. While this is a convention in topic models, it is not an appropriate assumption for a genuinely online application, where it is impossible to pre-calculate the vocabulary ahead of time. As emerging events are likely to contain critically-relevant phrases and terms (e.g. the name of a hitherto-unknown key figure in an event, the name of a natural disaster, or the little-known location of an event), the vocabulary of our proposed model is re-generated at each update: new words are added and previously-seen words that drop below our frequency threshold are removed.

3.4 Detection of Novel Topics

At every model update, the word distribution in topics (i.e. ϕ) changes, however a one-to-one correspondence between topics is maintained across adjacent updates (provided that $c \neq 0$). Topics can thus be viewed as constantly evolving as new documents are processed: topics that are rarely or not observed in the updated document set will fade away, replaced by newly-emerged topics.

To detect these novel topics, we calculate the degree of change, or, the *evolution* of a topic using the Jensen-Shannon divergence measure between the word distribution of each topic t before and after an update, and classify a topic as being *novel* if the measure exceeds a threshold.

4 Synthetic Dataset Experiments

4.1 Generation of Synthetic Data

Ultimately we are interested in applying our method to real-world document streams, ideally based on a microblog such as Twitter. For evaluation purposes, however, we require a document stream where we have document-level annotations of: (1) whether it mentions an event of potential interest; and (2) if it mentions an event, what that event is (in the form of an event ID, potentially shared across multiple documents over a time period). In the absence of such a dataset, and given the prohibitive expense in exhaustively annotating such a dataset over the volume of data that comes through Twitter, we created a suite of synthetic datasets. Other than annotation cost, one advantage of using a synthetic dataset is that it gives us the flexibility to generate events with different distributional properties.

Having said that we are resigned to generating a synthetic dataset, we want the data to mimic as closely as possible the actuality of event mentions on Twitter. To this end, we take a document stream from Twitter and replace the message content of tweets nominally relating to a particular event (based on hashtag analysis) with distinct tweet-length mentions of events from the Topic Detection and Tracking corpus (TDT3)[3] dataset. That is, our datasets contain "background events" in the original Twitter document stream that we don't have annotations for, and "novel events" from TDT that we do have annotations for, and that form the basis of our evaluation.

In detail, the following steps were taken to generate the background event dataset:

[3] `http://projects.ldc.upenn.edu/TDT3/`

Event Type	Document Content
Background	@allabouttaurus : be realist we see thing for what they be not what they could be .
Background	ugh i be go to be so sore tomorrow
Background	rt @pagswagxo : next status i see about m . burn and i be gonna go insane .
Background	! ! saatnya mencarus wifi supaya ipod touch gw conect ke internet , dan ngetweet via twitter for iph one ,
Novel	the kosovo information center claim serb police be pass out weapon to serb civilian in the region .
Background	rt @rickyricchi : rt @atikaftri : jan lupa nya jan lupa juga mention yaw ^ ^
Background	@1aurenheilman lol , do you spell "pet peeve " wrong on purpose ?
Background	had2let it be know ! & thanks for txtn back -___ - rt @phliwidapencil lmfao rt @skrillafoccapo : all big booty aint good big booty ! !
Background	rt @desintadict_cb : rtif u want follower (cont) http ://t.co/joej7wfz
Background	well i know where all my christmas money be go . municipal court of jasper .

Table 3: An example of 10 documents in the synthetic dataset for KIM-MILOSEVIC (mapped TDT3 Topic: Holbrooke-Milosevic Meeting).

1. Collect Tweets from September 2011 to January 2012.[4]
2. Identify a set of hashtags that occur over 80% of the days in this 5-month period.[5]
3. Designate these hashtags as the set of candidates for the background events. The treatment of these hashtags as background events is based on the assumption that hashtags are generally associated with events or topics in Twitter.
4. Randomly select a subset of hashtags from the candidate set, biased according to frequency, i.e. popular hashtags are more likely to be selected than rarely-used hashtags.
5. Extract out all messages tagged with the selected hashtags for a given time period (see below), and remove the hashtag.

The novel events were embedded into the dataset as follows:

1. Select a news event that occurred during the time of the crawl, across a range of genres ranging from natural disasters to celebrity deaths to terrorist attacks (see Table 4 for details).
2. Identify a set of Twitter hashtags that correspond to the particular news event. Tweets that contain these hashtags constitute the individual novel documents.
3. Select a TDT3 topic that is of a similar genre to the news event (to keep the injected documents as similar as possible in content to the messages they replace). All topics were events that took place in 1999.
4. Replace each identified tweet with a sentence sampled randomly from the TDT3 documents labelled with the selected event topic (removing the original hashtag in the process).

Our justification for this procedure is two-fold:

- *to generate tweet-level annotations for each event, across a range of event genres* — While the original tweets had one of the pre-identified hashtags, they could have been spam or hijacking the primary topic associated with the hashtag (i.e. we are attempting to ensure the *precision* of the annotations relative to a given event);

[4]We collected the Tweets via the Twitter Streaming API: `https://dev.twitter.com/docs/streaming-api/methods`. The corpus contains approximately 12 million tweets, spanning 1.39 million users.

[5]In this case, we assume that hashtags represent events or topics. This assumption is used only as a means to simplify the process for synthesising dataset. Note that these hashtags (that occur over 80% of the days in the period) are generally popular topics that are frequently discussed and may not necessarily be news events.

- *to guard against the possibility that the background tweets relate to injected event* — In the original dataset, it is highly likely that there are tweets which mention the original news event but aren't tagged with one of the pre-identified hashtags. It is unlikely, however, that there would be background tweets which mention an event from 1999 (i.e. we are attempting to ensure the *recall* of the annotations relative to a given event).

In doing a one-for-one replacement of an original tweet with a TDT3 sentence and maintaining the original timestamp of the tweet, we are additionally achieving an event propagation distribution which is as faithful as possible to actual event mentions in Twitter.

In total, we created 5 datasets, each with a single novel event and 25–150 background events. We initially experimented with 50 background events but later varied the number of background events—hence the variation in the number of background events in the synthetic dataset. Each dataset spans over a period of 9 days, with the novel event injected on the 5th day (and subsequent days at a level defined by the frequency of the original hashtags).[6]All documents are stopped and lemmatised.[7] An example of a sample of generated documents is displayed in Table 3.[8] A summary of the novel events, mapped TDT3 topics and other metadata is presented in Table 4.

4.2 Experiments

In all experiments, we set the time slice to 1 day and length of window $|L|$ to 2 days. Note, however, that the time slice setting is flexible: should more temporally fine-grained analysis be required, the time slice can be set to a shorter time frame, e.g. 1 hour.[9] We set the contribution factor c to 0.5, meaning old and new documents have approximately equal weighting in the 2-day window. We set α_0 to 0.001; a low value is preferred to produce a very sparse topic distribution over documents (each document, i.e. tweet or sentence from TDT3, should be assigned to no more than a few topics). For β_0, we use 0.01. We vary the setting of T, as detailed in Section 4.2.1.

On a daily basis, a new batch of documents is added to window L and the model is updated. At the end of every update, we calculate the topic evolution score for every topic (as per Section 3.4), and identify topics that exceed a set threshold as *novel topics*.[10] Each document in the model is associated with a distribution of topics. To determine the *novel documents*—documents that contain novel topics—we select those that have a novel topic (i.e. topics which have a topic evolution score above the threshold) assigned as its highest-probability topic. Note that only documents from the new time slice can be novel documents.

As the true set of documents that contain the injected novel event are known — they are all

[6]By injecting the novel event on the 5th day we mean to select a period such that the first occurrence of the novel event will always fall on the 5th day in the period. As such, the natural distribution for the novel event is preserved.

[7]We use OpenNLP for POS tagging and Morpha for lemmatisation (Minnen et al., 2001).

[8]The "language" of the novel event may seem different from a standard tweet, but we contend that the topic model does not gain any real advantage from this as the model is not attuned to the quality of the language in the documents. Also, the Twitter stream contains news releases from news and government organisations.

[9]The main bottleneck for shorter time slices is simply the number of documents, in that if the number of documents becomes too few, the topic model is unlikely to model the data well. Note that the time taken to process a 24-hour time slice on a somewhat high-end single-processor Linux machine is of the order of 15 minutes.

[10]We additionally introduce the constraint that novel topics must not contain user mention tokens (i.e. words of @XXXX format) in their top-10 topic words. This constraint is created on the observation that topics that contain user mentions in their top-10 topic words are usually not semantically coherent and hence do not constitute novel topics.

Event Name	VAN-MITCH			
News Event	Van Earthquake, Turkey			
Date of Occurrence	23 October 2011			
Mapped TDT3 Topic	30002: Hurricane Mitch			
Number of Background Events	25	50	100	150
Proportion of Novel Documents on Date of Occurrence	0.4989 (/15611)	0.2403 (/32371)	0.1232 (/63209)	0.1100 (/70749)
Event Name	WASHI-MITCH			
News Event	Tropical Storm Washi, Philippines			
Date of Occurrence	17 December 2011			
Mapped TDT3 Topic	30002: Hurricane Mitch			
Number of Background Events	25	50	100	150
Proportion of Novel Documents on Date of Occurrence	0.1469 (/8203)	0.0452 (/26656)	0.0222 (/54200)	0.0223 (/54098)
Event Name	LIÈGE-PINOCHET			
News Event	Liège Murder-suicide Attack, Belgium			
Date of Occurrence	13 December 2011			
Mapped TDT3 Topic	30003: Pinochet Trial			
Number of Background Events	25	50	100	150
Proportion of Novel Documents on Date of Occurrence	0.0857 (/8971)	0.0245 (/31296)	0.0106 (/72859)	0.0099 (/77494)
Event Name	KIM-MILOSEVIC			
News Event	Death of Kim Jong Il, North Korea			
Date of Occurrence	19 December 2011			
Mapped TDT3 Topic	30015: Holbrooke-Milosevic Meeting			
Number of Background Events	25	50	100	150
Proportion of Novel Documents on Date of Occurrence	0.3965 (/14614)	0.1468 (/39433)	0.0763 (/75858)	0.0750 (/77114)
Event Name	COSTA-SWISSAIR			
News Event	Costa Concordia Disaster, Italy			
Date of Occurrence	14 January 2012			
Mapped TDT3 Topic	30016: SwissAir111 Crash			
Number of Background Events	25	50	100	150
Proportion of Novel Documents on Date of Occurrence	0.1340 (/9875)	0.0418 (/31610)	0.0205 (/64404)	0.0182 (/72489)

Table 4: Metadata for the 5 synthetic datasets. Numbers in parentheses indicate the total number of documents on the day the event occurred.

TDT3 sentences — we can assess the effectiveness of the model by calculating the "tweet"-level precision, recall and F-score on the day the novel event occurred.

4.2.1 Detection of Novel Event over Varying Numbers of Topics T

The number of topics, T, is a key parameter in the model that affects the granularity of the topics. A high value of T allows the model to generate more specialised topics, while a low value of T generates higher level, more general concepts. We experiment with a range of T values to ascertain how sensitive the topic model is to the T setting, and attempt to arrive at a recommendation for an appropriate T setting for general-purpose applications.

In our initial experiments, we vary T and keep the number of background events constant at 50. A summary of the F-scores for the classification of novel documents is presented in Table 5. Encouragingly, we see that the topic model is able to detect the novel event with relatively high reliability when T is greater than 25 for all datasets. Bear in mind that these F-scores are at the message level not the topic level, and are predicated on the detection of the novel event via

Number of Topics T	VAN-MITCH	WASHI-MITCH	LIÈGE-PINOCHET	KIM-MILOSEVIC	COSTA-SWISSAIR
25	0.50	0.00	0.00	0.51	0.00
50	0.74	0.62	0.47	0.72	0.37
100	0.63	0.61	0.55	0.62	0.47
150	0.65	0.45	0.59	0.76	0.46

Table 5: F-scores with varying T (the number of background events is kept constant at 50).

Number of Background Events	VAN-MITCH	WASHI-MITCH	LIÈGE-PINOCHET	KIM-MILOSEVIC	COSTA-SWISSAIR
25	0.77	0.55	0.81	0.80	0.62
50	0.74	0.62	0.47	0.72	0.37
100	0.61	0.53	0.00	0.82	0.45
150	0.45	0.34	0.00	0.70	0.46

Table 6: F-scores with varying number of background events (T is kept constant at 50).

topic shifts. Beyond $T = 50$, the F-scores are largely similar, indicating the model's insensitivity to small changes in the T setting.

4.2.2 Detection of Novel Event with Varying Number of Background Events

In Section 4.2.1, we can observe that the F-scores are generally higher for datasets that have a greater proportion of novel documents (VAN-MITCH, KIM-MILOSEVIC). We similarly see the same trend in topic evolution score: VAN-MITCH and KIM-MILOSEVIC have higher $e(t)$ than LIÈGE-PINOCHET and COSTA-SWISSAIR. This implies that the proportion of novel documents in the data stream has an influence on the detection of topic(s) associated with the novel event.

To better understand this, we next keep T constant at 50 and vary the number of background events. Increasing the number of background events decreases the proportion of novel documents; effectively there will be more mixtures of topics in the data stream and thus it will be harder to detect the novel event.[11] F-scores for classifying the novel documents with varying number of background events are presented in Table 6.

We see that the injected novel event in all datasets except LIÈGE-PINOCHET is detected for all settings of the number of background events.[12] The proportion of novel documents in the LIÈGE-PINOCHET dataset is particularly low—0.0245 at 50 background events (Table 4). It is thus not surprising that the novel event is not detected when the number of background events is increased to 100: the proportion of novel documents drops to a mere 0.0106, the lowest out of all the datasets. Overall, the results are positive and the model has demonstrated its ability to detect novel events, even when the proportion of novel documents is as low as 0.0182 (COSTA-SWISSAIR at 150 background events).

For all the F-scores presented, a threshold over the topic evolution score $e(t)$ is required to determine the set of novel topics on each update. The threshold facilitates the classification of documents that are novel, and impacts directly on the F-score. To choose a suitable setting for the topic evolution score threshold, we plot a graph of F-scores of all datasets with varying number of topics T (Section 4.2.1) and background events (Section 4.2.2) against a range of topic evolution score threshold values in Figure 1. Based on Figure 1, we set the topic evolution

[11] The proportion of novel documents with varying number of background events is summarised in Table 4.

[12] We observe some fluctuation in the F-scores; this is quite possible as the background events are selected independently for each background event setting, i.e. the set of background events in BG=25 is not a subset of those in BG=50.

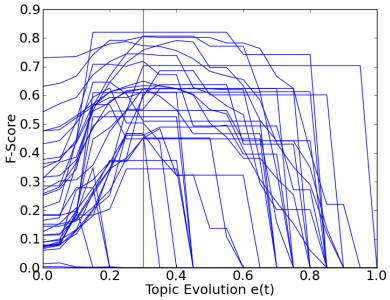


Figure 1: F-scores for the classification of novel documents against threshold values of topic evolution score $e(t)$. Each line represents a single dataset with a fixed number of topics T and background events. The vertical red line indicates the threshold at $e(t) = 0.3$

Injected Novel Events	WASHI-MITCH, LIÈGE-PINOCHET, KIM-MILOSEVIC		
Period	9 October 2012 – 23 October 2012		
Number of Background Events	50		
Novel Event	WASHI-MITCH	LIÈGE-PINOCHET	KIM-MILOSEVIC
Date of Occurrence	17 October 2012	13 October 2012	19 October 2012
Proportion of Novel Documents	0.0452	0.0245	0.1433
on Date of Occurrence	(/26681)	(/31312)	(/40386)

Table 7: Metadata for the synthetic dataset with multiple novel events.

score threshold to 0.3 based on the observation that it provides for consistent and generally optimal performance across all datasets. Note that the F-score results presented thus far are all based on this threshold value.

4.2.3 Detection of Multiple Novel Events

In the previous sets of experiments, each dataset had a single injected novel event. In a real-word setting, however, the number of novel events is unknown and multiple novel events can occur in a single time period. To further test the robustness of our system, and also test our claims about the "fluidity" of the topics under out model, we created another synthetic dataset with a number of novel events injected over a single time period, based on the dates of occurrence of the original events; details of the multi-novel-event dataset are presented in Table 7.

Setting the number of topics T to 50 and using 0.3 for the topic evolution score threshold, we obtain F-scores for novel document classification which are only slightly lower than those for the single-event experiments: 0.49, 0.60, and 0.59 for WASHI-MITCH, LIÈGE-PINOCHET and KIM-MILOSEVIC, respectively, on each event's date of occurrence.

5 Trend Detection in Twitter

The experiments to date have demonstrated the robustness and effectiveness of our methodology in detecting artificially-injected novel events. Ultimately, we are interested in applying the method to online analysis over a microblog such as Twitter to detect emerging trends or events in real-time. To this end, we ran our system for a month in February 2012, over tweets that

Date (UTC)	$e(t)$	Topic ID	Proportion	Topic Words
2012-02-05	0.55	95	0.0093	snow #uks london finally settle look #snow garden nom food
2012-02-06	0.91	256	0.0132	webb howard penalty unite chelsea #mufc game #cfc utd give
2012-02-09	0.77	49	0.0073	capello england fabio resign manager italian job sink ship #capello
2012-02-11	0.62	74	0.0156	suarez evra hand shake racist liverpool #lfc cunt #mufc win
2012-02-12	0.57	160	0.0097	whitney houston rip die dead omg sad amy r.i.p believe
2012-02-12	0.58	168	0.0101	whitney houston sad rip music r.i.p love bong voice remember
2012-02-12	0.61	197	0.0129	whitney houston rip sad r.i.p peace love voice #whitneyhouston song
2012-02-12	0.66	137	0.0134	whitney houston r.i.p rip sad die news #stalbans #harpenden dead
2012-02-13	0.57	91	0.0085	#bafta win film award bafta artist watch meryl #baftas love
2012-02-13	0.71	49	0.0122	zambium penalty ivory coast win #zambia cup zambia miss drogba
2012-02-14	0.46	81	0.0077	happy valentine love &lt xxx ;3&lt dear follow fan load
2012-02-17	0.47	20	0.0077	vagina #replacefilmtitleswithvagina lol war watch funny play movie love wear
2012-02-22	0.50	17	0.0072	win brit #britawards award adele artist british watch international woman
2012-02-22	0.56	251	0.0093	blur adele cut #brits love speech brit sound shit song
2012-02-26	0.60	289	0.0079	goal walcott van super arsenal persie #arsenal wait score game

Table 8: Top-15 trending topics in London in February 2012.

were returned for a geospatially-bound crawl of data from London and New York. We took the two sets of tweets and calculated the topic evolution score $e(t)$ to discover trending topics each day. We set $T = 300$ in each case (to deal with the larger volume of messages as compared to our synthetic experiments). Other parameter values were identical to those used in the synthetic dataset experiments in Section 4.

We display the top-15 February 2012 trending topics in London and New York in Tables 8 and 9, respectively. We filtered out any topics that occurred over less than 0.7% of the stream to show only the popular trends. Note that it is significant that we present both a date and a topic ID: it is possible for the same topic to shift significantly in content at multiple time slices across the topic-modelled time period, although we don't observe this in our limited display of results in Tables 8 and 9.

In London, there is much discussion about soccer (topic IDs 95, 256, 49, 74, 289). A search using the topic words reveal that these topics correspond to real soccer news events. To give a few examples, topic ID 256 is about the controversial penalties awarded by Howard Webb to Chelsea in a Manchester United vs. Chelsea match, topic ID 49 relates to the the resignation of Fabio Capello as England's manager, topic ID 74 is about Suárez refusing to shake Evra's hand before kick off, and topic ID 49 corresponds to Zambia's first victory in the Africa Cup of Nations.

Whitney Houston's death also triggered a massive reaction from Twitter users in London, so much so that it appears in multiple topics (topic IDs 160, 168, 197, 137). Reading over the topics, the difference in these topics seem indistinguishable, and the fact that it occurs across topics is a function of the sheer volume of traffic that mentioned the event; in a deployed setting, it would be relatively trivial to pick up on the fact that the topics are almost identical (Newman et al., 2010; Mimno et al., 2011). Lastly, entertainment award shows are another popular topic in the Twittersphere (topic ID 91, 17, 251), in the form of the BAFTA and Brits awards.

In New York, rather than soccer, American football is the dominant sport and there was a lot of talk of the Super Bowl (topic IDs 267, 50, 88, 60, 207). Similarly to London, Whitney Houston's death drew much attention, split across a number of largely-indistinguishable topics (topic IDs 51, 223, 45), with the exception of topic ID 250, which is related to her funeral. Entertainment award shows are again a popular topic, although New Yorkers are more interested in the Grammy's and Oscars (topic IDs 265, 227, 4, 290, 246).

Date (UTC)	$e(t)$	Topic ID	Proportion	Topic Words
2012-02-06	0.40	267	0.0088	giant 2012 superbowlpocalypse superbowl york fan win move championapocolypse target
2012-02-06	0.46	50	0.0071	#giants #superbowl giant win fuck die touchdown #giantsnation pat root
2012-02-06	0.50	88	0.0081	giant win #superbowl patriot wear shirt jersey fan superbowl today
2012-02-06	0.53	60	0.0101	super bowl giant 2012 champion york fan move target sunday
2012-02-06	0.69	207	0.0089	brady tom elus #superbowl #giants manning giant win catch game
2012-02-12	0.54	51	0.0112	whitney houston rip sad die love #whitneyhouston music r.i.p #rip
2012-02-12	0.58	223	0.0108	whitney houston die rip dead r.i.p sad bobby singer lol
2012-02-12	0.63	45	0.0109	whitney houston rip sad r.i.p dead die omg wow damn
2012-02-13	0.40	265	0.0081	adele win #grammys grammy award tonight watch congrat game artist
2012-02-13	0.42	227	0.0096	chri brown #grammys rihanna bobby love coldplay grammy performance sing
2012-02-13	0.51	4	0.0077	minaj nickus performance nicki #grammys wtf adele lol grammy perform
2012-02-14	0.64	273	0.0125	valentine happy love single &lt today holiday word tomorrow heart
2012-02-19	0.50	250	0.0083	whitney houston rip love #whitneycnn kevin costner funeral r.i.p carter
2012-02-27	0.59	290	0.0075	meryl #oscars streep win violum oscar bradley cooper love lin
2012-02-27	0.76	246	0.0082	#oscars win octavium oscar spencer speech love jlo look dress

Table 9: Top-15 trending topics in New York in February 2012.

In both locations, a new topic for Valentine's Day emerged on February 14th (topic IDs 81 (London) and 273 (New York)).

6 Discussion

The motivation for using a topic model-based approach as opposed to a keyword-based approach is borne out in looking at the detected Twitter trends in Section 5. Some of the detected topics, such as the London football news event examples, would be difficult to capture in a single string of one to three words as used in conventional keyword-based approaches. With our topic model-based detection system, however, the details of an event are summarised more appropriately with a list of associated words.

We note that we did not compare our methodology against existing approaches, such as OLDA or keyword-based approaches. The reasoning for excluding a comparison against OLDA is that in preliminary experiments we found that OLDA was not very effective in detecting the novel topics, as the model uses a fixed vocabulary. Keyword-based approaches are incompatible with our task set up, as they assume knowledge of a fixed information need, which we don't have—our model is looking for the *unknown* in detecting novel events. It is possible to use bursty term analysis (in which case the keywords don't need to be pre-identified), but this often leads to a highly cryptic and potentially misleading representation of the novel event, unlike topic models.

In the synthetic dataset experiments, we measure the model's performance in detecting novel events by calculating the F-scores of novel document classification. The F-score result is a straightforward and objective evaluation, but it is an underestimate of the model's true performance for two reasons. For one, we are actually interested in the detection of the injected event as a newly emerged topic in the data stream rather than correctly classifying every document that contains the injected event. The latter task is the one we evaluate, and a significantly harder task than the former (in fact, if our tweet-level F-score is non-zero, it means we have been successful in detecting the event as a novel topic, meaning we have succeeded in all cases for $T \geq 50$ other than LIÈGE-PINOCHET with the number of background events ≥ 100).

The second reason is that the model could potentially pick up other genuine novel events that occurred in the *background Tweets* that were not related to the injected novel event. Manual inspection on a sample of discovered novel topics revealed that this indeed happened, and the novel topic is often the original news event which was replaced by a TDT3 topic. As an example,

Topic ID	Topic Words
6	kim call korea north jong-il die fire pussy library jong
9	nato kosovo milosevic president force strike crisis military problem unite
13	serb albanian kosovo ethnic kill police hundred rebel home province
19	milosevic nato kosovo holbrooke president richard yugoslav official envoy force
55	kosovo nato force troops milosevic president albanian iraq serb news
72	2011 blue news kosovo maldive top white refugee stand #egypt
73	nato milosevic kosovo military war official international house tax demand
84	kosovo force security monitor mission organization milosevic europe international agreement
86	kim north jong leader die korea korean #fail news christmas

Table 10: Detected novel topics in KIM-MILOSEVIC (50 background events and $T = 100$).

we present a list of detected novel topics for KIM-MILOSEVIC with 50 background events and $T = 100$ topics in Table 10. We see that topic IDs 6 and 86 are related to Kim Jong Il's death, the original news event which was replaced by TDT3's "Holbrooke-Milosevic Meeting" topic. As the documents containing Kim's death do not constitute as part of the *gold* novel document set that have the TDT3 event, the model is penalised for classifying these Tweets as novel documents.

Admittedly, scalability of the model has not been the focus in this work. One reason is that we were initially interested in investigating the accuracy performance of topic models in detecting emerging events, leaving optimisation for future work. The second reason is that for the purposes of tracking *popular* emerging trends, we do not necessarily have to process the full collection of Tweets, as these trends occur in a significant portion of the data. Given that we have set our sights on Twitter, however, this is an obvious area to focus future attentions, given that an estimated 250 million tweets were posted per day on Twitter in 2011. If we were to apply the method to the full Twitter feed, we would use a much finer-grained time granularity and run our method over a larger number of cores than at present (our implementation is already parallelised), and are confident of being able to keep pace with this much greater data volume. Our implementation can be accessed from `http://www.csse.unimelb.edu.au/~tim/etc/online_lda.zip`.

In terms of the sensitivity of the model when scaling down to smaller numbers of documents, in our Twitter trend detection experiments conducted for London and New York, we have demonstrated that even over relatively small numbers of documents —each location has less than 60,000 tweets per day — interesting popular trends and fine-grained news events can be detected.

7 Conclusion

We have proposed a novel topic model-based approach to on-line trend analysis. On every update, we calculate the evolution of topics to detect newly emerged topics in the document collection. We first applied the methodology to a suite of synthetic datasets and demonstrated the model's strength in detecting individual documents describing novel events, and moved on to process raw Twitter data to detect trending topics. The discovered trends were promising and gave insights to the popular culture and events discussed in the Twittersphere.

Acknowledgements

NICTA is funded by the Australian government as represented by Department of Broadband, Communication and Digital Economy, and the Australian Research Council through the ICT centre of Excellence programme. JHL's travel was supported by a Google PhD Travel Prize.

References

Allan, J. (2002). Introduction to topic detection and tracking. In Allan, J., editor, *Topic Detection and Tracking: Event-based Information Organization*, pages 1–16. Kluwer.

AlSumait, L., Barbará, D., and Domeniconi, C. (2008). On-line LDA: Adaptive topic models for mining text streams with applications to topic detection and tracking. In *Proceedings of the 2008 Eighth IEEE International Conference on Data Mining (ICDM-08)*, pages 3–12, Washington, DC, USA.

Blei, D., Ng, A., and Jordan, M. (2003). Latent dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022.

Culotta, A. (2010). Towards detecting influenza epidemics by analyzing Twitter messages. In *Proceedings of the First Workshop on Social Media Analytics*, pages 115–122, New York, NY, USA.

De Longueville, B., Smith, R., and Luraschi, G. (2009). "omg, from here, i can see the flames!": A use case of mining location based social networks to acquire spatio-temporal data on forest fires. In *Proceedings of the 2009 International Workshop on Location Based Social Networks*, pages 73–80, Seattle, Washington.

Diao, Q., Jiang, J., Zhu, F., and Lim, E. (2012). Finding bursty topics from microblogs. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL HLT 2012)*, pages 536–544, Jeju Island, Korea.

Griffiths, T. and Steyvers, M. (2004). Finding scientific topics. *Proceedings of the National Academy of Sciences*, 101:5228–5235.

Hoffman, M., Blei, D. M., and Bach, F. (2010). Online learning for latent dirichlet allocation. In *Advances in Neural Information Processing Systems 23 (NIPS-10)*, pages 856–864.

Kireyev, K., Palen, L., and Anderson, K. (2009). Applications of topics models to analysis of disaster-related Twitter data. In *NIPS Workshop on Applications for Topic Models: Text and Beyond*, Whistler, Canada.

Lampos, V. and Cristianini, N. (2010). Tracking the flu pandemic by monitoring the social web. In *2nd IAPR Workshop on Cognitive Information Processing (CIP 2010)*, pages 411–416, Montreal, Quebec.

Mathioudakis, M. and Koudas, N. (2010). TwitterMonitor: Trend detection over the Twitter stream. In *Proceedings of the 2010 ACM SIGMOD International Conference on Management of data*, pages 1155–1158, New York, NY, USA.

Mimno, D., Wallach, H., Talley, E., Leenders, M., and McCallum, A. (2011). Optimizing semantic coherence in topic models. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing (EMNLP 2011)*, pages 262–272, Edinburgh, UK.

Minnen, G., Carroll, J., and Pearce, D. (2001). Applied morphological processing of English. *Journal of Natural Language Processing*, 7(3):207–223.

Newman, D., Lau, J., Grieser, K., and Baldwin, T. (2010). Automatic evaluation of topic coherence. In *Proceedings of Human Language Technologies: The 11th Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL HLT 2010)*, pages 100–108, Los Angeles, USA.

Osborne, M., Petrović, S., McCreadie, R., Macdonald, C., and Ounis, I. (2012). Bieber no more: First story detection using twitter and wikipedia. In *SIGIR 2012 Workshop on Time-aware Information Access*, Oregon, USA.

Petrović, S., Osborne, M., and Lavrenko, V. (2010). Streaming first story detection with application to twitter. In *Proceedings of Human Language Technologies: The 11th Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL HLT 2010)*, pages 181–189, Los Angeles, USA.

Sakaki, T., Okazaki, M., and Matsuo, Y. (2010). Earthquake shakes Twitter users: Real-time event detection by social sensors. In *Proceedings of the 19th international conference on World wide web (WWW-2010)*, pages 851–860, New York, NY, USA.

Signorini, A., Segre, A., and Polgreen, P. (2011). The use of twitter to track levels of disease activity and public concern in the u.s. during the influenza a h1n1 pandemic. *PLoS ONE*, 6:e19467.

Weng, J. and Lee, B. (2011). Event detection in Twitter. In *Proceedings of the Fifth International Conference on Weblogs and Social Media (ICWSM-2011)*, Barcelona, Catalonia, Spain.

Zanzotto, F., Pennacchiotti, M., and Tsioutsiouliklis, K. (2011). Linguistic redundancy in twitter. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing (EMNLP 2011)*, pages 659–669, Edinburgh, United Kingdom.

Zhao, X., Shu, B., Jiang, J., Song, Y., Yan, H., and Li, X. (2012). Identifying event-related bursts via social media activities. In *Proceedings of the 2012 Conference on Empirical Methods in Natural Language Processing (EMNLP 2012)*, pages 1466–1477, Jeju Island, Korea.

Learning Compositional Semantics for Open Domain Semantic Parsing

Phong LE and Willem ZUIDEMA
Institute for Logic, Language and Computation
University of Amsterdam, the Netherlands
{p.le,zuidema}@uva.nl

Abstract
This paper introduces a new approach to learning compositional semantics for open domain semantic parsing. Our approach is called Dependency-based Semantic Composition using Graphs (DeSCoG) and deviates from existing approaches in several ways. First, we remove the need of the lambda calculus by using a graph-based variant of Discourse Representation Structures to represent semantic building blocks and defining new combinatory operations for our graph structures. Second, we propose a probability model to approximate probability distributions over possible semantic compositions. And third, we use a variant of alignment algorithms from machine translation to learn a lexicon. On the Groningen Meaning Bank (a recently released, large-scale, domain-general, semantically annotated corpus; Basile et al. (2012)), where we preprocess sentences with an existing dependency parser, we achieve results significantly better than the baseline. On Geoquery we obtain performance comparable to semantic parsers that were developed specifically for that domain.

Title and Abstract in Vietnamese

Học Tổng hợp Ngữ nghĩa cho Phân tích Ngữ nghĩa Miền Mở

Bài báo này giới thiệu một phương pháp mới cho học tổng hợp ngữ nghĩa trong phân tích ngữ nghĩa miền mở. Phương pháp của chúng tôi được gọi là Dependency-based Semantic Composition using Graphs (DeSCoG) và khác biệt với các phương pháp có sẵn trên nhiều phương diện. Trước tiên, chúng tôi loại bỏ sự cần thiết của phép tính lambda bằng cách dùng một biến thể dựa trên đồ thị cho Discourse Representation Structure để miêu tả các khối ngữ nghĩa và định nghĩa các phép kết hợp cho những đồ thị này. Thứ hai, chúng tôi đề xuất một mô hình xác suất để xấp xỉ những phân bố xác suất trên các ngữ nghĩa tổng hợp có thể có. Và thứ ba, chúng tôi dùng một biến thể của các thuật toán gióng hàng từ dịch máy để học tập từ vựng. Thực nghiệm trên Groningen Meaning Bank (một ngữ liệu vừa được công bố, lớn, tổng quát, và đã được gán nhãn ngữ nghĩa; Basile et al. (2012)), với các câu được tiền xử lý bằng một bộ phân tích phụ thuộc có sẵn, chúng tôi đạt được kết quả tốt hơn rất nhiều so với phương pháp cơ sở. Đối với Geoquery, chúng tôi có được kết quả tương đương với các bộ phân tích ngữ nghĩa được phát triển cho chính lĩnh vực đó.

KEYWORDS: semantics, parsing, dependency structure, graph.

KEYWORDS IN VIETNAMESE: ngữ nghĩa, phân tích, cấu trúc phụ thuộc, đồ thị .

Proceedings of COLING 2012: Technical Papers, pages 1535–1552,
COLING 2012, Mumbai, December 2012.

1 Introduction

Semantic parsing is the task of translating natural language sentences to formal meaning representations. The dominant approach for solving this task has been, since Montague (1970), to handcraft a semantic lexicon, using a logic to represent meanings and the lambda calculus to regulate meaning combination. Semantic parsing has until recently thus remained relatively immune to the wave of corpus-based, probabilistic approaches that have revolutionized neighboring fields, such as syntactic parsing (Petrov, 2009; Clark and Curran, 2007; Sangati and Zuidema, 2011), speech recognition (Rabiner, 1989) and machine translation (Koehn et al., 2007; Mylonakis and Sima'an, 2011).

However, an increasingly popular line of work has emerged in the last few years on using probabilistic and corpus-based methods for semantics as well. Much of this work makes use of the semantically annotated Geoquery corpus, that provides representations of queries to a geographic information system in both a natural language and a meaning representation (MR) language. For instance, the query *Which states border Arizona?* is manually translated into `answer(A,(state(A),const(B,stateid(arizona)),next_to(A,B)))`. Although the corpus is small and the domain restricted, some interesting results have been obtained in trying to learn the mapping from natural to formal language. For instance, Zettlemoyer and Collins (2005); Kwiatkowski et al. (2010, 2011) approached the problem with structural learning, Wong (2007) with machine translation, Kate and Mooney (2006) with a kernel method, and so on. Their approaches are supervised in the sense that a set of (sentence, MR) pairs is needed for training. Other work has also addressed learning with less supervision: Clarke et al. (2010); Liang et al. (2011) use the "world's response" (the answer to the query), while Goldwasser et al. (2011) even need no supervision at all.

These studies (and related ones using the CLang domain (Chen et al., 2003), where the input comes from natural language instruction) have started filling a glaring gap in statistical natural language processing. However, surveying this literature, we find some major difficulties in applying these techniques to real NLP applications in less restricted domains. First, many of the used MR languages lack expressiveness. For example, FUNQL, used in the Geoquery domain, and CLang are not designed to express tense (i.e., they make no distinction between *is*, *will be* and *has been*) or to handle possibility (e.g. *can*) nor necessity (e.g. *must*). Moreover, scalability is a big problem because of the explosion of the number of concepts as well as the number of syntactic structures in open-domain texts. For instance, in Geoquery, learning in many of the current approaches succeeds because a small number of constructions like *What is the...*, *How many states have...* and *A borders B* dominate the corpus.

Hence, the challenge of developing corpus-based, probabilistic techniques for *open-domain* semantic parsing remains unsolved. Approaches to this challenge face two major obstacles. First, sentences in open-domain texts are more complicated than those in domain-dependent texts, i.e. they are longer and contain more linguistic phenomena as well as syntactic structures. Second, large, standardized semantic corpora are not available (Bos, 2011). Wide coverage semantic parsers therefore continue to rely on hand-written rules (Bos, 2008; Allen et al., 2008), or try to use unsupervised techniques (Poon and Domingos, 2009). Alshawi et al. (2011) is based on so-called "natural logical forms" (derived directly from dependency structures) with limited applicability.

In this paper, we present an ambitious attempt to bridge the gap between hand-built approaches for open-domain semantic parsing and learning methods for domain-dependent semantic

parsing. Crucial to the success of our approach, named DeSCoG (Dependency-based Semantic Composition using Graphs), is that we make maximum use of the information provided by the syntactic structure of a sentence, and that we abandon the lambda calculus in favor of a more flexible graph-based representation. The syntactic structure comes in because we use a state-of-the-art syntactic dependency parser to drive semantic composition. There are several reasons to do that. First, dependency structures encode predicate-argument relations which are strongly related to semantics (Covington, 2001). Second, the total complexity is reduced significantly compared with parsing syntax and semantic simultaneously (Poon and Domingos, 2009). Finally, according to Ge (2010), prior knowledge of syntax is particularly helpful when sentences are long and complex.

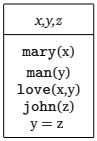
We abandon the lambda calculus, like Liang et al. (2011), but unlike many approaches (Bos, 2008; Zettlemoyer and Collins, 2005; Kwiatkowski et al., 2010; Wong, 2007; Ge and Mooney, 2009; Baral et al., 2011). Although the lambda calculus is undeniably an elegant tool for semantic composition in a bottom-up manner, it makes the learning of a lexicon difficult. This is because of the notorious λ-inverse problem, which is to find h and g such that $f = F(h, g)$ where f and F are given (Kwiatkowski et al., 2010). Zettlemoyer and Collins (2005, 2007) solved the λ-inverse problem via generating all candidates in predefined templates which correspond to categories. That solution, however, works only in domains containing sentences with simple syntactic structures such as Geoquery. To open-domain texts, because there are more than 300 frequent categories (in the WSJ corpus) (Bos, 2005), this method needs a huge effort to build a set of templates. Kwiatkowski et al. (2010) used restricted higher-order unification to carry out the λ-inverse. To adapt this method to open-domain semantic parsing, we need loosened restriction, which leads to higher complexity in learning a lexicon. Deviating from those approaches, DeSCoG represents semantics by graphs, which provide a way to flexibly break and combine components. At the same time, the formalism we use is very expressive: the graphs we obtain for complete sentences are equivalent to Discourse Representation Structure (DRS) introduced by Kamp (1981) and the formalism of choice in the hand-built Boxer system (Bos, 2008).

The paper is structured as follows. Section 2 gives a brief introduction to DRS and how to use graphs to replace the traditional box-like representations. Section 3 presents our semantic composition method, which is based on dependency structures and a probabilistic parsing model. We will describe the learning in Section 4, and finally, give experimental results on both the Groningen Meaning Bank (GMB) and Geoquery in Section 5.

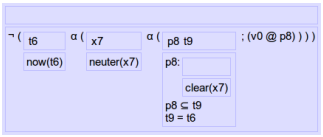
2 Meaning Representation with Semantic Graphs

2.1 Discourse Representation Structure

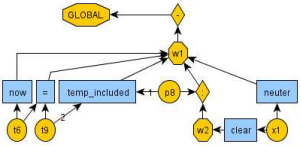
Discourse Representation Theory (DRT), which was introduced by Kamp (1981), is a theoretical framework that can handle a wide range of linguistic phenomena such as tense and anaphora, within and across sentences. It uses Discourse Representation Structures (DRS) to represent a mental representation of the hearer as the discourse unfolds (Geurts and Beaver, 2011). The traditional representation for DRS is a box-like structure which contains two components: (i) *discourse referents* (e.g. x, y, z) representing entities in the discourse, and (ii) *discourse conditions* (e.g. $\texttt{man}(x), \texttt{love}(y, x)$) representing the information about the discourse referents which is encoded in the discourse. For instance, Figure 1a shows a DRS representing the meaning of the discourse *Mary loves a man. He is John.*



(a) A sample DRS.



(b) DRS for the sentence *It is not clear .*



(c) Semantic graph for the DRS shown in Figure 1b.

Figure 1: Sample DRSs and semantic graph. Note that because our parser resolves presuppositions by local accommodation (see Section 5.1), the α-operator is treated as the $merge$-operator.

Because the pronoun *he* can only link back to the indefinite *a man*, y and z must represent the same entity. This is called *pronoun resolution*. In DRT, pronoun resolution has to satisfy the *accessibility constraint*, which says that "if y is the discourse referent introduced by a pronoun, and x is a previously introduced discourse referent, then we are only allowed to add the condition $y = x$ if x is in the universe of a DRS that is *accessible* from the DRS whose universe contains y" (Blackburn and Bos, 2000). Here, a DRS B_1 is immediately accessible from another DRS B_2 if (i) B_1 contains a condition of one of the forms: $\neg B_2$, $B_2 \vee B$, $B_2 \Rightarrow B$, for some DRS B; or (ii) $B_1 \Rightarrow B_2$ is a condition in some DRS B. B_1 is accessible from B_2 if B_1 is immediately accessible from B_2 or there is some DRS B such that B_1 is accessible from B which is immediately accessible from B_2.

Bos (2009) then developed Partial DRS which is a combination of the classic DRS, modal logic, type theory, and lambda calculus. It is used by Boxer (Bos, 2008), a state-of-the-art wide-coverage hand-built open-domain semantic parser.

2.2 Semantic Graph

Although the box-like representation captures the DRS structure very well, it does not provide us with a way to freely break and combine components. Therefore, we represent a DRS via a *semantic graph*, which is simple and directed (see Figure 1c). There are four node types corresponding to basic elements of a DRS: referent (ellipse) (e.g. x_1), predicate (rectangle) (e.g. $thing(.)$), wrapper/box (hexagon) (e.g. w_1), and operator (rhombus) (e.g. $\neg$). A global wrapper node, named `GLOBAL`, corresponds to the outermost box of a DRS. An edge is a link

- from a referent node x to a predicate node p (i.e. p has an argument x), or
- from a predicate node p to a wrapper node w (i.e. p belongs to the DRS labelled w), or
- from a wrapper node w to an operator node o (i.e. the DRS labelled w is an argument of the operator o), or
- from a referent node x to an operator node o (i.e. referent x is an argument of the operator o).

Thanks to this representation, we can break or combine components simply by removing or adding links/nodes. From this point, we will call graphs which represent meanings of words/constituents, and need to be combined with other graphs, *partial graphs*[1].

[1]A partial semantic graph should be seen as a *unsaturated formula* or a *schema*.

3 Semantic Composition

In this section we describe the components of the probabilistic, semantic parser that we will use in our model. This parser computes the best semantic graph (or DRS) for a natural language sentence, given a dependency parse of that sentence and a lexicon (which might be handcrafted, or learned, as explored in section 4).

3.1 Combinatory Operators

By abandoning the lambda calculus, we also loose its method for meaning combination. To combine partial graphs anyway, we introduce two new combinatory operators. The first (dealing with argument identity) is *binding*[2], which is to bind a referent node x with another referent node v, denoted by $x \bowtie v$. The second (dealing with the box structure in DRS) is *wrapping*, which is to link a predicate/operator node p to a wrapper node w, denoted by $p \odot w$.

Now that the lambda calculus plays no role in our new semantic representation, λ-inverse is no longer a problem. Indeed, semantic graphs provide us with maximal freedom in breaking components. Therefore, learning a semantic lexicon becomes easier. However, because it is very flexible in combining partial semantic graphs, the search space explodes. As a consequence, parsing becomes more difficult. In order to efficiently search in such a large space, we need a mechanism to bias the search. We will, in the following, give details about two key ideas: (i) partial semantic graphs are combined following the guide of given dependency structures; and (ii) some combinations are more preferable than others thanks to a probability model.

3.2 Partial Graph Combination with Dependency Structures

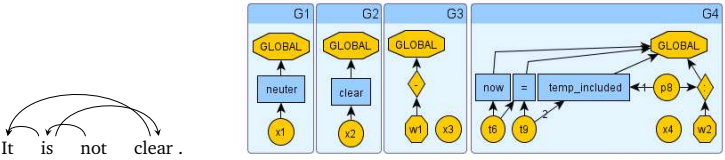
In the following, we describe how to combine partial graphs following a dependency structure, which is given by the C&C parser (Clark and Curran, 2007) (however, the model is general in the sense that it does not depend on which dependency grammar is used). Given a sentence $S = s_1 ... s_n$, a set of partial graphs $\{G^i\}$, and a dependency structure $D = \{s_i \curvearrowright s_j\}$ where $u \curvearrowright v$ means word u is a head of word v, the generative process to create a graph G consists of three steps:

1. for each word $s_i \in S$, select a partial graph G^i to represent its meaning;
2. for each $s_i \curvearrowright s_j \in D$, select a pair of referents (u, v) such that u in G^j and v in G^i to apply a binding operation to;
3. for each predicate/operator node f, select a wrapper node w to apply a wrapping operation to.

An example is given in the following. Given four partial graphs shown in Figure 2b, and a sentence with its dependency structure in Figure 2a, firstly, we assign a partial graph to each word: *It* :- $G_c^1 = G_1$, *is* :- $G_c^2 = G_4$, *not* :- $G_c^3 = G_3$, *clear* :- $G_c^4 = G_2$. Then, we bind x_1 with x_2, x_1 with x_4, x_2 with x_4, and p_8 with x_3. Finally, using the wrapper operation, we link `neuter, temp_included, =, now, :` to w_1, and `clear` to w_2. The graph in Figure 1c is what we should achieve.

It is important to note that wrapping operations should not be carried out arbitrarily. For instance, the node `clear` (Figure 1c) must not be allowed to link to the wrapper node `GLOBAL`. To stipulate that, we introduce a *wrapping constraint*: for all dependencies $s_i \curvearrowright s_j \in D$, if a

[2]It works similarly to *unification*.



(a) Dependency structure. (b) Set of partial graphs.

Figure 2: Dependency structure and set of partial graphs for the example in Section 3.2

referent node v in G^j binds with a referent node u in G^i then all the predicate/operator nodes in G^i linked from u must link to wrapper nodes which have access[3] to v. The constraint is based on the observation that in a C&C predicate-argument relation, the argument is always the one introducing referents.

Even with the dependency structure given, there are typically many choices to make among the possible partial graphs for each word, and possible binding or wrapping operations for each dependency. Computing the probabilities of all of these choices, and the most probable resulting semantic graph, is the task of the probability model.

3.3 Probability Model

The parsing task is to find the most probable semantic graph G^* such that

$$G^* = \arg\max_G Pr(G|S) = \arg\max_G \sum_D Pr(G|D,S)Pr(D|S) \quad (1)$$

where the sum is taken over all dependency derivations D for a sentence S. Similar to the model given by Ge and Mooney (2009), we assume that the dependency parse we obtain from a third party parser (here C&C) is correct. That is, we assume that the probability distribution over dependency structure derivations given a sentence S densely locates at the derivation D_S yielded by the used syntactic dependency parser, i.e. $Pr(D|S) = \delta(D = D_S)$ where $\delta(.)$ is the Kronecker delta function. As a consequence, the parsing problem becomes: given a sentence S and a dependency structure D, find the best semantic graph G^* such that $G^* = \arg\max_G Pr(G|S,D)$.

Let $G_c = \{G_c^1, ..., G_c^n\}$ be a set of assigned partial graphs, $B = \{u \bowtie v\}$ be a set of binding operations, and $W = \{f \hat{\circ}_k o\}$ be a set of in-wrapper relations, then we denote $G = (G_c, B, W)_{S,D}$, which means G is yielded by applying B, W on G_c following the dependency-structure D. We can now factorize $Pr(G|S,D)$ as follows:

$$Pr(G|S,D) = Pr(G_c, B, W|S,D) = Pr(G_c|S,D)Pr(B|G_c,S,D)Pr(W|G_c,B,S,D) \quad (2)$$

Under some independence assumptions,

$$Pr(G_c|S,D) = \prod_{i=1}^{n} Pr_l(G_c^i|s_i, POS(s_i), POS(Dep(s_i))) \quad (3)$$

$$Pr(B|G_c,S,D) = \prod_{u \bowtie v \in B} Pr_b\big(u \bowtie v|G_c(u), G_c(v), POS(s(u)), POS(s(v))\big) \quad (4)$$

[3]Because a wrapper node corresponds to a box of a DRS, the definition of *accessibility* is also applicable in this case.

where $G_c(x)$ is the partial graph containing node x and $s(x)$ is the word of which the partial graph is $G_c(x)$; $POS(.)$ is the function to get part-of-speech tags; and $Dep(s)$ is the function to get the list of the dependents of s. For the third component of Equation 2,

$$Pr(W|G_c,B,S,D) = Z \times \psi(W) \times \prod_{f\,\hat{\odot}_k o \in W} Pr_w\big(f\,\hat{\odot}_k o | G_c(f), G_c(o), POS(path(s(f), s(o)))\big) \quad (5)$$

where Z is the normalizing factor. $\psi(W)$ is the indicator function that returns 1 if W satisfies the wrapping constraint and 0 otherwise. We use the notation $f\,\hat{\odot}_k o$ for the in-wrapper relation[4] where there is a directed path from f to the k-th wrapper of the operator node o (i.e., f links to the k-th wrapper of an operator node o, or there is an operator node o' such that f links to one of o's wrappers and $o'\hat{\odot}_k o$). $f\,\hat{\odot}_0 o$ then means that there are no directed paths from f to o (e.g. in Figure 1c, `clear` $\hat{\odot}_1\langle\neg\rangle$ and `neuter` $\hat{\odot}_0\langle:\rangle$). Finally, $path(s_i, s_j)$ is the shortest path from s_i to s_j on the graph representing the dependency structure D. For instance, $path(It, not) = It \uparrow is \uparrow not$ (Fig. 2a).

Given this probability model, we can use standard techniques for finding the best semantic graph[5] G^*, given a sentence S and its dependency structure D:

$$G^* = \underset{G=(G_c,B,W)_{S,D}}{\arg\max}\ Pr(G_c|S,D)Pr(B|G_c,S,D)Pr(W|G_c,B,S,D) \quad (6)$$

Solving this optimization problem required using beam search, integer linear programming and some additional heuristics, as detailed in the appendix.

4 Learning

In this section we describe how to obtain a lexicon of partial graphs and estimate the parameters which are used in the parsing model. The learning algorithm is presented with combinations of sentences, their dependency parses and their complete semantic graphs, i.e., with a set of triples (S, D, G).

Given a sentence S and its semantic graph G, we need to split G into partial graphs corresponding to individual words. It turns out that the task is similar to the word-to-word alignment problem in machine translation. Here, English is the source language, the set of all semantic graphs is the target language, and a "word" in the target language is a connected partial graph. Hence, our problem is *word-to-graph alignment*. In Figure 3 we give a more complicated example sentence and its dependency graph (where verbs are split up in stems and tags with tense/aspect information, as explained below). In Figure 4 we give an example of an alignment with a semantic graph for this sentence .

For each triple (S, D, G), let V be the node set of G, V_F be the predicate/operator node set of G, an alignment A be a set of pairs (u, w) where $u \in V$, and $s \in S$. An alignment is feasible if all of the following constraints are satisfied

[4] In the box representation of a DRS, $f\,\hat{\odot}_k o$ means the predicate/operator f locates in the k-th box of the operator o (directly or indirectly), and $f\,\hat{\odot}_0 o$ means f does not locate in any box of o.

[5] Are the resulting DRSs valid? Although we have not provided a formal proof, we are confident they are well-formed because of the following two reasons. First, the definition of 'link' ensures that elements of a DRS (referents, predicates, boxes, etc.) are in proper relationships (e.g. it is impossible that a predicate is an argument for another predicate). Second, the wrapping constraint guarantees that the accessibility constraint is not violated. A rigorous proof of this statement is part of our future work.

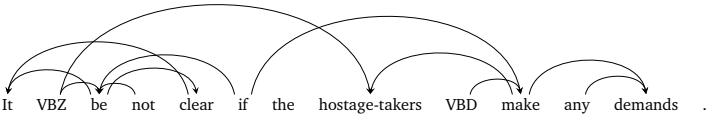


Figure 3: Dependency structure after extracting tenses out of verbs.

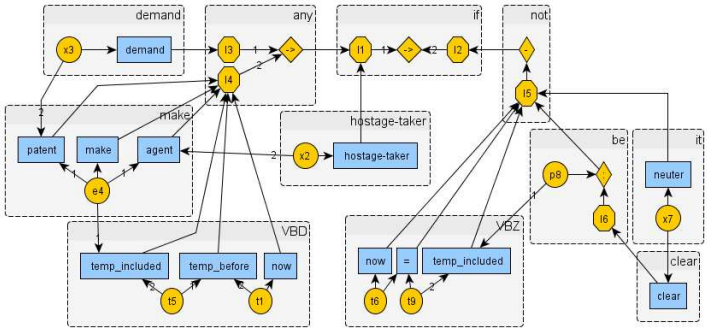


Figure 4: Semantic graph with the correct alignment for the sentence given in Figure 3.

1. for each $u \in V$, if u is a predicate or operator node, there is one and only one $s \in S$ such that $(u,s) \in A$; otherwise, for each predicate/operator node f that u links to, if $(f,s) \in A$, then $(u,s) \in A$;
2. for each $s \in S$, let V_s^A be the set of nodes associated with word s, V_s^A must be connected on G regardless of direction;
3. for each $s \curvearrowright s' \in D$, there are $u' \in V_{s'}$ and $u \in V_s$ such that u' is a predicate node. If u is a predicate node, there is a referent node x linking to both u, u'; else if u is an operator node, there is a wrapper node w linking to u such that u' links to w.

Let $\Lambda(S,D,G)$ be the set of all feasible alignments. The alignment phase is given as follows. First, a word-to-word alignment application[6] is used to estimate the probabilities $Pr(\text{node}|\text{word})$. Then, for each triple (S,D,G), the best alignment A^* is defined as[7]

$$A^* = \underset{A \in \Lambda(S,D,G)}{\arg\max}\ Pr(V_F | S = s_1 ... s_n, A) \tag{7}$$

where

$$Pr(V_F | S = s_1 ... s_n, A) \propto \prod_{s \in S} Pr(V_F \cap V_s^A | s) \propto \prod_{s \in S} \Big(Pr_n(|V_F \cap V_s^A| | s) \prod_{f \in V_F \cap V_s^A} Pr_f(f|s) \Big) \tag{8}$$

Considering solving Equation 7 as searching an A in the alignment space to minimize the objective function $g(A) = -\log\big(Pr(V_F^A | S^A, A)\big)$, we use the **A**-star algorithm with the future cost

[6] We used GIZA++ (Och and Ney, 2003) with the default parameters.

[7] Note that we use only V_F, the set of predicate and operator nodes, because the other types of nodes are determined by them.

Level	$Pr_l(G\|...)$
1	$Pr_l(G\|s, POS(s), POS(Dep(s)))$
2	$Pr_l(G\|s, POS(s))$

Level	$Pr_b(u \bowtie v\|...)$
1	$Pr_b(u \bowtie v\|G(u), G(v), POS(s(u)), POS(s(v)))$
2	$Pr_b(u \bowtie v\|G(u), G(v))$
3	$U_{u,v}(u \bowtie v\|G(u), G(v))$

Level	$Pr_w(f \hat{\odot}_k o\|...)$
1	$Pr_w(f \hat{\odot}_k o\|G(f), G(o), POS(path(s(o), s(v))))$
2	$Pr_w(f \hat{\odot}_k o\|G(f), G(o), POS(s(u)), POS(s(v)))$
3	$U_k(f \hat{\odot}_k o\|G(f), G(o), POS(s(u)), POS(s(v)))$

Table 1: Multilevel back-off for the parameter estimation. $U_{u,v}(u \bowtie v|G(u), G(v))$ and $U_k(f \hat{\odot}_k o|G(f), G(o), POS(s(u)), POS(s(v)))$ are respectively the uniform distributions over u, v and over k.

function $h(A) = \min_{A'}\{-\log\big(Pr(V_F \setminus V_F^A | S \setminus S^A, A')\big)\}$, where $V_F^A = \{u|\exists s, (u,s) \in A\} \cap V_F$ and $S^A = \{s|\exists u, (u,s) \in A\}$. It is worth noting that, because the future cost is smaller than the real cost, the optimal solution (if it exists) is guaranteed to be found.

When a lexicon of partial graphs is obtained, the parameters which are used in the parsing model are estimated via relative frequencies. To avoid the problem of sparse data, two techniques are used (see Jurafsky and Martin, 2009, Section 4.5 and Section 4.7): (i) Good-Turing technique, and (ii) multilevel back-off which is shown in Table 1.

Implementation The word-to-graph alignment method has a drawback: large partial graphs are not preferred because the probabilities of some nodes in a large partial graph given the best matched word tend to be small. Unfortunately, verbs with tenses have complex DRSs, which lead to large partial graphs (e.g. G_4 in Figure 2b represents the meaning of the verb *is*). To avoid this phenomenon, tense/aspect information is extracted from verbs (e.g. Figure 3).

We apply some heuristics to make the word-to-graph alignment more efficient. Operator nodes are only allowed to align to functional words (e.g. *be, will, could*), and predicate nodes related to tenses (e.g. `temp_included, temp_before`) are only allowed to align to tense words (e.g. *VBZ, VBN*). Because the **A**-star algorithm will search the whole solution space in the worst case, the alignment process spends a lot of time on difficult sentences. Therefore, sentences with lengths larger than MAX_LENGTH are ignored and the time for processing a sentence is not allowed to exceed MAX_TIME[8].

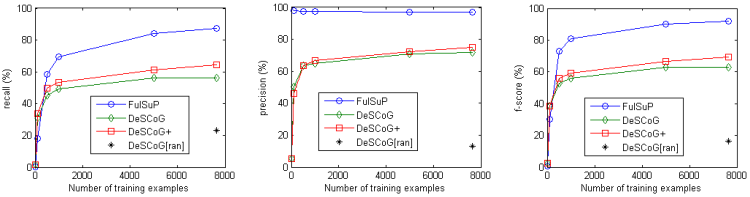
5 Experimental Results

In this section, we will show experimental results on the two corpora: Groningen Meaning Bank (GMB), an open-domain corpus, and Geoquery, a popular domain-dependent corpus.

5.1 Groningen Meaning Bank

The Groningen Meaning Bank (GMB) corpus (Basile et al., 2012) has been developed at the University of Groningen for the tasks related to deep semantic representation. The current version 1.1 contains 2000 documents with 9418 sentences from many public sources: Voice of America, fables, CIA World Factbook, and MASC Full. Each sentence is annotated with

[8] In our experiments, we set $MAX\_LENGTH = 25, MAX\_TIME = 5sec$.



(a) Recall (b) Precision (c) F-score

Figure 5: Recall, precision, and f-score curves comparing the four parsers on GMB.

a CCG parse tree with a Partial DRS at each tree node representing the meaning of the corresponding constituent. Although GMB covers many linguistic phenomena, such as anaphora and presupposition, we leave them aside. For instance, DeSCoG does not resolve anaphora whereas it resolves presuppositions by local accommodation.

Evaluation Method Our evaluation method is similar to the one introduced by Allen et al. (2008), which is based on partial credit assignment. Let G and T be semantic graphs for a gold-standard DRS and a test DRS respectively. Given an one-to-one alignment between G's nodes and T's nodes such that it results their maximum common subgraph, then

- a predicate node p in T is counted if it and all referent nodes linking to it are aligned,
- an operator node o in T is counted if it and all wrapper and referent nodes linking to it are aligned,
- count one for a predicate/operator node f in T if the wrapper node linked from f is aligned and the operator node that it links to is counted and f itself is also counted.

Intuitively, the last counting assigns credit to a component in the test DRS if its position is correct w.r.t the gold-standard DRS. Let $\Omega(G,T)$ be the number of counts, then recall $= \frac{\Omega(G,T)}{\Omega(G,G)}$, precision $= \frac{\Omega(G,T)}{\Omega(T,T)}$, and f-score $= \frac{2\Omega(G,T)}{\Omega(G,G)+\Omega(T,T)}$.

Settings We split GMB into two parts: 7642 examples in the sections from 0 to 79 for training (GMB.0-79), and the rest, 1776 examples, for testing (GMB.80-99). Because DeSCoG, to our knowledge, is the first working on this corpus, we created the following three parsers for the comparison purpose. **FulSuP** (Fully Supervised Parser) is a probabilistic semantic parser using the supervised semantic lexicon directly extracted from the GMB and some manually created semantic combinatory rules. This parser is actually a probabilistic interpretation of Boxer, the core builder of GMB. **DeSCoG+** is DeSCoG with the help from an "oracle" for the alignment process: the information about which predicate/operator nodes' labels are aligned with which words is known beforehand thanks to the semantic lexicon given by GMB. **DeSCoG[ran]** (baseline) is DeSCoG with random parameters. It is important to point out that the four parsers use different forms of supervision. FulSuP, as its name recalls, uses most supervision form; DeSCoG+ needs a set of (sentence, MRs) pairs and an oracle for training. DeSCoG and DeSCoG[ran] learn from only (S, D, G) triples.

Results Figure 5 shows the performance of DeSCoG compared to the other three parsers. The results clearly reflect the advantage of using the given semantic lexicon for training in FulSuP and DeSCoG+. The fact that FulSuP achieves the best performance (f-score 92.1% with the full

training dataset) is not surprising at all because its model is a probabilistic interpretation of the Boxer's model, which is the core builder of GMB. At the other end, DeSCoG[ran] performs remarkably poorly with an f-score of 16.3%. The result discloses the effectiveness of the parameter estimation, which helps DeSCoG gain the f-score 63.1%, which is significantly better than DeSCoG[ran].

DeSCoG+ performs better than DeSCoG. An interesting point here is that the distance between their performances is proportional to the training data size. This suggests that the alignment process of DeSCoG insufficiently exploited new structures in the added data. This suspicion is supported by the recall curve of DeSCoG: the recall is nearly unchanged after the training data size reaches 5000. DeSCoG+, on the other hand, overcomes this obstacle thanks to the help of the oracle.

Analysing the Alignment Phase

As discussed above, the quality of the alignment process strongly affects the final performance of DeSCoG. Therefore, in this section, we will look into it in some more details. First of all, because sentences longer than $MAX\_LENGTH = 25$ are ignored and the time for processing one sentence is not allowed to exceed $MAX\_TIME = 5sec$, it is clear that not all the sentences and their semantic graphs in the training dataset are successfully aligned. We found that, for the GMB.0-79, the alignment phase succeeded 5725 times, which is 74.9%.

To see when the alignment phase is wrong, let's consider an alignment result. Figure 3, 4 show the dependency structure and the semantic graph with the correct alignment for the sentence *It is not clear if the hostage-takers made any demands*. The alignment algorithm correctly aligned all of the words except *any* and *if* (Figure 6). Because $Pr(\rightarrow|any) = 0.69 > Pr(\rightarrow|if) = 0.48$, both the operator node →'s were aligned with the word *any*, and, consequently, no nodes were with the word *if*.

Theoretically, incorrect alignments could be tolerated thanks to the probability model presented in Section 3.3. For instance, if *any* and *if* do not appear together so frequently, the above fault will not have a strong effect. That is why DeSCoG+ does not perform significantly better than DeSCoG. However, the fact that the recall of DeSCoG is nearly unchanged when the training data size is over 5000 suggests that there are some structures that the alignment phase missed even when more training data were used.

5.2 Geoquery

Geoquery is a corpus which is used in the Geoquery domain, a natural language interface to a U.S geography database (Zelle and Mooney, 1996). The database contains 800 facts (e.g. names, areas, and populations), which are represented as Prolog assertions. The corpus is a collection of 880 English queries and their manually annotated MRs in a first-order Prolog language augmented with meta-predicates and Functional Query Language (FUNQL).

Although our purpose is learning open-domain semantic parsing, we found two reasons to test DeSCoG on Geoquery. First, because there are many existing semantic parsers working on this corpus, the comparison is more reliable than the previous. Second, because of many differences between Geoquery and GMB, we want to investigate the flexibility of DeSCoG. In our experiments, 10-fold cross validation was used. A test MR is correct if it and the gold-standard MR receive the same answer. Let n_c, n_D, n_P be respectively the number of correct MRs, the test

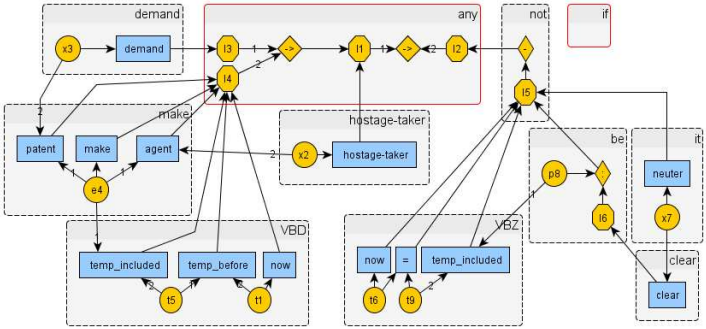
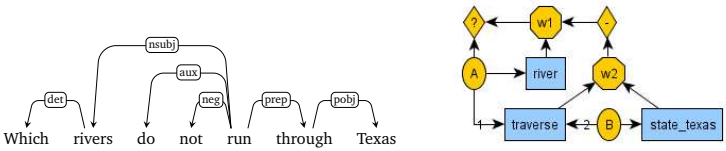


Figure 6: The alignment phase correctly aligned all the words except *any* and *if*. The correct one is given in Figure 4.



(a) Dependency structure. (b) Semantic graph.

Figure 7: Dependency structure and semantic graph for the query *which rivers do not run through Texas ?*, which has the MR
`answer(A,(river(A),not((traverse(A,B),const(B,stateid(Texas))))))`.

data size, and the number of completed MRs, then $recall = n_c/n_D$, $precision = n_c/n_P$, and $fscore = 2n_c/(n_D + n_P)$.

We made the following changes in DeSCoG. Firstly, semantic graphs are now for representing Prolog-based first-order forms. This is easily done by using operator nodes to represent metapredicates such as `answer` and `largest` (see Figure 7). Secondly, the Stanford parser[9] (De Marneffe et al., 2006) was used to syntactically parse English sentences, because it better deals with questions. Finally, the wrapping constraint was removed.

DeSCoG was compared with the following alternatives: SCISSOR (Ge and Mooney, 2005), an integrated syntactic-semantic parser; KRISP (Kate and Mooney, 2006), a SVM-based parser using string kernels; WASP (Wong and Mooney, 2006) and λ-WASP (Wong, 2007), two parsers based on synchronous grammars; Z&C05[10] (Zettlemoyer and Collins, 2005), a parser using structural learning with CCG grammars; and SYN0 (Ge and Mooney, 2009), a parser using an existing syntactic parser. Among those systems, SYN0 is the closest to DeSCoG: both of them use existing syntactic parsers and a Prolog-based first-order language as their MR languages. Table 2 shows the experimental results.

[9] http://nlp.stanford.edu/software/corenlp.shtml

[10] The results were reported on 600 training examples and 280 testing examples.

	Recall	Precision	Fscore
DeSCoG	74.89	87.40	80.66
SYN0	78.98	81.76	80.35
λWASP	86.59	91.95	89.19
Z&C05	79.29	96.25	86.95
SCISSOR	72.3	91.5	80.77
WASP	74.8	87.2	80.5
KRISP	71.7	93.3	81.1

Table 2: The results for DeSCoG and the alternatives on the Geoquery corpus.

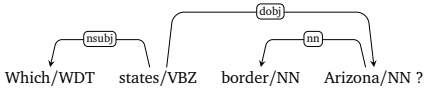


Figure 8: A sentence with its incorrect syntactic parse.

The results show that DeSCoG performs equivalently to SYN0, SCISSOR, WASP, and KRISP, but worse than λWASP and Z&C05. According to Wong (2007), SCISSOR, WASP, and KRISP perform worse than λWASP and Z&C05 because they use a different MR language, FUNQL. However, because FUNQL is variable-free, it is clear that those parsers cannot be widely used (e.g. they are not able to work on GMB). DeSCoG, on the other hand, is flexible. Although it was mainly designed for GMB (i.e. using a DRS language, the C&C parser), it achieved equal performance with many other parsers.

As mentioned above, DeSCoG and SYN0 use existing syntactic parsers and the same MR language with λWASP. So why do they perform worse than λWASP and Z&C05? Firstly, syntactic parsers are not really helpful when sentences are short. This is because sentence structures could be easily learned just from (sentence, MR) pairs in this case (Ge, 2010). Secondly, incorrect syntactic parses could lead to incorrect (or incomplete) MRs. For example, with the incorrect syntactic parse shown in Figure 8, it is very difficult to build the correct MR `answer(A,(state(B),next_to(A,B),const(B,stateid(arizona))` because one of the two words *state* and *Arizona* has to receive the meaning `next_to(A,B)`, which is very unlikely. On the other hand, parsers which learn parsing syntax and semantics simultaneously could easily overcome this problem when they recognize that the word *border* should be a head of two dependents. This is thanks to the high frequency of the appearance of the structure *A border B* in the Geoquery corpus, and that the word *border* likely matches the meaning *next_to(A,B)*.

Conclusion

This paper introduced a new learning approach, DeSCoG, mainly for open-domain semantic parsing. Our approach, on the one hand, is different from others which use lambda calculus. In those approaches, learning semantic lexicon requires huge effort because they face the λ-inverse problem. In our approach, this problem is avoided thanks to the usage of semantic graphs, which provides maximal freedom for breaking and combining MRs. In order to bias the search in a very large parse space, we used the prior knowledge of syntax encoded in dependency structures and a probability model. The former provides a skeleton for semantic composition whereas the latter tells the parser which combinations are more preferable.

On the other hand, DeSCoG has common points with some approaches. First, the idea of

flexible semantic composition is also proposed by Clarke et al. (2010); Liang et al. (2011) where lambda forms and semantic composition rules are removed. Second, the use of existing syntactic parsers in DeSCoG is inspired by Ge and Mooney (2009). Finally, UPS (Poon and Domingos, 2009), Alshawi et al. (2011), and DeSCoG use dependency structures as the prior knowledge of syntax for semantic composition.

We tested DeSCoG on the two corpora, GMB for open-domain semantic parsing and Geoquery for domain-dependent semantic parsing. DeSCoG achieved the f-score 63.1% on GMB. The fact that this f-score is significantly higher than the f-score of the random parser (16.3%) reflects that our proposed parsing model works for open-domain semantic parsing. Moreover, DeSCoG performed on Geoquery equally to many existing parsers: SYN0, SCISSOR, WASP, and KRISP.

Appendix: Search heuristics

We will describe a method for finding the best semantic graph G^* in Equation 6 given a sentence S and its dependency structure D. The problem is indeed searching within an enormous semantic graph space. The search is biased by the dependency structure D and the probability model given above. Here, we use the beam search strategy to cut off branches which could lead to solutions with low probabilities. To reduce the total complexity, we divide the search into two stages. First, the search, with the goal to maximize $Pr(G_c|S,D)Pr(B|G_c,S,D)$, will output a list of N-best (G_c,B)'s. Then, the search will look for the best W for each of those N-best (G_c,B)'s.

Stage 1 The goal of this stage is to find N-best (G_c,B)'s. The search processes one word at a time with the order: (i) from left to right, and (ii) all the dependents of the word were already processed. And, each word is processed only one time (i.e. no backtracking required). For example, given the sentence in Figure 2a, *It* is processed first, then *clear* and *is*; *not* is processed last. When processing a word (e.g. *is*), the search will considers all the candidate partial graphs for the meaning of this word, then use the binding operator to combine them with the available parses yielded after processing the prior words (*it, clear*). When there are no words left, it will remove all but N-best candidates with the highest $Pr(G_c|S,D)Pr(B|G_c,S,D)$.

Stage 2 After finding the set of N-best (G_c,B)'s, the search will look for the most probable W^* for each (G_c,B), i.e. solve $W^* = \arg\max_W Pr_w(W|G_c,B,S,D)$ where $Pr_w(W|G_c,B,S,D)$ is given by Equation 5. Because the normalizing factor Z is the same for given G_c,B, the task is to maximize the third component under the wrapping constraint, which is encoded in $\psi(W)$. It is worth noting that this is indeed a linear optimization problem where the logarithm of the third component, i.e. $\sum_{f\,\hat{\circledcirc}_k o \in W} \log Pr_w\big(f\,\hat{\circledcirc}_k o|G_c(f),G_c(o),POS(path(s(f),s(o)))\big)$, is the objective function. Therefore, we solve it by making use of Integer Linear Programming[11]. In order to calculate Z for each G_c,B, we need to take the sum over all feasible W's, which is intractable. Therefore, Z is approximated on M-best W's $Z \approxeq \big(\sum_{i=1}^{M} e^{F(W_i)}\big)^{-1}$. Together, this give us a set of triples (G_c,B,W) with their probabilities.

Acknowledgments

We thank Remko Scha for valuable discussions and three anonymous reviewers for helpful comments.

[11] Gurobi solver is used. http://www.gurobi.com/

References

Allen, J., Swift, M., and de Beaumont, W. (2008). Deep semantic analysis of text. In *Symposium on Semantics in Systems for Text Processing (STEP)*, volume 2008.

Alshawi, H., Chang, P. C., and Ringgaard, M. (2011). Deterministic statistical mapping of sentences to underspecified semantics. *Proceedings of the Ninth IWCS*, pages 15–24.

Baral, C., Dzifcak, J., Gonzalez, M. A., and Zhou, J. (2011). Using inverse lambda and generalization to translate english to formal languages. *Arxiv preprint arXiv:1108.3843*.

Basile, V., Bos, J., Evang, K., and Venhuizen, N. (2012). Developing a large semantically annotated corpus. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC'12)*.

Blackburn, P. and Bos, J. (2000). Working with discourse representation theory: An advanced course in computational semantics. *Draft available at www. comsem. org*.

Bos, J. (2005). Towards wide-coverage semantic interpretation. In *Proceedings of Sixth International Workshop on Computational Semantics IWCS*, volume 6, pages 42–53.

Bos, J. (2008). Wide-coverage semantic analysis with boxer. In *Semantics in Text Processing. STEP 2008 Conference Proceedings*, volume 1, pages 277–286.

Bos, J. (2009). Towards a large-scale formal semantic lexicon for text processing. In *Proceedings of the Biennal GSCL Conference From Form to Meaning: Processing Texts Automatically*, pages 3–14.

Bos, J. (2011). A survey of computational semantics: Representation, inference and knowledge in wide-coverage text understanding. *Language and Linguistics Compass*, 5(6):336–366.

Chen, M., Dorer, K., Foroughi, E., Heintz, F., Huang, Z., Kapetanakis, S., Kostiadis, K., Kummeneje, J., Murray, J., Noda, I., Obst, O., Riley, P., Steffens, T., Wang, Y., and Yin, X. (2003). *Users Manual: RoboCup Soccer Server — for Soccer Server Version 7.07 and Later*. The RoboCup Federation.

Clark, S. and Curran, J. (2007). Wide-coverage efficient statistical parsing with CCG and log-linear models. *Computational Linguistics*, 33(4):493–552.

Clarke, J., Goldwasser, D., Chang, M., and Roth, D. (2010). Driving semantic parsing from the world's response. In *Proceedings of the Fourteenth Conference on Computational Natural Language Learning*, pages 18–27.

Covington, M. A. (2001). A fundamental algorithm for dependency parsing. In *Proceedings of the 39th annual ACM southeast conference*, pages 95–102.

De Marneffe, M., MacCartney, B., and Manning, C. (2006). Generating typed dependency parses from phrase structure parses. In *Proceedings of LREC*, volume 6, pages 449–454.

Ge, R. (2010). Learning for semantic parsing using statistical syntactic parsing techniques. Technical report, DTIC Document.

Ge, R. and Mooney, R. (2009). Learning a compositional semantic parser using an existing syntactic parser. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 2-Volume 2*, pages 611–619.

Ge, R. and Mooney, R. J. (2005). A statistical semantic parser that integrates syntax and semantics. In *Proceedings of the Ninth Conference on Computational Natural Language Learning*, pages 9–16.

Geurts, B. and Beaver, D. I. (2011). Discourse representation theory. In Zalta, E. N., editor, *The Stanford Encyclopedia of Philosophy*. Fall 2011 edition.

Goldwasser, D., Reichart, R., Clarke, J., and Roth, D. (2011). Confidence driven unsupervised semantic parsing. In *Proc. of the Meeting of Association for Computational Linguistics (ACL), Portland, OR, USA*.

Jurafsky, D. and Martin, J. H. (2009). *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition*. Pearson Prentice Hall.

Kamp, H. (1981). A theory of truth and semantic representation. *Formal Semantics*, pages 189–222.

Kate, R. and Mooney, R. (2006). Using string-kernels for learning semantic parsers. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, pages 913–920.

Koehn, P., Hoang, H., Birch, A., Callison-Burch, C., Federico, M., Bertoldi, N., Cowan, B., Shen, W., Moran, C., Zens, R., Dyer, C., Bojar, O., Constantin, A., and Herbst, E. (2007). Moses: open source toolkit for statistical machine translation. In *Proceedings of the 45th Annual Meeting of the ACL on Interactive Poster and Demonstration Sessions*, ACL '07, pages 177–180, Stroudsburg, PA, USA. Association for Computational Linguistics.

Kwiatkowski, T., Zettlemoyer, L., Goldwater, S., and Steedman, M. (2010). Inducing probabilistic CCG grammars from logical form with higher-order unification. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 1223–1233.

Kwiatkowski, T., Zettlemoyer, L., Goldwater, S., and Steedman, M. (2011). Lexical generalization in CCG grammar induction for semantic parsing. In *Empirical Methods in Natural Language Processing (EMNLP)*.

Liang, P., Jordan, M., and Klein, D. (2011). Learning dependency-based compositional semantics. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, pages 590–599. Association for Computational Linguistics.

Montague, R. (1970). Universal grammar. *Theoria*, 36(3):373–398.

Mylonakis, M. and Sima'an, K. (2011). Learning hierarchical translation structure with linguistic annotations. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 642–652, Portland, Oregon, USA. Association for Computational Linguistics.

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

Petrov, S. (2009). *Coarse-to-Fine Natural Language Processing*. PhD thesis, University of California at Berkeley.

Poon, H. and Domingos, P. (2009). Unsupervised semantic parsing. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 1-Volume 1*, pages 1–10.

Rabiner, L. (1989). A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286.

Sangati, F. and Zuidema, W. (2011). Accurate parsing with compact tree-substitution grammars: Double-dop. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 84–95, Edinburgh, Scotland, UK. Association for Computational Linguistics.

Wong, Y. W. (2007). *Learning for semantic parsing and natural language generation using statistical machine translation techniques*. ProQuest.

Wong, Y. W. and Mooney, R. J. (2006). Learning for semantic parsing with statistical machine translation. In *Proceedings of the main conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics*, pages 439–446.

Zelle, J. M. and Mooney, R. J. (1996). Learning to parse database queries using inductive logic programming. In *Proceedings of the National Conference on Artificial Intelligence*, pages 1050–1055.

Zettlemoyer, L. and Collins, M. (2005). Learning to map sentences to logical form: Structured classification with probabilistic categorial grammars. In *Proc. of UAI*, volume 5.

Zettlemoyer, L. S. and Collins, M. (2007). Online learning of relaxed CCG grammars for parsing to logical form. In *In Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL-2007)*.

Evaluating different methods for automatically collecting large general corpora for Basque from the web

Igor LETURIA
ELHUYAR FUNDAZIOA, Usurbil, Basque Country (Spain)
i.leturia@elhuyar.com

ABSTRACT

In the last few years, much work has been done to build Basque corpora. But we still lack a large general corpus of a size comparable with those existing in other major languages, and much more so if we take into account the corpora lately built automatically from the web, which nowadays account for billions of word-sized corpora for English, German, Spanish, etc. As Basque is an under-resourced language, it is thus logical that we should also turn to this cheap and fast method of collecting corpora.

In this paper we present the research we have done to build a large general corpus of Basque from the web. We have tried and evaluated which of the two methods mentioned in the literature, that is, by crawling or by using search engines, best suits Basque, in terms of parameters such as speed, cost, size or quality. Our conclusion is that crawling is the one that has the potential for building the largest corpora for Basque. Using this method we have built a good quality corpus of more than 100 million words, and we expect to build a much larger one in the near future.

TITLE AND ABSTRACT IN BASQUE

Webetik euskarazko corpus orokor handiak automatikoki biltzeko metodoen ebaluazioa

Azken urteotan lan handia egin da euskarazko corpusgintzan. Baina oraindik ez dago tamainari dagokionez beste hizkuntza handiagoetakoekin konpara daitekeen corpus orokor handirik; are gehiago kontuan hartzen baditugu azkenaldian automatikoki webetik bildu diren corpusak: milaka milioi hitzetakoak daude ingelesa, alemana, gaztelania eta abarrerako. Euskara baliabide urriko hizkuntza izanik, logikoa da corpusak biltzeko metodo merke eta azkar honetara jotzea.

Artikulu honetan webetik euskarazko corpus orokor handi bat biltzeko egin dugun ikerketa aurkezten dugu. Literatura zientifikoan aipatzen diren bi metodoetatik, hau da, crawling bidez edo bilatzaileak erabiliz, euskararentzat hobekien zeinek funtzionatzen duen aztertu eta ebaluatu dugu, abiadura, kostua, tamaina edo kalitatea moduko parametroei erreparatuz. Gure ondorioa da crawling bidezkoa dela euskarazko corpus handiena eraikitzeko aukera ematen duena. Metodo hori erabiliz 100 milioi hitz baino gehiagoko corpus kalitatezko bat osatu dugu, eta etorkizun hurbilean askoz handiago bat eraikitzea espero dugu.

KEYWORDS: Basque, Corpora, Web as Corpus.

KEYWORDS IN BASQUE: euskara, corpusak, weba corpus gisa.

Proceedings of COLING 2012: Technical Papers, pages 1553–1570,
COLING 2012, Mumbai, December 2012.

1 Summary in Basque

1.1 Motibazioa eta helburuak

Corpusak oso baliabide garrantzitsua dira mundu modernoan biziraun eta komunikabideetan, hezkuntzan eta abar normaltasunez erabilia izan nahi duen hizkuntza batentzat, besteren artean (baina ez bakarrik) beharrezkoak direlako hizkuntza-teknologiak garatzeko. Euskararen kasuan are gehiago, oraindik ere estandarizazio prozesuan baitago eta estatusaren normalizazioan asko baitu egiteko. Baina euskara urria da corpusetan, modu klasikoan eraikitzea garestia delako eta ez dagoelako behar adina baliabide (ekonomikoak zein giza baliabideak) berauek eraikitzeko. Besteak beste, ez du corpus orokor handirik beste hizkuntzen pareko tamainakorik.

Hala ere, azkenaldian weba gero eta gehiago erabili da hizkuntz ikerketetarako edo corpusgintzarako; izan ere, merkeagoa da, corpus handiagoak lor daitezke eta beti eguneratuta dago. Horregatik logikoaz gain ia beharrezkoa da euskararentzat webera jotzea. Azken urteetan egin dira lanak euskarazko corpus mota ezberdinak automatikoki webetik lortzeko, baina oraindik ez da corpus orokor handien gaia landu. Artikulu honetan azaltzen dira helburu horrekin dauden metodo ezberdinak probatuz egin dugun ikerketa-lana eta beronen emaitzak.

1.2 Antzeko lanak

Bi modu nagusi aipatzen dira literaturan webetik corpus orokor handiak lortzeko. Bata crawling metodoa da: URL zerrenda batetik abiatuta, orri horiek jaisten dira eta bertako esteken URLak zerrendara gehitzen dira, eta horrela errekurtsiboki egiten jarraitzen dugu. Metodo honekin egin dira WaCky (Web-as-Corpus kool ynitiative) proiektuko corpusak, milaka milioi hitzetako corpusak alemana, italiera, ingelesa eta frantseserako, eta gehiago daude bidean (Baroni et al., 2009). Bestea bilatzaileak erabiltzean datza. Hitz zerrenda batetik abiatuta, horien konbinazioak bidaltzen zaizkie bilatzaileen APIei eta itzulitako emaitzen orriak jaisten dira. Metodo hau erabiltzen du Sharoff-ek (2006) 100-200 milioi hitz inguruko corpusak osatzeko hainbat hizkuntzatarako.

Bi metodoek emaitza onak lortu badituzte ere, ez dira beraien artean konparatu, beraz galdera asko geratzen dira airean. Zein da azkarragoa? Zeinek lortzen ditu kalitate handiagoko corpusak? Lor daitezke milaka milioietako corpusak bilatzaileen bidez? Gainera, euskararentzat baliteke ezberdin funtzionatzea metodoek: batetik, euskarazko bilaketa lematizatuak egiteko erabili behar diren teknikek emaitzei eragin diezaiekete; bestetik, euskarazko webaren tamaina txikiak eragin lezake crawling-a ez hain eraginkorra izatea. Beraz, biak aztertu eta ebaluatzea beharrezkoa da.

1.3 Metodologia

1.3.1 Bilatzaileen bidezko metodoa

Bilatzaileen bidezko metodoan, Sharoff-ek (2006) 500 hitz-forma maiz eta orokorren zerrenda bat erabiltzen du. Guk XX. mendeko Euskararen Corpuseko lema maizenak erabili ditugu eta gero sorkuntza morfologiko bidezko kontsultaren hedapena aplikatu (Leturia et al., 2008b). Euskarazko emaitzak soilik lortzeko, hizkuntza iragazteko hitzen teknika baliatu dugu (Leturia et al., 2008b). Sharoff-ek 500 hitz baino gehiago ere erabil daitezkeela iradokitzen du; horren eragina frogatu nahi izan dugu, 500, 1.000, 2.000, 5.000 eta 10.000 hitzeko zerrendak erabiliz

corpus ezberdinak jaitsi eta ebaluatuta. Sharoff-ek 4 hitzeko konbinazioak bidaltzen dizkie bilatzaileen APIei eta hizkuntza txikiagoei 3; guk 1, 2, 3, 4 eta 5 hitzezko konbinazioekin egin nahi izan ditugu frogak. Sharoff-ek bilatzaileek itzulitako lehen 10 orriak jaisten ditu, guk badaezpada ere 50.

1.3.2 Crawling metodoa

WaCky proiektuan, crawling egiteko hasierako URLen zerrenda bilatzaileengandik lortzen zuten, hainbat hitzen konbinazioak bidalita; guk DMOZeko "Euskara" ataleko 1.500 URLak erabili ditugu. Haien proiektuan bezala, guk ere ataza anitz paraleloan abiarazita azkartzen dugu orrien deskarga eta webgune aniztasuna lehenesteko estrategia darabilgu.

1.4 Emaitzak

1.4.1 Bilatzaileen bidez

Bilatzaileak erabiliz, corpusik handienak 2.000 edo 5.000 hitzeko hasierako zerrendatik abiatuta lortu dira, 120-130 milioi hitz ingurukoak: lehenak webgune aniztasun handiagoa lortzen du, bigarrenak PDF gehiago (normalean arazoak ematen dituzte bihurtzean eta kalitatea ez da hain ona) eta dokumentu handiagoak (testu jarrai gehiago). Hitzen konbinazioaren luzera egokia 2 dela dirudi, berak lortzen baitu corpusik handiena (130 milioi baino gehiago) eta anitzena, PDF gutxienekin.

1.4.2 Crawling bidez

Bilatzaileen bidezko metodoekin lortu diren corpusek ez dute lortutako tamainak baino askoz gehiago handitzeko gaitasunik, ez behintzat modu produktiboan: bilatzaileei egindako lehen mila galderekin 37 milioi hitz lortzen dira bataz beste, baina egindako azken mila galderekin (12.000raino iritsi gara) 2 milioitik behera lortzen dira. Crawling bidez, aldiz, 100 milioi hitzetik gorako corpusak lortu ditugu (ia PDFrik gabe eta webgune aniztasun handienarekin) eta hazkunde erritmoak hasierakoaren erdia izaten jarraitzen du, beraz handitzen jarraitzeko potentziala du.

1.4.3 Konparazio kualitatiboa

Bi corpus klasikorekin (XX. mendeko Euskararen Corpusa eta Lexikoaren Behatokia) konparatu ditugu crawling bidez eta bilatzaile bidez lortutako web corpus bana. Bilatzaileen bidezkoa besteengandik ezberdintzen duena da testu administratibo gehiago izatea (ziurrenik erakundeen aldizkari ofizialetako PDFengatik), eta crawling bidezkoa webeko berezko generoko testu gehiago izatea. Corpus klasikoen hitz baliagarrien %90etik gora badaude web corpusetan, eta azken hauek %80 inguru hitz baliagarri berri gehiago dituzte lehenek baino.

1.5 Ondorioak

Frogatu dugu weba lehengaitzat hartuta posible dela 100 milioi hitzetik gorako corpusak osatzea bai bilatzaileen bidez bai crawling bidez, eta azken hau erabiliz ziurrenik askoz handiagoak ere osa daitezkeela. Gainera kalitatezkoak dira, corpus klasikoen hitz gehienak barne hartzen dituzte eta berri asko dauzkate. Beraz, webetik corpus orokor handiak biltzeak ekarpen handia egin diezaioke euskarazko corpusgintza eta hizkuntzalaritzari eta baita hizkuntzari orokorrean ere.

2 Motivation and objectives

A language that wants to survive in the modern world and be used normally in the media, in education, etc. needs to have language resources such as dictionaries or corpora. Besides, because language technologies are ever more present in everyday life through the web and our gadgets, it is imperative for any language with perspectives for the future to develop these language technologies; and these in turn need electronic dictionaries and corpora in order to be developed. And dictionaries (lexicographical or terminological) are nowadays produced on the basis of empirical evidence or previous use at least is studied, and these are both provided by corpora (they can even be semi-automatically extracted from corpora using NLP methods). So it is clear that corpora are a very valuable resource for many aspects of the development of a language.

In the case of the Basque language, the need for corpora is even greater, since its standardization did not start until the late 1960s and is still ongoing. But less resourced languages like Basque are not exactly rich in corpora: on the one hand, building a corpus in the classical way, i.e. out of printed texts, is normally a very costly process; on the other, the number of language experts or researchers dealing with these languages is much smaller than that of the major languages. So, the only Basque corpora that are currently available to the public are as follows:

- Orotariko Euskal Hiztegiaren Testu-Corpusa: a 6 million-word non-tagged corpus of classical literary texts produced by Euskaltzaindia, the Royal Academy of the Basque Language.
- XX. mendeko Euskararen Corpusa (http://www.euskaracorpusa.net/XXmendea): a 4.6 million-word balanced corpus produced by Euskaltzaindia; it consists mainly of twentieth century literary texts.
- Ereduzko Prosa Gaur (http://www.ehu.es/euskara-orria/euskara/ereduzkoa/): a 25.1 million-word corpus compiled by the UPV/EHU-University of the Basque Country, composed of literary and press texts regarded as "reference texts" from the years 2000 through 2006.
- Zientzia eta Teknologiaren Corpusa (http://www.ztcorpusa.net): a 8.5 million-word corpus compiled by the Elhuyar Foundation and the IXA Group of the UPV/EHU-University of the Basque Country, consisting of texts on science and technology published between 1990 and 2002 (Areta et al., 2007).
- Klasikoen Gordailua (http://klasikoak.armiarma.com/corpus.htm): a non-tagged 11.9 million-word corpus compiled by the publishing house Susa, consisting of classical texts.
- Lexikoaren Behatokia (http://lexikoarenbehatokia.euskaltzaindia.net): an 18.1 million-word corpus produced by Euskaltzaindia, the Elhuyar Foundation, the IXA Group of the University of the Basque Country and UZEI, made up of 21st century media texts.

But in recent years the web has been used increasingly for linguistic research, both via tools like WebCorp (Renouf et al., 2006) or KWiCFinder (Fletcher, 2006) that query search engines directly and show concordances, or via tools that use the Internet as a source of texts for building corpora to be used the classical way, after linguistic tagging and indexation (Baroni and Kilgarriff, 2006). As Kilgarriff and Grefenstette (2003) put it, although the use of the web as a source for building linguistic corpora has its detractors (who basically object to its lack of

representativeness and reproducibility and to the quality of its texts), this approach offers undeniable advantages, mainly that the corpora that can be obtained are much larger, that the cost of the automatic building processes is much lower and that the web is constantly up to date.

So it is not only logical but also almost unavoidable that a less resourced language like Basque should proceed as other languages have and turn to the Internet, and this is what has been done in recent years. The following are the tools or resources built by using this web-as-corpus approach:

- CorpEus (http://www.corpeus.org/): a web service to query the web live as if it were a Basque corpus, by showing KWiCs of pages in Basque (Leturia et al., 2007).
- AutoCorpEx: a tool to automatically collect specialized corpora from the web (Leturia et al., 2008a)
- Co3: a tool to obtain domain comparable corpora from the web (Leturia et al., 2009).
- PaCo2: a tool to build parallel corpora from the web (San Vicente and Manterola, 2012).

Using the last three tools, various corpora (monolingual specialized, comparable and parallel) have been built. But there is still no large general corpus in Basque with a state-of-the-art size; as we have seen, the largest Basque corpus has 25 million words, whereas, taking English as an example, the BNC was finished in 1994 and runs to 100 million words (Aston and Burnard, 1998), and the web-derived corpus ukWaC contains more than 2 billion words (Ferraresi et al., 2008). So, the main objective of the research described in this paper was to build a general Basque corpus that was as large as possible (comparable to the sizes of the corpora we have mentioned, if possible). But in order to achieve this, we have had to test the different methods mentioned in the literature for collecting large general corpora from the web to see which performed best for Basque, because the features of the language might affect the results; so the results of the evaluation of the different methods are also shown.

3 Related work

There are roughly two methods that are mentioned in the literature when it comes to building large corpora out of the web. One of them is the crawling method: starting from a list of seed URLs, the pages they point to are downloaded, and the links found in them are added to the list of URLs to do likewise with them; we apply this recursively until the list is finished or we reach a predefined endpoint. As the web is a collection of interconnected pages, starting from almost any seed list sufficiently large and applying this method, the whole public web can be downloaded. This is the method used in the WaCky project (Web-as-Corpus kool ynitiative), an initiative to build gigantic web corpora for many languages (Baroni et al., 2009), with which they have already built four corpora for four languages of around or more than 2 billion words each (and more are on the way): deWaC for German (Baroni and Kilgarriff, 2006), itWaC for Italian (Baroni and Ueyama, 2006), ukWaC for English (Ferraresi et al., 2008) and frWaC for French.

The other method relies on the use of search engines. A list of seed words is used, combinations of them are sent to the APIs of search engines and the resulting pages are downloaded, until the goal size is reached, no more combinations are left or no new pages are returned. This is the method used by Sharoff (2006) to build BNC-sized corpora (around 100-200 million words) for various languages.

Both methods report success stories. They are able to obtain corpora of the desired size and the word frequencies are comparable with those in classical corpora such as BNC. However, the two methods or the corpora obtained with them have not been compared with each other, so many questions remain in the air. Which is the fastest method? Which obtains the best quality corpora? Is it possible to obtain corpora of billions of words with the search engines method?

And even if it was clear which of the methods is the best, it does not necessarily have to be so for obtaining a corpus in Basque, due to the singularities of the language. For example, no search engine offers the possibility of restricting its results to pages that are in Basque or to perform a search taking the rich morphology of Basque into account, so some hacks have to be used when querying search engines for content in that language, which might affect the results of the corpora obtained. Or, due to the smaller size of the Basque web, the crawling method might not obtain a sufficient size because it might leave out a significant part of the Basque web if the seed URLs list is not good or large enough. So, testing and evaluating both methods (and with different parameters) for collecting a large general corpus of Basque is a necessary task.

4 Methodology

4.1 Search engine method

Sharoff (2006) uses the search engine method to build 100-200 million-word corpora for various languages. He uses a seed list of 500 words, which have to meet certain requirements: they must be frequent, they have to be general (i.e., they should not indicate a specific topic) and they must not be function words (prepositions, articles, pronouns, conjunctions, etc.).

In our case, we acted likewise. We took the list of frequent words from XX. mendeko Euskararen Corpusa (see above), and we removed the non-desired ones. We took out the pronouns and conjunctions, but there was no need to remove articles or prepositions (Basque is an agglutinative language and these are appended to the words). Topic-specific words were not removed, because there were not many of them among the first 500 (the most frequent words tend to be general).

However, we take a different approach to Sharoff's regarding lemmatization. As he points out, general search engines do not perform lemmatization, so his seed words list is formed by word forms. We use a list of lemmas and apply morphological query expansion when calling the API of the search engine. Basically, this consists of obtaining the inflections of a word by morphological generation and sending the most frequent ones within an OR operator. For example, if the lemma of a word is *etxe* ("house"), the search engine is asked for "etxe OR etxea OR etxeak OR etxeari OR etxeek OR etxearen OR…". This method has proven to be effective for obtaining from search engines a lemma-based search for Basque, and is the method usually used in services or projects that need to search for content in Basque (Leturia et al., 2008b; Leturia et al., 2009).

In order to obtain from search engines only the pages in the language of the corpus to be collected, some studies consider the need to select words that are unique to the language (Kilgarriff and Grefenstette, 2003; Ghani et al., 2003), rejecting words like "restaurant" that exist in several different languages. The above-mentioned work by Sharoff uses the language filter of search engines, except for Ukrainian (which is not covered by search engines), for which the query is complemented with a couple of very frequent function words that are not used in cognate languages. We do not reject words existing in other languages but we are not in a

position, either, to use the language filters of search engines because none of the existing search services can limit the results to pages in Basque. We apply the technique of the language-filtering words, like Sharoff for Ukrainian, by appending the most frequent words in Basque to the query ("… AND eta AND da AND ez AND ere"). As proven by Leturia et al. (2008b), this is the most effective method for obtaining results in Basque alone from search engines, although it means a loss in recall.

In the aforementioned paper, Sharoff also suggests that more than 500 words can be used. It would indeed be interesting to test the effect of the length of the seed word list in the corpus collection process and the obtained corpora; so, we tried with seed words lists of 500, 1,000, 2,000, 5,000 and 10,000 words.

In that same work, 4-word combinations are sent to the APIs, in order to get pages that contain relatively long pieces of connected text and a smaller number of noisy pages, i.e. tables or lists of links. However, he states that it is possible to relax the condition for four words in a query for languages which do not have sufficient number of Internet pages, and in fact he used 3-word combinations for Romanian. Because the Basque web may be orders of magnitude smaller than that in other languages, there is justification in seeing if there is in fact improvement with a shorter combination length; and there is no reason why the effect of longer ones should not be checked as well. That is why we also tried and evaluated 1, 2, 3, 4 and 5-word combinations.

Regarding the search engine, we used Google's API, just like Sharoff. From the results returned by the API, Sharoff downloads the first 10 pages. We decided to download the first 50, for one reason: because of the smaller size of the Basque web, many searches return no results (especially in the longer seed words lists and the longer combinations); so in order to build larger corpora while making the least possible number of queries, we downloaded more results from the productive queries.

4.2 Crawling method

The crawling method needs a list of seed URLs as a starting point. The corpora collected by the WaCky initiative (Baroni et al., 2009) obtain these seed URLs by making random queries of 2-word combinations to search engines (they make about 1,000 queries for getting around 10,000 seed URLs). In our case, we took the 1,500 URLs of the *Euskara* (meaning Basque language) section of DMOZ, the Open Directory Project. Although it is not an exhaustive list of all the websites in Basque and it is not as active and updated as it used to be, all the most important sites are undoubtedly there, and by following the links present in them recursively, we believe that it is almost certain that ultimately no site would be left out (except for island sites that have no inbound links from anywhere, but neither would these be indexed by search engines). However, this is one of the points that we wanted to test in our experiments.

The crawling is done in a multi-threaded parallel way with a breadth first strategy (prioritizing website variety above website completeness), just as in the WaCky initiative.

4.3 Common filters in both methods

The pages that are downloaded, whether by using search engines or by crawling, need to go through some cleaning and filtering if we want to build a quality corpus. Here we will describe how we implemented these steps.

Language filtering. When building a corpus, one is usually looking for texts in a language. When using search engines, the language filter is done by telling the search engine to return only results in that specific language. But when using the crawling method or, in some cases, also the search engine method (if search engines do not offer filtering by the language we want), it is up to us to do the language filtering after downloading. For this task we make use of LangId, a language identifier based on character and word trigram frequencies specialized in Basque, applied at paragraph level so that we can also extract content from bilingual documents.

Length filtering. Fletcher (2004) proved that filtering web documents by their size improved the quality of the web corpora. Those that do not reach a minimum are usually error messages from web servers or tend to have little textual content once page headers, menus, etc. are removed. On the other hand, those that are too large are not good for linguistic corpora, since they are often not representative of real language and tend to be lists, catalogues, spam and such things. We do in fact apply a length filter but, unlike most projects, it is not based on the size of the downloaded file. We reject documents the length of which after conversion to plain text is under 1,000 characters or over 100,000 characters.

Spam and porn filtering. The web is full of spam, porn and other kinds of noise. When we build a corpus out of web documents it is essential to get rid of these elements, but it is not always easy. The size filter proposed by Fletcher (2004) decreases this kind of noise but does not eliminate it completely. If we use search engines, we will most probably get less spam and porn, since they already do this filtering. But it is always desirable, and in the case of crawling methods necessary, to implement the detection of spam and porn. We do not apply any specific filter for spam and porn, because there is hardly any in Basque. People with commercial intentions target larger audiences and do not bother about minority languages spoken by communities that speak some other major language. Therefore, the language filter does the job perfectly.

Boilerplate removal. Web pages are full of "*boilerplate*", which is the linguistically uninteresting material that web server software automatically creates and which is repeated throughout every page in a website: headers, navigation menus, copyright notices, ads, etc. It is advisable to remove this boilerplate for various reasons: it makes ugly KWiCs, it distorts word frequencies and it makes the work of other filters (near-duplicate filtering, for example) more difficult. For boilerplate removal, we use Kimatu (Saralegi and Leturia, 2007), a language-independent system based on heuristics and features like tag density, punctuation signs, function words, etc. that scored second (74.3%) in the Cleaneval competition (Baroni et al., 2008).

Near-duplicate detection. The detection of exact duplicates is a straightforward task easily accomplished by hashing techniques. But much content is repeated across different websites (news from agencies in media sites, CC licensed articles in many blogs...) which are not exact duplicates, and these cannot be detected by hashing methods. The method most used for this job is Broder's algorithm (2000). We included a near-duplicate detection module based on Broder's shingling and fingerprinting algorithm.

Containment detection. It is very common for a web page containing an article with its own URL to be included in its entirety in the main page of its home newspaper or blog. Broder also implemented an algorithm for detection of already contained documents (1997). It is not as optimized as near-duplicate detection, but it is possible to use it for small- and medium-sized corpora building. In our downloading process, we included a containment detection method also based on Broder's works.

5 Evaluation

5.1 Corpora obtained by the search engine method

5.1.1 Effect of length of seed word list

As we have already stated, as a first experiment we tested and downloaded 5 different corpora using 5 different lengths of seed word list: 500, 1,000, 2,000, 5,000 and 10,000 words. In all of them, we used combinations of 3 words (as Sharoff suggested for languages with a smaller presence on the web and applied to Romanian), made 12,000 queries and downloaded the first 50 results of each query. The sizes obtained can be seen in table 1. We will now analyse various aspects of the corpora obtained.

Seed word list length	Documents	Words	Words per document	Different websites
500	49,387	81,508,628	1,650.41	4,452
1,000	83,941	105,374,227	1,255.34	4,849
2,000	83,147	119,474,991	1,436.91	4,675
5,000	52,913	129,342,982	2,444.45	4,398
10,000	25,350	85,271,975	3,363.79	3,021

TABLE 1 – Sizes of the collected corpora for each length of seed word list.

In the table we observe that the smaller the seed words list we use, the smaller the resulting corpora are, although the APIs return many more results. The reason for this is that the words in the smaller lists are more common and many pages contain them, but search engines will always be returning the same ones (the ones rated highest in their page rank) and the duplicate filters will remove them; if the words are more rare, fewer pages will contain them and they will not be repeated as much. Nevertheless, for the 10,000 list, the words are so rare that very few pages contain 3 of them and so the corpus obtained is smaller, and will probably be likewise for seed words lists above that.

For all these reasons, it can be concluded that, unlike for English or other languages, a 500-seed word list is not optimal for a language with a moderate presence on the web like Basque. Looking at the sizes, the optimal seed word list length seems to be 5,000, because that is the one that obtains the largest corpus. However, the type of documents from which the corpus has been built is something to take into account, which is shown in fig. 1. The 5,000 seed word corpus is the one containing more words from PDF documents, and PDFs are problematic: it is a visual format instead of an edition one (it does not contain the original continuous text, but rather the coordinates in the page of each line of text, word or even character), so original text extraction from them is never perfect and often very bad. PDF to text converters commit many errors when trying to rearrange the original paragraphs: two-column documents' lines are all messed up, header and footer text are repeated for every page and inserted into other paragraphs... As Fletcher (2007) points out, “PDF does not encode the logical formatting of the text (headings, paragraphs, captions etc.)” and “one problem that plagues all PDF to text converters persists: spaces are occasionally dropped or inserted between or within words”. Because of all of these, the 2,000 seed word corpus may be more appropriate (it is the one that has more words from HTMLs and second in total size), depending on our preference for size or quality.

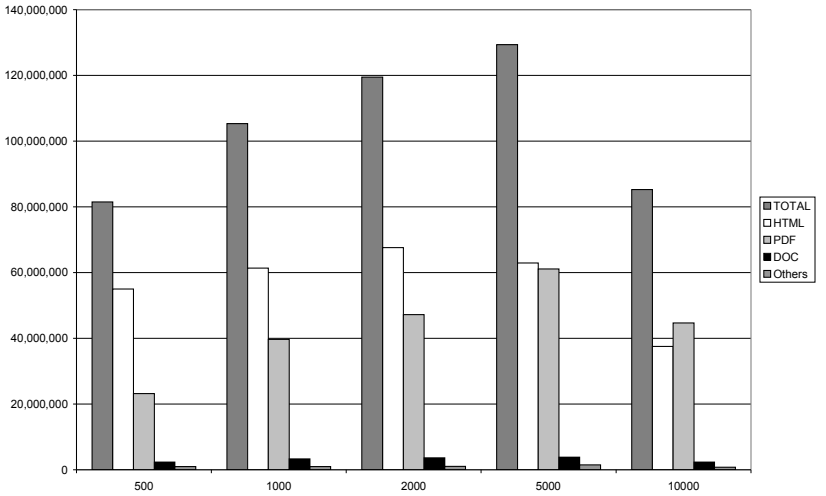


FIGURE 1 – Size in words of the corpora obtained for each seed word list length, total and by type of document.

Another clear difference in the corpora is the average size of the documents (table 1), which grows with the size of the seed word list (logically: if the words are rarer, they are more likely to be found in larger documents). Because corpora are used for linguistic research, the interesting documents for corpora are those that contain a reasonable amount of connected text (Sharoff, 2006). Although we apply the length filter in the collection process and all texts in the corpus have a minimum text, if we are interested in obtaining texts that are as long as possible, then we should opt for corpora obtained with longer seed word lists.

One more thing we have studied is the website variety of the corpora. It is usually interesting for a corpus to be from as many different sources as possible, to be able to analyse more diversity in the use of language; otherwise, style books of media, internal glossaries, etc. can lead to corpora that are too homogeneous. The number of different websites of each corpus is also shown in table 1. As can be seen there, in the last one the number of different websites falls drastically (again, it is logical if that corpus is composed of bigger documents and is smaller), but there is no significant difference among the rest.

Finally, there is one more point worth mentioning: using the search engines method, corpora do not grow continuously at a constant rate. Due to the page ranking these engines use, the same pages tend to appear over and again and are discarded by the duplicates detection, so the bigger the corpus is, the lower its growth rate gets, as the graph in fig. 2 shows. The growth rate is the number of new words obtained for each call to the search engine API, and is represented by the inclination of the curve in the graph.

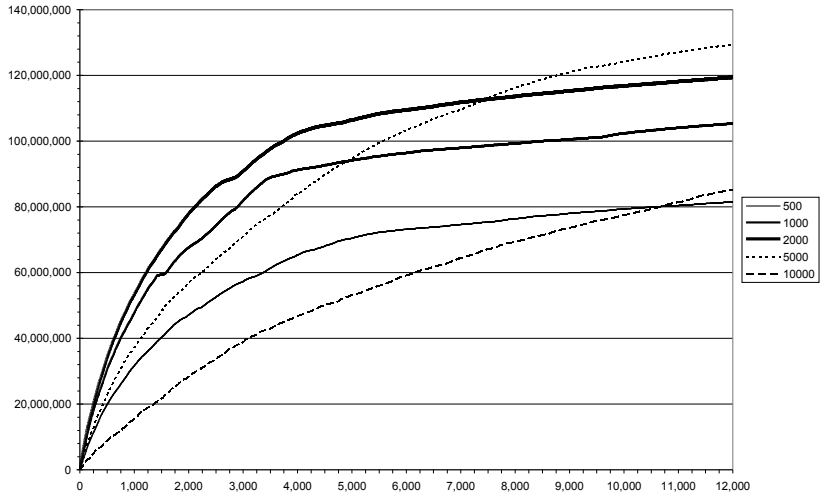


FIGURE 2 – Growth rate of the corpora obtained for each seed word list length.

So, it is not clear whether by using the search engines method we can build corpora as large as we would like to; and even if we could, it would be in a very unproductive way: while with the first 1,000 queries we obtain 37 million words on average (with a maximum 53 million words), in the last 1,000 queries we obtain less than 2 million words on average. And queries to the search engines are not an infinite resource: either they are paid services or have a maximum of calls per month.

5.1.2 Effect of length of combination sent to search engine

In the other experiment, we collected 5 corpora using 5 different search engine word combination lengths: 1, 2, 3, 4 and 5 words. For all of them, the rest of the parameters were the same: a seed word list of 2,000 words was used, 12,000 queries were made and the first 50 results of each query were downloaded. The details of the collected corpora are shown in table 2. Again, we will take a look at some features of these.

Combination length	Documents	Words	Words per document	Different websites
1	36,093	44,692,614	1,238.26	4,089
2	85,562	131,738,927	1,539.69	6,095
3	83,147	119,474,991	1,436.91	4,675
4	41,568	116,371,032	2,799.53	3,824
5	23,108	89,139,248	3,857.51	2,547

TABLE 2 – Sizes of the collected corpora for each combination length.

We can see that there is not a direct correlation between combination length and corpus size. If we send 1-word combinations of a 2,000 word seed list, there are only 2,000 different combinations, as the rest are repeated and do not return new results; therefore, we get the smallest corpus by far. With 2-word combinations we get the largest corpus, but from then on it gradually decreases again, because there will be fewer pages that have all the words. And in this case it is also the 2-word combination corpus that has the most words coming from HTML documents.

Regarding the document length, the same phenomenon as with the seed list length happens: for longer combinations, the size of the documents grows. But the website variety in this case falls for combinations longer than 2. And the dramatic fall in the growth rate also occurs in all these cases.

5.2 Corpus obtained by the crawling method

With the crawling method, and starting with the 1,500 seed URLs from the *Euskara* section of DMOZ, we have so far queued 39,163,290 links, tried to download 6,520,000 of them, successfully downloaded 3,472,166 pages (the rest were not available at the time, or had been discontinued, or gave errors) and included 168,991 out of them in the corpus. The rest were discarded because they were not in Basque (a high percentage of pages in Basque point to pages in other languages, mainly Spanish and English), or were in a format that could not be converted into text, or did not get through the filters (length, duplicate, etc.). The size in words of the downloaded corpus is 115 million. Its features can be seen in table 3.

Documents	Words	Words per document	PDFs	Different websites
168,991	114,565,240	677.94	1,276	5,060

TABLE 3 – Size of the corpora collected by crawling.

Only 1,276 documents come from PDFs, that is, only 0.75%. But the average document length is small, 678 words.

The website variety that could be obtained by the crawling method was one of our concerns. Starting from a set of seed URLs, there is a risk that they may not be enough or good enough, and that many websites are left out because they are not linked to in the initial pages or in the ones recursively linked by these. However, we can see that we have got a number of different websites larger than all but one of the search engines corpora, and compared to that single larger one it is proportionally not much smaller.

It is also interesting to take a look at the growth rate of the corpus (fig. 3). There is certainly a decrease in it, but not that pronounced: in the first million links followed, 23.3 million words were collected, whereas with the last million links we obtained 11.3 million words. It has gone down to 48.5% of the initial rate, while in the search engine corpora, this number is 5.4% on average. This proves that this method has the potential to collect a still much larger corpus.

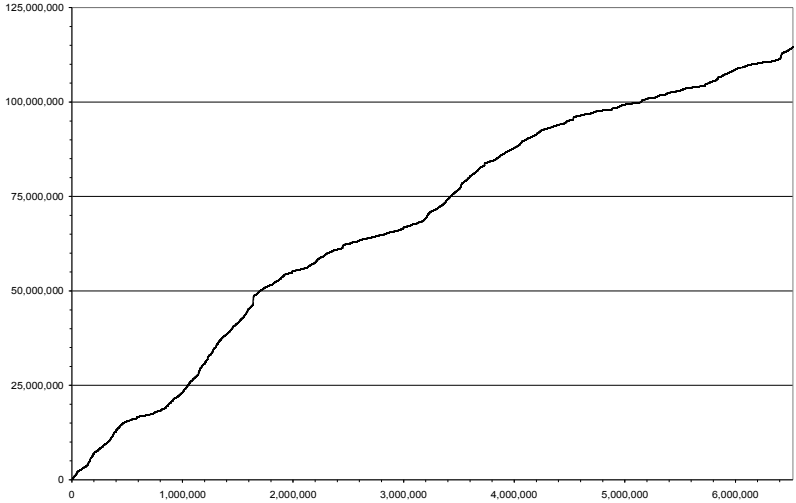


FIGURE 3 – Growth rate of the corpus obtained by the crawling method.

5.3 Qualitative analysis

In the previous subsections, the different corpora obtained were evaluated quantitatively (size, cost, etc.), but a more qualitative evaluation is necessary when corpora are involved, that is, an analysis according to linguistic criteria, because that is what corpora are used for.

5.3.1 Most characteristic words by LLR

There is no absolute method or measure to evaluate the linguistic quality of a corpus. Instead, what is usually done is to compare it with another by using the log-likelihood ratio or LLR association measure (Dunning, 1993) to identify the words that are more characteristic of one with regard to the other (Rayson and Garside, 2000); this is the method used both by Sharoff (2006) and Ferraresi et al. (2008) for evaluating the search engine method corpora and the ukWaC, respectively.

To carry out the evaluation, we chose the largest of the search engine method corpora, i.e. the one obtained with 2000 seed words and 2 combinations (we will call this corpus SE henceforth) and the crawling method corpus (CR henceforth). The number of URLs coinciding in both is only 6,815 (SE is made of 85,562 URLs and CR of 168,991).

Apart from comparing these two corpora with each other, we compared them with two reference corpora: XX. mendeko Euskararen Corpusa (a 4.6 million-word balanced corpus of twentieth century literary texts), henceforth XX, and Lexikoaren Behatokia (an 18.1 million-word corpus of 21st century media texts), henceforth LB.

Ferraresi et al. (2008) used nouns, adjectives and verbs for their analysis, but we also took adverbs and pronouns. We used the lemmas of words. Because in the case of XX we did not have access to the corpus but only to a list of lemma-frequencies and because it was lemmatized with a tagger different to the one we use, we had to discard proper nouns and numbers (the XX frequency list did not contain them) and make some adjustments (there were some deprecated lemmas that are now written in another way).

The most outstanding words of XX compared with any of the other three corpora can be put into three groups: religious words (*jaungoiko* and *jainko* "God", *eliza* "church", *apaiz* "priest", *santu* "saint", *otoitz* "prayer", etc.), pronouns (*hura* "he", *neu* and *ni* "me", *zu* and *hi* "you", *gu* "we", etc.), and words that are scarcely used any more, either because they are now usually said another way, or because they were dialectal or incorrect forms of the times before the standardization, or because they are objects that do not exist or are no longer used (*gizaldi* "century", *eroan* "to take", *ipini* "to put", *ezkero* "if", *pezeta* old currency of Spain, etc.). The prominence of words from the first and third groups is easily understood in view of the difference in temporal deixis across XX and the other three corpora. The greater presence of words from the second group is a normal phenomenon in fiction and narrative texts compared with media and web texts, as Sharoff (2006) also confirmed.

The words characteristic of LB in comparison with any of the others can be divided into two groups: adverbs of time (*atzo* "yesterday", *gaur* "today", *herenegun* "the day before yesterday", *iaz* "last year", *bihar* "tomorrow", etc.) and words from typical media sections such as sports (*talde* "team", *partida* "match", *jokatu* "to play", etc.), politics (*presidente* "president", *gobernu* "government", *nazioarte* "international", etc.), society (*atxilotu* "to arrest", *auzitegi* "court", *epaile* "judge", etc.), culture (*film* "film", *disko* "record", *kontzertu* "concert", etc.) or economy (*euro* "euro", *krisi* "crisis", *lan* "to work", etc.). Both word groups are typical of media texts.

The web corpus we collected using search engines, SE, differs from the other three in words from the administrative domain (*prozedura* "procedure", *lege* "law", *artikulu* "article", *administrazio* "administration", *eranskin* "appendix", *dekretu* "decree", etc.) or the educational domain (*hezkuntza* "education", *ikasle* "pupil", *ikastetxe* "school", *irakaskuntza* "teaching", *irakasle* "teacher", etc.). The prominence of administrative words is greater when compared with the CR corpus. The cause of this might lie in the fact that regional, provincial and local governments publish their official gazettes in PDF format; and, as we saw before, the SE corpus has a large proportion of PDFs, so these might be mostly of an administrative nature.

Finally, the words characteristic of the corpus obtained by crawling, CR, are words typical of web pages (*iruzkin* "comment", *orri* "page", *sare* "net", *erabiltzaile* "user", *web* "web", *blog* "blog", *erantzun* "to comment", *internet* "Internet", *lizentzia* "license", *software* "software", etc.) or of media websites (*albiste* "news", *argazki* "photo", *bideo* "video", *emisio* "broadcast", *kanal* "channel", *telebista* "TV", etc.), month and weekday names (*azaro* "November", *urri* "October", *igande* "Sunday", *astearte* "Tuesday", etc.) or words from the cultural domain (*dantza* "dance", *euskara* "Basque language", *kultura* "culture", *ikastaro* "course", *antzoki* "theatre", etc.). Except for the last, all the groups of words are common in web pages, so we can say that the main feature of this corpus is that it is mostly composed of genuine web pages.

5.3.2 Number of distinct and 'useful' words

Baroni et al. (2009) compared ukWaC and itWaC with reference corpora in each of those languages (the BNC and la Repubblica corpus) looking at three parameters: the number of distinct words in a corpus, the coverage of a corpus within another and the enrichment a corpus gives to another. We have done the same with the four corpora analysed in the previous subsection. We counted the lemmas of all types of words, except proper nouns and numbers (because of the reasons already explained).

Just as in the aforementioned work by Baroni et al., we show the number of distinct words in terms of absolute numbers and of words that occur at least 20 times. This frequency threshold was chosen by them as a rough way of estimating the number of 'useful' words in a corpus, following Sinclair's (2005) claim that at least 20 occurrences of a word are usually needed for an experienced lexicographer to describe its behaviour, and taking into account that low frequency words will not be of any use in NLP applications either. Although admittedly arbitrary, we also used the 'Sinclair cutoff'. The number of distinct words that each corpus has is shown in table 4.

Corpus	Total words	Words $f \geq 20$
XX	53,993	9,147
LB	36,311	12,922
SE	74,132	33,056
CR	64,424	27,238

TABLE 4 – Number of distinct words in each corpus.

As we can see, the number of total and 'useful' words is much greater in the web corpora; this is logical due to their much greater size. However, the high number of total words of the XX corpus is striking: it has almost as many words as the web corpora (which are more than 20 times larger) and much more than the LB corpus (which is almost 4 times bigger). This is due to the fact that a considerable part of the XX corpus is made up of texts from before the standardization of the Basque language and it contains many obsolete, outdated, out-of-use or non-standard words that were tagged manually but which Basque taggers do not usually recognize.

5.3.3 Coverage and enrichment

In order to prove that those 'useful' words attested in the web corpora are the sort of words linguists and lexicographers would be typically interested in, rather than, say, web-related terms of limited general interest, Baroni et al. looked at two measures of overlap, namely coverage and enrichment. The coverage of a corpus in another one is the proportion of words that are above the Sinclair cutoff in both over the total words above this threshold in the first corpus; it can be considered as a rough measure of the extent to which the first corpus is “substitutable” by the second, because it gives an idea of how many of its useful words are also present in the other. The enrichment of a corpus in another one is defined as the proportion of words that are above the Sinclair cutoff in the second corpus but below it in the first, over the total words below the threshold in the first one (to avoid noise in the form of typos or loanwords, only words with at least 10 occurrences are considered); this gives a rough idea of the number of words for which the first does not have enough information, but the second does. We have also calculated these measures, and the statistics obtained are reported in table 5.

Corpora type	Corpora	Coverage	Enrichment	Corpora	Coverage	Enrichment
Classical	XX / LB	57.14%	14.36%	LB / XX	80.71%	38.36%
Classical / Web	XX / SE	26.13%	0.48%	SE / XX	94.44%	83.67%
	XX / CR	31.43%	1.01%	CR / XX	93.59%	77.43%
	LB / SE	36.74%	0.81%	SE / LB	93.99%	83.85%
	LB / CR	44.21%	1.48%	CR / LB	93.19%	79.11%
Web	SE / CR	95.80%	56.24%	CR / SE	78.94%	9.52%

TABLE 5 – Coverage and enrichment of each corpus with regard to each of the others.

It shows that the web corpora cover high above 90% of the classical corpora with an enrichment over them of around 80%, whereas the coverage of the classical corpora over the web ones is normally below 40% and their enrichment is always below 2%; these data are similar to the ones obtained in the aforementioned research by Baroni et al. with ukWaC/BNC and itWaC/La Repubblica.

However, the comparison between the two web corpora, SE and CR, offers surprising results. Although they are of almost equal size, we have seen in the previous subsection that SE contains many more distinct and 'useful' words than CR, and the coverage and enrichment are not symmetrical: CR is almost completely covered by SE (95.80%), but in the other direction this number is only 78.94%; and SE enriches CR by 56.24%, whereas CR only contributes to SE with 9.52% of new words. It looks as if, for equal sizes, the search engine method obtains more linguistically varied corpora than the crawling method. Nevertheless, we have shown that the crawling method can collect much larger corpora, so this deficiency will supposedly be corrected if we continue crawling and enlarging the corpus.

Conclusions

We have proven that both crawling and using search engines are valid methods for obtaining BNC-sized corpora for Basque. With the search engines method, using 2,000 or 5,000 seed words we obtained the largest corpora: the former obtains greater website variety, the latter obtains more PDFs (usually problematic) and larger documents (more connected text). The optimal word-combination length to send to the APIs seems to be 2, because it obtains the largest and most varied corpus with the least number of PDFs. However, if more than 100-150 million words are needed, crawling is the way to go: we have collected a corpus of a size and website variety comparable with those obtained via search engines, with much fewer PDFs and the potential to get much bigger. This corpus is now 115 million words big, but we expect to make it much larger in the near future.

When compared with classical corpora, these web corpora differ in that the search engine ones contain more administrative texts (most probably due to the PDFs of official gazettes) and the crawling one more web-domain texts. Since almost all of the words in the classical corpora are present in the web ones, whilst they provide many new words, we can conclude that collecting large corpora from the web can make a great contribution to Basque corpus building, linguistics and the language in general.

Acknowledgments

This research has been partially funded by the Industry Department of the Regional Government of the Basque Autonomous Community in its Saiotek 2011 call (ref. SA-2011/00216).

References

Areta, N., Gurrutxaga, A., Leturia, I., Alegria, I., Artola, X., Diaz de Ilarraza, A., Ezeiza, N. and Sologaistoa A. (2007). ZT corpus: Annotation and Tools for Basque Corpora. In *Proceedings of Corpus Linguistics 2007*, Birmingham, U.K.

Aston, G. and Burnard, L. (1998). *The BNC handbook: Exploring the British National Corpus with SARA*. Edinburgh University Press, Edinburgh, U.K.

Baroni, M. and Kilgarriff, A. (2006). Large linguistically-processed Web corpora for multiple languages. In *Proceedings of the 11th Conferenceof the European Chapter of the Association for Computational Linguistics (EACL)* (pp. 87-90), Trento, Italy.

Baroni, M. and Ueyama, M. (2006). Building general- and special purpose corpora by Web crawling. In *Proceedings of the 13th NIJL International Symposium* (pp. 31-40), Tokyo, Japan.

Baroni, M., Chantree, F., Kilgarriff, A. and Sharoff, S. (2008). Cleaneval: a competition for cleaning web pages. In *Proceedings of the 6th International Conference on Language Resources and Evaluation (LREC)*, Marrakech, Morocco.

Baroni, M., Bernardini, S., Ferraresi, A. and Zanchetta E. (2009). The WaCky Wide Web: A Collection of Very Large Linguistically Processed Web-Crawled Corpora. *Language Resources and Evaluation Journal*, 43(3): 209-226.

Broder, A. (1997). On the resemblance and containment of documents. In *Proceedings of Compression and Complexity of Sequences 1997* (pp. 21-29), Salerno, Italy.

Broder, A. (2000). Identifying and filtering near-duplicate documents. In *Proceedings of Combinatorial Pattern Matching: 11th Annual Symposium* (pp. 1-10), Montreal, Canada.

Dunning, T. (1993). Accurate methods for the statistics of surprise and coincidence. *Computational Linguistics*, 19(1): 61-74.

Ferraresi, A., Zanchetta, E., Baroni, M. and Bernardini, S. (2008) Introducing and evaluating ukWaC, a very large Web-derived corpus of English. In *Proceedings of the 4th Web as Corpus Workshop (WAC4)* (pp. 47-54), Marrakech, Morocco.

Finn, A., Kushmerick, N. and Smyth, B. (2001). Fact or fiction: Content classification for digital libraries. In *Proceedings of Personalisation and Recommender Systems in Digital Libraries Workshop*, Dublin, Ireland.

Fletcher, W. H. (2006). Concordancing the Web: Promise and Problems, Tools and Techniques. In Hundt, M., Nesselhauf, N. and Biewer, C. (Eds.), *Corpus Linguistics and the Web* (pp. 25–46). Amsterdam, The Netherlands: Rodopi.

Fletcher, W. H. (2007). Implementing a BNC-compare-able web corpus. In Fairon, C., Naets, H., Kilgarriff, A. and De Schryver G.-M. (Eds.), *Building and exploring web corpora* (pp. 43–56). Louvain-la-Neuve, Belgium: Cahiers du Cental.

Ghani, R., Jones, R. and Mladenić, D. (2003). Building minority language corpora by learning to generate Web search queries. *Knowledge and Information Systems*, 7(1): 56-83.

Kilgarriff, A. and Grefenstette, G. (2003). Introduction to the Special Issue on Web as Corpus. *Computational Linguistics*, 29(3): 333-347.

Leturia, I., Gurrutxaga, A., Alegria, I. and Ezeiza, A. (2007). Corpeus, a 'web as corpus' tool designed for the agglutinative nature of basque. In *Proceedings of the 3rd Web as Corpus Workshop (WAC3)* (pp. 69–81), Louvain-la-Neuve, Belgium.

Leturia, I., San Vicente, I., Saralegi, X. and Lopez de Lacalle, M. (2008). Collecting basque specialized corpora from the web: language-specific performance tweaks and improving topic precision. In *Proceedings of the 4th Web as Corpus Workshop (WAC4)* (pp. 40–46), Marrakech, Morocco.

Leturia, I., Gurrutxaga, A., Areta, N., Pociello, E. (2008). Analysis and performance of morphological query expansion and language-filtering words on Basque web searching. In *Proceedings of the 6th International Conference on Language Resources and Evaluation (LREC)*, Marrakech, Morocco.

Leturia, I., San Vicente, I. and Saralegi, X. (2009). Search engine based approaches for collecting domain-specific Basque-English comparable corpora from the internet. In *Proceedings of the 5th International Web as Corpus Workshop (WAC5)* (pp. 53–61), Donostia/San Sebastian, Spain.

Rayson, P. and Garside, R. (2000). Comparing corpora using frequency profiling. In *Proceedings of Workshop on Comparing Corpora of ACL 2000* (pp. 1–6), Hong Kong, China.

Renouf, A., Kehoe, A. and Banerjee, J. (2006). WebCorp: an Integrated System for WebText Search. In Hundt, M., Nesselhauf, N. and Biewer, C. (Eds.), *Corpus Linguistics and the Web* (pp. 47–67). Amsterdam, The Netherlands: Rodopi.

San Vicente, I. and Manterola, I. (2012). PaCo2: A Fully Automated tool for gathering Parallel Corpora from the Web. In *Proceedings of the 8th International Conference on Language Resources and Evaluation (LREC)*, Istanbul, Turkey.

Saralegi, X. and Leturia, I. (2007). Kimatu, a tool for cleaning non-content text parts from html docs. In *Proceedings of the 3rd Web as Corpus Workshop (WAC3)* (pp. 163–167), Louvain-la-Neuve, Belgium.

Sharoff, S. (2006). Creating General-Purpose Corpora Using Automated Search Engine Queries. In Baroni, M. and Bernardini, S. (Eds.), *WaCky! Working Papers on the Web as Corpus* (pp. 63-98). Bologna, Italy: Gedit Edizioni.

Sinclair, J. McH. (2005). Corpus and text – Basic principles. In: Wynne , M. (Ed.), *Developing linguistic corpora: A guide to good practice* (pp. 1–16). Oxford, U.K.: Oxbow Books.

Approximate Sentence Retrieval for Scalable and Efficient Example-based Machine Translation

Johannes Leveling *Debasis Ganguly* *Sandipan Dandapat* *Gareth J.F. Jones*
Centre for Next Generation Localisation (CNGL)
School of Computing, Dublin City University
Dublin, Ireland
{jleveling, dganguly, sdandapat, gjones}@computing.dcu.ie

ABSTRACT
Approximate sentence matching (ASM) is an important technique for tasks in machine translation (MT) such as example-based MT (EBMT) which influences the translation time and the quality of translation output. We investigate different approaches to find similar sentences in an example base and evaluate their efficiency (runtime), effectiveness, and the resulting quality of translation output. A comparison of approaches demonstrates that i) a sequential computation of the edit distance between an input sentence and all sentences in the example base is not feasible, even when efficient algorithms to compute the edit distance are employed; ii) in-memory data structures such as *tries* and *ternary search trees* are more efficient in terms of runtime, but are not scalable for large example bases; iii) standard IR models which only cover material similarity (e.g. term overlap), do not perform well in finding the approximate matches, due to their lack of handling word order and word positions. We propose a new retrieval model derived from language modeling (LM), named LM-ASM, to include positional and ordinal similarities in the matching process, in addition to material similarity. Our IR based retrieval experiments involve reranking the top-ranked documents based on their true edit distance score. Experimental results show that i) IR based approaches result in about 100 times faster translation; ii) LM-ASM approximates edit distance better than standard LM by about 10%; and iii) surprisingly, LM-ASM even improves MT quality by 1.52% in comparison to sequential edit distance computation.

KEYWORDS: Example-Based Machine Translation, Information Retrieval, Approximate Sentence Matching, Edit Distance.

Proceedings of COLING 2012: Technical Papers, pages 1571–1586,
COLING 2012, Mumbai, December 2012.

1 Introduction

The quality of data-driven machine translation (MT) mostly depends on the size of parallel data available for training. Although statistical MT is considered as the state-of-the-art, its one limitation is that it discards the training data once the translation model and language model have been generated. This sometimes can lead to poor quality translations because translation context is limited by the value of n. In contrast, example-based MT (EBMT) usually stores the full sentence pairs in source and target data in an example base and uses translations of sentences similar to the input sentence as a template for translation. Thus, EBMT systems can often better capture long range dependencies and rich morphology. State-of-the-art MT systems comprise both statistical and example-based MT components (Dandapat et al., 2012). Typical EBMT systems (Somers, 2003; Nagao, 1984) comprise three processing stages to translate an input sentence Q with information from an example base of sentence pairs (S_i, T_i), where S_i and T_i are in source and target language, respectively: i) Matching, where sentences similar to Q are identified, and the translation template T_i with the maximum similarity between Q and S_i is used as a skeleton for the translation. ii) Alignment, where the matching parts of Q and S_i are found to identify the remaining translation gaps and translation alternatives for the mismatches are obtained from TM. iii) Combination, where the translation fragments found in the previous steps are aggregated into the final translation. The matching stage involves maximising a similarity between the input Q and all source language sentences S_i in the example base. The most widely used similarity measures in EBMT matching are based on the edit distance (ED), also known as Levenshtein distance (LD) (Levenshtein, 1966). Thus, the matching step of an EBMT system is a time-consuming process with runtime depending on the complexity of the similarity measure and the size of the example base. EBMT systems can usually only handle a moderate size example base in the matching stage. However, using a large example base is important to ensure high quality MT output. In order to make MT applicable for larger example bases while improving or maintaining its speed, we investigate different approaches to efficient approximate sentence matching: a) sequential algorithms for the computation of the similarity, where speed improvements are based on limiting the number of symbol comparisons (e.g. cut-off heuristics); b) data structures, where a traversal of the structure can be employed to compute the similarity; c) sentence indexing and retrieval, where aspects of similarity are traditionally modeled by factors based on frequencies such as *tf* and *idf*, but in the case of approximate sentence retrieval should also include word order and position.

We demonstrate that sequential (brute-force) approximate matching becomes too expensive for large example bases. Using in-memory data structures is efficient with respect to runtime, but requires much more memory. Information retrieval based on standard term weighting functions is not appropriate for finding best matches. Standard information retrieval (IR) is more efficient than sequential comparison, but not accurate enough in finding the most similar matches in top ranks. We propose a hybrid, two-stage approach. First we retrieve sentences from the example base, scoring the results based on our proposed edit distance approximating retrieval model. Secondly, we rerank the results by their true LD score. Our approach thus restricts the edist distance computation to the set of top-ranked retrieved sentences instead of the full example base. The proposed retrieval model uses positional indexing and retrieval, reflecting three aspects of similarity: how similar term positions are, how similar the word order is, and how similar the sets of terms are. The approach is shown to be fast (i.e. efficient and more scalable), accurate in terms of mean reciprocal rank (MRR) (i.e. effective in retrieving approximate matches), and can yield even better translations than the sequential approaches (i.e. results in a higher BLEU score (Papineni et al., 2002)).

The rest of this paper is organized as follows: Section 2 introduces related work; Section 3 presents

approaches to approximate sentence matching; Section 4 describes our proposed IR model; Section 5 explains our experimental setup, data and evaluation metrics; Section 6 presents and discusses the results. Section 7 concludes with an outlook on future work.

2 Related Work

Despite of a long history of research in IR and MT, there is still relatively little research on applying IR methods for MT. Two years before Levenshtein proposed the edit distance in 1966 Levenshtein (1966) and Faulk (1964) argued that three aspects of similarity should be considered for approximate sentence matching for translation: 1. positional similarity of items (e.g. words occur in the same positions), 2. ordinal similarity (e.g. words have the same order), and 3. material similarity (e.g. the sets of words are similar). He investigates different similarity metrics in different languages and demonstrates that a high sentence similarity in the source language correlates with a high similarity in the target languages. The edit distance has been widely used in diverse applications such as approximate name matching (Berghel and Roach, 1996), in the biomedical domain for the comparison of gene sequences (Yap et al., 1996), for spelling correction and dictionary search (Boytsov, 2011). and in music retrieval (Mongeau and Sankoff, 1990). Improvements of the runtime complexity of the original Levenshtein algorithm include Ukkonen's cut-off heuristics (Ukkonen, 1985b,a) and Berghel and Roach's extension of this approach. Finally, Levenshtein automata (Schulz and Mihov, 2002) have been suggested as an efficient approach to spelling correction when an upper bound for the distance is known in advance. Due to the large number of different symbols (words) and because an upper bound for the distance is not known in advance, we did not investigate Levenshtein automata for the experiments described in this paper. Alternative approaches to approximate matching include q-grams (Gravano et al., 2001) (i.e. character n-grams), variants of the longest common subsequence (Lin et al., 2011), and affine gap alignment (Needleman and Wunsch, 1970). Navarro (2001) presents an excellent survey on different approaches to approximate string matching.

The standard application of MT in IR is in Cross-language IR, where given a query in the source language, documents in a target language have to be retrieved (Di Nunzio et al., 2008). IR techniques have been applied to machine translation only recently. (Hildebrand et al., 2005) apply IR to improve the quality of the training data for a statistical MT system. They adapt the language model for translation by selecting similar sentences from a large training corpus for training data and experiment with *tf-idf* and cosine similarity and Okapi's BM25 model (Robertson et al., 1998), finding no significant difference in their performance. They conclude that adaptation of the LM is helpful in improving translation quality. Similar to the experiments described in this paper, Koehn et al. apply IR and use a combination of n-gram matching and A* pruning (Koehn and Senellart, 2010). However, they do not report individual results for the retrieval effectiveness, only optimize the speed of approximate matching, and do not report the effect of applying their matching approach to TM in detail. They achieve a processing time per query of 247ms for sentence matching in the JRC-Acquis corpus and 4.3ms in a smaller corpus with product information. Dandapat et al. (2012) investigate two methods to achieve scalable EBMT. First, they try bucketing sentences by length to limit the number of sentences in the example base that have to be compared against the input, assuming that similar sentences will have similar length. Thus, the best match is not guaranteed to be found when this heuristics is used. Their second method includes IR based on standard language modeling (LM), but individual results for the IR stage are not reported. For the retrieval experiments described in this paper, we use LM as a baseline. We investigate parameter settings and preprocessing options to obtain the best baseline for our proposed approach.

The application of edit distance for problems such as spell checking is different to its application in EBMT and TM in several aspects: 1. Typically, character sequences (i.e. strings) are considered for comparison. For our experiments, we aim at computing the similarity for sequences of symbols or tokens (i.e. words and all punctuation marks). 2. Given a large dictionary for approximate *string* matching, the distance of an input word to some word in the dictionary is usually quite low. This does not have to be the case for approximate *sentence* matching, where the size of the alphabet (the number of possible symbols) is much higher (i.e. different characters vs. different words) and the minimum edit distance can also be quite high. Thus, a general minimum edit distance threshold can not be specified in practice. 3. Providing an upper bound for the edit distance can speed up the computation considerably. For the application we consider here, no upper limit for the number of mismatches is known in advance. 4. We are interested in finding all matches with the highest similarity as opposed to finding a single close match or one best match. For the experiments described in this paper, we did not employ heuristics such as bucketing or assuming an upper bound for the edit distance to speed up processing. These methods can actually be utilized to reduce runtime for our proposed approach even more, but at the cost of the loss of accuracy.

Our proposed LM-ASM retrieval model is an extension of language modeling. An extension to language modeling, known as positional language modeling (PLM), includes a proximity heuristic rewarding a document where matched query terms occur close to each other (Lv and Zhai, 2009). The PLM favours documents where the query terms appear in the same order as that in the query. In PLM, relative term positions are modelled via their term context. In constract, our proposed retrieval model for approximate sentence matching takes into account the absolute term positions.

3 Approaches for Approximate Sentence Matching

Approximate sentence matching (ASM) is the problem of finding the sentences with the highest similarity to a given inpuit sentence in a collection of sentences. The matching stage of EBMT can be considered as an instance of ASM. It identifies sentence pairs (S_i, T_i) from the example base where S_i closely matches with the input sentence Q. The EBMT system considers the translation T_i where Q has maximum similarity to the source sentence S_i to build a skeleton for translation of the input. From the perspective of IR, we are trying to find sentences (documents) in the example base (document collection) which have a high similarity (document score) with the input sentence (query). The results are then used in the alignment and recombination stage of EBMT to produce a translation. The set of sentences with the highest Levenshtein score (LS), i.e. the highest fuzzy match score, is computed by a sequential approach and corresponds to the set of relevant documents. Relevance assessment for IR is typically based on manual assessment of pooled results, whereas in the experiments described in this paper, the relevant (correct) results are determined based on a sequential computation of fuzzy match scores. We introduce similarity metrics between sequences Q and D with length $|Q|$ and $|D|$, where i and j denote an index in a sequence and Q_i and D_j denote the ith and jth symbol in sequence Q and D.

3.1 Sequential Search

Levenshtein Distance Algorithms ($\mathbf{LD}_{WF}$ and $\mathbf{LD}_{BR}$). The Levenshtein distance or edit distance is typically employed in EBMT to find the closest matching sentence in the example base. The edit distance for two sequences Q and D is defined as the minimum number of edits, i.e. symbol insertions, deletions and substitutions, needed to transform Q into D (Levenshtein, 1966). The sequential computation of the distance can be computationally expensive, because the Levenshtein algorithm has a runtime complexity of $O(|Q|$ x $|D|)$, which has to be calculated for every sentence

in the example base. The straightforward solution to compute the edit distance is via dynamic programming, where a $|Q|$ x $|D|$ matrix is filled following a recursive schema (Wagner and Fischer, 1974). For all approaches based on computing the exact LD, we use the normalized Levenshtein similarity score (fuzzy match score), which is computed as

$$LS(q, d) = 1 - LD(Q, D)/\max(|Q|, |D|) \quad (1)$$

Several improvements have been proposed to improve runtime complexity. Ukkonen's Enhancend Dynamic Programming Approximation algorithm (Ukkonen, 1985b) for computing edit distance has the worst case complexity $O(|Q|$ x $B)$, where B is an upper bound of the edit distance. The improvement results from the fact that $distance(i, j)$ values are non-decreasing along any given diagonal. Only those $distance(i, j)$ values p for which i is the highest numbered row on which p occurs on diagonal k for a given threshold k have to be computed. The modified Berghel-Roach algorithm (Berghel and Roach, 1996) is an extension of Ukkonen's cut-off heuristic. It achieves 42% speed-up compared to Ukkonen's approach for name matching and is 79% faster than the Wagner-Fischer algorithm. We employ the Berghel-Roach algorithm (LD_{BR}) to investigate speed-up by an algorithm with lower runtime complexity. As a baseline for runtime, we compute the edit distance based on the Wagner-Fischer algorithm (LD_{WF}).

Longest Common Subsequence (LCS). Our experiments to improve MT with IR methods are based on the assumption that the sentence with the minimum edit distance in the source language is a good template for its translation. This assumption is widely accepted and has been validated for translation memory (TM) (Sikes, 2007). We compare our experimental results for the edit distance with corresponding results for the longest common subsequence (LCS) (Gusfield, 1997). The LCS is the longest shared subsequence of – not necessarily consecutive – symbols in two sequences. The corresponding score is computed by replacing the edit distance LD in Equation 1 with the LCS and calculating the scores by iterating over all examples in the example base.

3.2 Data Structures

Efficient data structures have been proposed for fast approximate lookup operations in dictionaries (e.g. for spelling correction), but to the best of our knowledge, the following approaches have not been investigated for ASM, i.e. for computing the edit distance for sequences of words. In fact, the existing implementations consider only characters as symbols, while our implementation abstracts from that view and allows any type of symbol as the basic element of a sequence.

Tries (TR). A trie is an ordered tree data structure that can be used to store key-value pairs where the keys are sequences of symbols (Gusfield, 1997). Each node in a trie stores a single symbol of the corresponding sequence and a node represents the prefix of the key on the path of the root up to that node. All the children of a node have a common prefix of the sequence associated with that node, and the root represents the empty sequence. Values are associated with nodes corresponding to the end of a sequence, i.e. leaf nodes and some inner nodes. The main idea to efficiently compute the edit distance with tries is to compute only the part of the distance matrix up to the length of the current prefix, so that redundant computations are avoided for sequences sharing the same prefix. For our experiments, we regard sentences comprised of tokens as sequences of symbols and store the corresponding document ID (i.e. sentence ID) as a value.

Ternary search trees (TST). Ternary search trees (Bentley and Sedgewick, 1997) are tree structures where each node has three children. Similar to hash tables, ternary search trees can be em-

ployed as an associative structure to store key-value pairs (here: pairs of sentence and document ID). Each node of a ternary search tree stores a single symbol for comparison with a symbol of a search key, and pointers to three children which determine which subtree to search next, based on the result of a comparison, i.e. lower, equal or higher lexicographical order. As for tries, only part of the edit distance matrix – a single row – is computed at a given node.

BK-trees (BKT). Burkhard and Keller (1973) proposed BK-trees for efficient file searching. A BK-tree is a metric tree adapted to discrete metric spaces, defined by a distance metric $D(x, y)$. A BK-tree can be formed as follows. An arbitrary element a is selected as the root node. The root node may have zero or more subtrees. The kth subtree is recursively built of all elements y such that $D(x, y) = k$. As BK-trees support distance metrics, and not similarities, we directly employ the edit distance as a metric. To improve runtime, we use a distance threshold of 2, i.e. subtrees exceeding the threshold distance are not visited. This setting was obtained empirically and chosen to yield the best results in terms of effectiveness and efficiency. Lower values result in finding fewer correct results, higher values more than double the runtime because more nodes have to be visited. Note that this cutoff heuristic results in lower runtime but also lower effectiveness.

3.3 Information Retrieval Approaches

The problem with brute force, sequential computation is that it involves a linear search for the closest matching sentence by computing edit distance between the given query and each sentence in the example base. The time complexity is thus $O(N)$, N being the number of sentences in the collection, which clearly makes the brute-force approach infeasible for large collections. In-memory data structures could be much faster in LD computation, but have high space complexity, i.e. require a huge memory and are thus not scalable for very large collections.

Standard IR models. Information retrieval (IR) is concerned with finding relevant documents from a document collection, given a query (Manning et al., 2008). IR involves computing document scores for a given query by aggregating scores from the local importance of a term in a document (e.g. the term frequency, *tf*) and a global importance factor of the term in the document collection (e.g. the inverse document frequency, *idf*). Generally speaking, a retrieval model aims to compute the similarity between a document and a query. As a simplification, queries and documents can be defined as vectors over the vocabulary term space, so that the similarity can be computed as a dot product between query and document term vector, i.e. $sim(d, q) = \sum_{i=1}^{|V|} d_i q_i$. In standard IR, similarity features such as the word position in the query or a document are typically ignored. Features such as word ordering are modeled implicitly when indexing word n-grams.

IR makes use of *inverted list* structures which are a combination of in-memory data structures and file structures. The list of documents where a term occurs, namely the *postings list* for a term is loaded into memory from files hashed by the term identifier. This combination results in very fast retrieval and also makes retrieval scalable to very large collections. The inverted list data structure is suitable for computing the query-document similarity, because document vector term weights i.e. the d_i values are read from the postings list of the ith query term, and accumulated over all query terms to calculate the final similarity. The computational complexity of the similarity is thus $O(\sum_{i=1}^{n} s_i)$, where n is the number of query terms and s_i is the size of the postings list for the ith query term. In practise, n is a small constant and $s_i << N$, where N is the number of document in the collection. Hence the method is very fast.

Although the inverted list structure of IR looks promising, it should not be directly applied to

the ASM problem because of basic differences: In standard IR, documents are much longer than queries, compared to ASM, where documents and queries have similar length. Preprocessing techniques such as removing stopwords and eliminating punctuation, and applying a stemmer are common for IR and reduce the index size and the retrieval time, because highly frequent terms such as stopwords are excluded from the inverted lists.[1] In ASM, keeping all terms (i.e. words and punctuation) is important for exact matching of symbols and to retain the syntactic sentence structure for translation. Relative and absolute term positions are usually ignored in standard IR (except for phrase search, which partially models relative term position). In ASM, it might be important to implicitly retain word order as a similarity factor by matching n-grams.

The LD computation for ASM cannot directly be performed, i.e. it is not possible to compute the exact edit distance between a document and a given query within an IR application. One thus has to strive for designing a similarity function which produces rankings as close as possible to the ranking as computed by the LD computation. However, this proves difficult because standard IR similarity scores work on the principle of material similarity, i.e. a document is potentially ranked higher if it contains a higher number of query terms. For each ith query term match, the contributing factor of $d_i q_i > 0$ is added to the document score. However, material similarity alone may not be a good approximation for ranking based on LD.

Towards better sentence similarity metric for LD approximation. Three measures of similarity are crucial for ASM, namely the positional, ordinal and material similarity (Faulk, 1964). A direct application of standard IR methods for ASM can only estimate the last one, i.e. the material similarity. In addition to material similarity, it is required to consider the following, particularly for finding closest edit distance sentences for EBMT (Somers, 2003). Firstly, a sentence from the example base with a partial exact match of words (identical surface forms of words) in the query sentence is helpful because the longer the matched portion, the higher the likelihood is of generating a target translation of good quality. Secondly, identical term positions between Q and D indicate structural similarity, which is helpful in inferring the inherent reordering between D and the translation. Thirdly, similarity in length between Q and D requires substitution operations between mismatched portions during the *recombination* stage of EBMT. This is advantageous over insertion, since the positions in the translation of D to substitute are exactly known during recombination. To the best of our knowledge, a retrieval model incorporating aspects of material, ordinal, and positional similarity has not yet been proposed for ASM.

4 ASM Retrieval Model

The objective of ASM is to estimate the edit distance ranking as accurately as possible without computation of the actual distances. Let Q be a query sentence and D a sentence for which we need to estimate its edit distance from Q.

Language modeling (LM) is a state-of-the-art retrieval model (Ponte and Croft, 1998), where a document D is scored by the posterior probability of generating the given query Q, i.e. $P(D|Q)$ (Hiemstra, 2000). This in turn is estimated at indexing time from the prior probability $P(Q|D)$ using the assumption that the query terms can be generated independently from a document D by a linear combination of two events of either generating a query term q from D (i.e. the *tf* component) with probability λ, or generating it from the collection (i.e. the *idf* component) with probability $(1 - \lambda)$, as shown in Equation 2 ($cf(q)$ and cs denote the collection frequency of term q and the

[1] Note that while we aim at reducing runtime, we still include stopwords and punctuation for indexing in our main experiments, which is contrary to a standard IR setup.

collection size respectively). Equation 2 does not take into account the relative (or even absolute) positions of documents and query terms.

$$P(D|Q) \propto P(D)P(Q|D) = P(D) \prod_{q \in Q} P(q|D) = P(D) \prod_{q \in Q} \lambda \frac{tf(q,D)}{|D|} + (1-\lambda)\frac{cf(q)}{cs} \quad (2)$$

We propose an extension of the LM retrieval model for ASM, which we call LM-ASM. In contrast to standard LM, the probability of generating the query Q from a document D in case of LM-ASM has two components: i) $P_{len}(Q|D)$, which denotes how close is the length of Q to the length of D; and ii) $P_{pos}(Q|D)$, which is representative of how likely it is for a query term to belong to the same absolute position where it occurs in D. The first likelihood component minimizes the likely number of insertions and deletions while transforming D to Q and is given by Equation 3. Note that this likelihood function decreases with an increase in the absolute difference of the lengths of D and Q, attaining a maximum for $|D| = |Q|$.

$$P_{len}(Q|D) = \min(|Q|, |D|) / \max(|Q|, |D|) \quad (3)$$

To model the generation of a query term in its absolute position from the corresponding absolute position of that term in a document D, we use the following notation. Let $pos(q, D)$ be the set of absolute positions of a term q in a document D. In contrast to standard LM, for ASM, we also need to consider the current position of a query term. Let q_i be the query term at position i. Thus, even if the same query term q occurs in multiple positions in a query, we compute $P_{pos}(q_i|D)$ for all positions i where q occurs.

$$P_{pos}(q_i|D) = 1 \,/ \min_{j \in pos(q_i, D)} (|j - i| + 1) \quad (4)$$

In Equation 4, $|j - i|$ denotes the absolute value of the minimum difference of term positions. The final probability of $P(Q|D)$ is thus given by multiplying the two components $P_{len}(.)$ and $P_{pos}(.)$ of Equations 3 and 4 into Equation 5.

$$P(Q|D) = P_{len}(Q|D)P_{pos}(Q|D) = \frac{\min(|Q|, |D|)}{\max(|Q|, |D|)} \prod_{i=1}^{|Q|} \Big(\frac{1}{\min\limits_{j \in pos(q_i, D)} |j - i| + 1} \Big) \quad (5)$$

Analogous to LM, in order to avoid underflow for multiplications of small numbers, we implement the positional score with the log transform (omitted for brevity). A closer look at Equation 5 reveals that for every matching query term in a document D, we include the reciprocal of the absolute differences of the term positions in the score. The higher this difference, the lower is the similarity component being aggregated. In case a query term does not occur in D, nothing is added to the score. We propose to use this approach for approximate sentence retrieval and call it LM-ASM.

5 Experimental Setup

Data. We conduct experiments to report the accuracy of our EBMT approach for English-Turkish (EN-TR) and English-French (EN-FR) translation tasks. In order to compare the translation performance of our approaches, we use the EBMT system described in (Dandapat et al., 2012), which follows the framework in (Nagao, 1984) as a baseline. We do not combine our EBMT approach with an SMT system, as we focus on effectiveness and efficiency evaluation of IR for ASM to improve the EBMT component. For further details, we refer the reader to (Dandapat et al., 2012).

Name	Domain	Language pair	Example base	Avg. sentence length	Test data
IWSLT 09	Travel	EN-TR	19,972 sentences	9.5 words	414 sentences
EMEA	Medicine	EN-FR	250,806 sentences	18.8 words	10,000 sentences

Table 1: Statistics on the two evaluation data sets.

The two data sets used for all our experiments represent two language pairs with parallel data of different size and type. Statistics for the data sets, IWSLT 09[2] and European Medicines Agency[3] (EMEA), are shown in Table 1. The original EMEA corpus comprises approximately 1M sentences, including many duplicates. We discard duplicates and consider only sentences with unique translation equivalents. Note that, to the best of our knowledge, efficient solutions for ASM for EBMT that scale up to millions of sentences have not been reported in the literature.

Evaluation Objectives. Our experiments focus on different research questions: *Which approach is the most efficient and scales up to large example bases for EBMT?* To investigate this question, we conduct experiments using different algorithms and data structures for EBMT matching as described in Section 3 and 4 and compare them with our proposed two stage approach based on IR methods and reranking IR results.

Which approach has the highest accuracy when a trade-off between efficiency and effectiveness becomes necessary? Naturally, sequential computation will yield the highest accuracy (e.g. MRR) of results. However, when using IR methods, we expect a drop in effectiveness because the edit distance similarity is only approximated by the scoring method. Related to this question is the aspect of preprocessing (word matching), e.g. *Should stopwords or punctuation be removed or should a stemmer be applied?* We expect that stopwords and punctuation are actually important for ASM and that stemming would decrease retrieval effectiveness. We perform experiments using different preprocessing methods and indexing word-level n-grams.

Which approach leads to the highest translation quality? With the sequential computation, we will find all and only exact approximate matches, i.e. all sentences with the exact maximum edit distance score LS. Using IR, potentially only a subset of all "relevant" sentences[4] will be identified. A related question is *How many documents should be retrieved in the IR stage?* One objective in ASM is to restrict the number of retrieved documents to a small number, e.g. 10 or 20, since it is more efficient to retrieve a small number of sentences; and it is more efficient to rerank a small set of sentences by the true LS (e.g. by computing $LS(Q, D)$ for the top-ranked results). For example, retrieving 100 or more documents in the retrieval step could result in higher runtime, because the set of retrieved documents has to be reranked again. We try to determine the number of documents needing to be retrieved empirically to achieve this trade-off, i.e. achieve a satisfactory MT performance without sacrificing computation speed.

Evaluation Metrics. The experiments described in this paper aim at maintaining or improving three aspects of performance for large example bases: *efficiency*, *effectiveness*, and *translation quality*, described as follows. 1. *Efficiency:* A lower runtime implies that larger example bases can be used and the MT system becomes more scalable. We report the time for reading and indexing the documents in seconds (IT), computation time (CT), and the average time per sentence

[2] http://mastarpj.nict.go.jp/IWSLT2009/2009/12/downloads.html
[3] http://opus.lingfil.uu.se/EMEA.php
[4] Relevant in this context means the best translation template in the target language.

(AT), exlcuding indexing time. 2. *Effectiveness:* As some similarity scores and the proposed retrieval approach approximate the edit distance instead of actually computing it, the accuracy of finding correct matches with minimum LD (more specifically: with maximum LS) is measured. We employ IR metrics such as mean average precision (MAP) and the mean reciprocal rank (MRR) (Voorhees, 1999). The significance of MRR is that the closer the MRR is to 1, the fewer documents need to be reranked using their true edit distances. We also report the number of queries in the test set with a reciprocal rank (RR) of 1 (i.e. the top ranked document is relevant) and with $|RR > 0|$, i.e. the result set contains at least one relevant result. A reciprocal rank of 1 means that a correct result is already at rank 1, and $RR > 0$ implies that there is at least one correct result contained in the set of retrieved results. In addition, we report the number of retrieved documents (#ret) and the number of relevant documents retrieved, i.e. *recall* (#rel_ret). Achieving a high recall enables an additional level of processing to choose a particular sentence for the EBMT matching phase among a set of candidate sentences with equal LS values with respect to the input sentence. Since developing an efficient tie-breaking heuristic is outside the scope of this paper and we are not aware of any such existing work, we simply resolve ties by choosing the sentence with minimum identifier for all our experiments. However, we include recall and MAP as evaluation metrics to justify the relative usefulness of a system under the presence of a tie-breaking mechanism. 3. *Translation quality:* Finally, we compute how good the final translation output is, based on the standard MT evaluation score BLEU (Papineni et al., 2002).

Relevance Judgements. Since the objective is to approximate edit distance scores, the target documents (relevant documents) are those with the maximum LS scores with respect to the query. The "relevance judgements" are thus obtained from computing the sentence pairs in the example base with maximum LS score (see Equation 1) for a query, presuming that all sentences with the minimum distance (or highest sentence similarity) are correct or relevant. This can lead to a high number of relevant results for queries which have many exact or near-matches in the data. For the EN-TR data, there are 4.74 relevant results per query on average; queries in the EN-FR data have 16.38 relevant results per query on average. To resolve these ties for EBMT, the highest scoring document with the lowest document ID is selected for subsequent translation stages.

Test System. The test system is a standard PC with a 3.16 GHz Core 2 Duo CPU and 8 GB RAM. The retrieval engine is SMART[5], which was modified to support positional indexing and language modeling. In the postings list of every term, we store the document ID for each document the term appears in and the absolute position of that term in that particular document. The EBMT system is an implementation of the translation by analogy approach described in (Nagao, 1984).

6 Experimental Results

Effect of Preprocessing. ASM is different from standard IR. Therefore, we explore how to optimize the IR settings for the ASM problem. We have argued that standard IR preprocessing such as stopword removal and stemming may not be suitable for ASM. Table 2 shows the results for different IR approaches. In this section, we are only interested in the retrieval effectiveness which is measured in terms of how close the retrieved set is with respect to the reference set of minimum edit distance sentences from the training set for each test sentence. Results are obtained by retrieving 50 sentences. The results in Table 2 can be interpreted as follows. Simple retrieval methods such as raw term frequency (*tf*) or *tf-idf* do not perform well for ASM. In contrast to standard IR, stopword removal and stemming decrease performance. The number of retrieved documents (#ret) is also

[5] ftp://ftp.cs.cornell.edu/pub/smart/

Run	Parameter			IR effectiveness			
Name	SR	ST	n	#ret	#rel_ret	MAP	MRR
tf	N	N	1	20700	615	0.289	0.289
tf-idf	N	N	1	20700	710	0.338	0.346
LM	N	N	1	20700	**1270**	**0.593**	**0.617**
LM-ASM	N	N	1	20700	**1295**	**0.638***	**0.699***
LM	N	Y	1	18944	1148	0.513	0.547
LM-ASM	N	Y	1	18944	1174	0.579*	0.641*
LM	Y	Y	1	13349	296	0.180	0.520
LM-ASM	Y	Y	1	13349	265	0.183	0.617*
LM	N	N	2	20700	1414	**0.658**	0.688
LM-ASM	N	N	2	20700	**1482**	0.650	**0.733***

Table 2: IR results on EN-TR data for retrieval of 50 documents per query. Parameter settings include stopword removal (SR), stemming (ST), and the use of n-grams.

lower compared to the runs where neither stopword removal nor stemming is performed, because some query sentences comprise of only stopwords e.g. "I am sorry". Stemming also degrades retrieval performance. So the the question *'Should stopwords or punctuation be removed or should a stemmer be applied?'* can be answered with no. In our experiments, all forms of preprocessing degrade performance.

We employ LM as the baseline for our retrieval experiments. In order to obtain the best LM baseline, we conducted experiments with varying values for the λ parameter in the interval $[0, 1]$. Selecting $\lambda = 0.99$ yields the highest MAP (0.593). This explains that *idf* is not important for ASM. Normalized *tf* (i.e. $tf(t, D)/|D|$) suffices to approximate LS (see Equation 2).

Our proposed method LM-ASM produces significantly better results both in terms of MAP and MRR as compared to LM[6]. LM-ASM, unlike LM, takes into consideration the term position differences between document words and the given query words, thus better estimating the number of edit operations and hence the edit distance. Note that LM-ASM does not rely on collection statistics such as *idf*.

We also experimented with bigrams ($n = 2$) on the best IR settings for the unigrams i.e. without stopword removal and stemming. The results are shown in the last two rows of Table 2. It can be seen that the use of bigrams yields significant improvement in MRR (0.733 vs. 0.699) at the cost of an insignificant decrease in MAP. The reason for the improvement can be attributed to the better estimation of the word order and term overlap achieved by bigrams. The use of higher order n-grams ($n \geq 3$) did not further significantly improve results.

Comparison of Approaches. Results for our experiments are shown in Table 3 and 4. Numeric indices in the run name indicate the number of documents retrieved per query. The improvement of the LM-ASM approach with bigram indexing over the baseline MT as obtained by brute-force sequential computation, is statistically significant with a reliability of 97%, for the EN-TR dataset, as seen by comparing the BLEU score of LD_{WF} (21.71) with that of LM-ASM-2_{50} (22.08)[7].

Results on the English-Turkish data set are shown in Table 3. We conduct selected experiments on

[6] A '*' denotes a significantly better result of LM or LM-ASM over its counterpart with the same experimental settings.
[7] This improvement is statistically significant as measured by the paired-bootstrap resampling(Koehn, 2004).

Name	Efficiency (runtime)			IR effectiveness						MT
	IT [s]	CT [s]	AT [s]	#ret	#rel_ret	MAP	MRR	|RR=1|	|RR>0|	BLEU
LD_{WF}	0.959	227.709	0.550	1962	1962	1.000	1.000	414	414	21.71
LD_{BR}	0.530	53.380	0.129	1962	1962	1.000	1.000	414	414	21.71
LCS	0.403	52.328	0.126	2143	1679	0.814	0.858	340	378	21.48
TR	0.958	30.816	0.074	1962	1962	1.000	1.000	414	414	21.71
BKT	2.130	13.147	0.032	1301	795	0.536	0.633	262	262	18.25
TST	0.845	12.672	0.031	1962	1962	1.000	1.000	414	414	21.71
tf-idf$_1$	0.696	0.340	0.001	414	87	0.182	0.210	88	88	9.94
LM_1	0.671	0.338	0.001	414	190	0.316	0.459	191	191	17.42
$LM\text{-}ASM_1$	0.617	1.743	0.004	414	239	0.347	0.577	240	240	18.78
$LM\text{-}2_1$	0.914	0.467	0.001	414	232	0.377	0.560	233	233	21.10
$LM\text{-}ASM\text{-}2_1$	0.840	2.460	0.006	414	262	0.367	0.642	266	266	20.29
tf-idf$_{10}$	0.696	0.364	0.001	4140	380	0.312	0.216	43	218	20.29
LM_{10}	0.671	0.596	0.001	4140	788	0.553	0.611	191	368	21.69
$LM\text{-}ASM_{10}$	0.617	1.963	0.005	4140	902	0.603	0.670	240	376	21.09
$LM\text{-}2_{10}$	0.914	1.271	0.003	4140	827	0.619	0.684	233	377	21.15
$LM\text{-}ASM\text{-}2_{10}$	0.840	3.264	0.008	4140	893	0.609	0.730	266	377	21.76
tf-idf$_{50}$	0.663	0.458	0.001	20700	1272	0.514	0.549	164	391	21.40
LM_{50}	0.671	2.195	0.005	20700	1417	0.593	0.617	191	405	21.51
$LM\text{-}ASM_{50}$	0.617	3.562	0.009	20700	1390	0.638	0.701	240	400	21.58
$LM\text{-}2_{50}$	0.914	2.311	0.006	20700	1414	0.658	0.688	233	409	21.39
$LM\text{-}ASM\text{-}2_{50}$	0.840	4.304	0.010	20700	1480	0.638	0.733	266	402	22.08

Table 3: Experimental results on EN-TR data set. Evaluation metrics include indexing time (IT), computation time (CT), and the average time per sentence (AT).

the English-French data set (see Table 4) to investigate scalability and efficiency of the approaches on a larger example base. Additional IR experiments with the BM25 retrieval model (Robertson et al., 1998) were based on indexing all word n-grams up to 5-grams (e.g. unigrams and bigrams for $n = 2$) and retrieving 50 results per query. For the EN-TR data, 1160 relevant results were retrieved when indexing unigrams. For other n-grams, 1348-1356 relevant results are retrieved, thus showing an considerable increase, but not much change for higher values of n. Other performance metrics show similar behaviour: higher performance but little variance for values of $n \geq 2$ compared to unigrams. We briefly revisit the unanswered questions raised in Section 5.

Which approach is the most efficient and scales up to large example bases for EBMT? A sequential approach to compute the edit distance between an input sentence and all source language sentences in the example base is not feasible, because it requires too much run-time (columns CT and AT in Table 3). The sequential brute force approach has the lowest efficiency, but can be improved by either using a metric that can be computed faster, such as LCS, or more efficient data structures such as TR and TST. Interestingly, LCS is a good approximation of the edit distance (as can be seen from the MT scores in the table) and requires a similar run time compared to LD_{BR}. Sequential approaches do not scale up well when using large example bases, even when efficient implementations of the Levenshtein algorithm are used (see Table 4). The use of data structures such as ternary search trees and tries is more efficient, but requires that the structures are kept in memory. Thus, these approaches have higher memory requirements. IR approaches are the most efficient, even

Name	Efficiency (runtime)			IR effectiveness						MT
	IT [s]	CT [s]	AT [s]	#ret	#rel_ret	MAP	MRR	|RR=1|	|RR>0|	BLEU
LD_{WF}	4.653	105813.952	10.581	163751	163751	1.000	1.000	10000	10000	48.42
LD_{BR}	7.355	48695.741	4.870	163751	163751	1.000	1.000	10000	10000	48.42
TR	30.611	27248.317	2.725	163751	163751	1.000	1.000	10000	10000	48.42
TST	8.004	13502.705	1.350	163751	163751	1.000	1.000	10000	10000	48.42
LM_1	8.293	10.546	0.001	100000	3457	0.322	0.410	3654	3654	35.18
LM-ASM_1	8.448	65.604	0.006	100000	3536	0.325	0.381	3804	3804	37.68
LM_{10}	8.293	55.570	0.006	100000	7314	0.436	0.410	3654	5927	41.35
LM-ASM_{10}	8.448	110.808	0.011	100000	7521	0.444	0.418	3804	6168	42.53
LM_{50}	8.293	93.831	0.009	100000	19949	0.431	0.445	3654	7053	44.36
LM-ASM_{50}	8.448	148.889	0.015	100000	26159	0.435	0.465	3804	7247	44.92

Table 4: Experimental results on EN-FR data set.

when a reranking phase based on sequential computation of LS is included.

From the results in Table 4, we observe that when the sequential computation or the efficient data structures are used, the translation quality is high, but at the cost of a runtime of 1.4 seconds in the best case (TST) and 10.6 seconds in the worst case (LD_{WF}). This makes a close-to real-time translation nearly impossible. The use of IR yields a much better runtime, but performance depends on how many documents are retrieved in the initial stage. Note for all of these top-ranked documents, the true edit distance has to be computed to facilitate result reranking. Runtime experiments with the Moses[8] decoder took 0.34s for EN-TR and 1.86s for EN-FR on average per sentence, i.e. a $5\times$ increase in the runtime for a $10\times$ larger dataset. In contrast, our proposed IR approach scales up better (0.010 to 0.015).

Which approach has the highest accuracy when a trade-off between efficiency and effectiveness becomes necessary? The best exact approach to compute the edit distance is the ternary search tree (see columns MRR), which however suffers from higher memory requirements because the data structure has to be kept in memory. As expected, standard IR approaches such as *tf-idf* do not perform well in finding approximate matches, due to missing constraints for the search (e.g. word ordering and word position). Additional reranking of results by the normalized similarity score (based on LD_{WF}) ensures that correct matches are at top ranks (see columns $|\mathrm{RR} = 1|$ and $|\mathrm{RR} > 0|$). The combination of IR followed by a reranking stage is efficient and effective and leads to a high translation quality (see columns AT, MRR, and BLEU in Tables 3 and 4).

Standard LM performs better than *tf-idf*, but does not take into account word positions. Positional ranking in LM-ASM manages to rank the sentences candidates with a high precision and takes into account word positions and word order. It consistently outperforms standard LM in terms of MAP, MRR, and recall (see Table 3), at a moderate cost of additional runtime per query.

Which approach leads to the highest translation quality? Sequential computation of the edit distance based scores yields the highest effectiveness and MT scores, together with all other approaches aiming at finding the exact results with minimum edit distance, i.e. TR and TST.

Surprisingly, we found that the MT scores for our proposed approach with bigram indexing are ac-

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

tually higher than the scores for the sequential computation. To explain why our proposed approach to ASM achieves a higher BLEU score than experiments based on sequential LD computation, we examined some sentences in detail. It can be argued that choosing the minimum edit distance sentence globally from the example base may not necessarily lead to the best translation. For the English input sentence *"where can i buy accessories"* (TR: *nereden aksesuar alabilirim*), the most similar sentence retrieved by sequential LD computation from the full example base is *"where can i buy plates"* (TR: *tabak almak istiyorumi*). Since ties in LD are resolved by taking the sentence with minimum document identifier, the IR method LM-ASM-2_{50} retrieves a different sentence – with the same LS – namely *"where can i buy stockings"* (TR: *nereden çorap satın alabilirim*). The target language sentence parallel to the source sentence candidate found by LD_{WF} compared to *"tabak almak istiyorumi"* does not have a single word common to the reference translation (TR: *nereden aksesuar alabilirim*), whereas the sentence retrieved by LM-ASM-2_{50} has two words in common. The translation template is thus better for the latter which is also reflected in a higher overlap of the final translation output (TR: *nereden aksesuarı satın alabilirim*) with the reference output thus resulting in a higher BLEU score for LM-ASM-2.

How many documents should be retrieved in the IR stage? The results in Table 3 and 4 show that there is a trade-off between speed and quality of translation when using IR: A very fast translation can be achieved by simply retrieving a single sentence, thus completely eliminating the need for a reranking stage, but this requires a high MRR of the initial retrieval. Retrieving more documents results in a higher translation score because for more sentences the best approximate match is found. However this also results in a higher processing time per sentence, because the edit distance has to be computed for more sentences. In practice, this implies that the user of an EBMT system can control whether he is more interested in a high-quality or in a fast translation.

Thus, it proves important to choose source-side sentences with care. Edit distance can be a used as a first filter to choose a set of candidate sentences. A second stage is required to carefully break the ties to ensure selection of the best source side sentence with an equivalent target language sentence *close* to the reference translation.

7 Conclusions

IR can benefit in solving problems beyond the search for information. In this paper, we have described how the adaptation of an information retrieval model to fit specific requirements of approximate retrieval can help to make EBMT more scalable, efficient, and even produce better translations. To solve the problem of approximate sentence matching, we proposed and evaluated LM-ASM, a novel IR model which incorporates three aspects of similarity, namely positional, ordinal, and material similarity. Our approach demonstrates the general usefulness of IR for other tasks. The evaluation covers three aspects: IR metrics, metrics of the problem domain (e.g. BLEU score), and non-functional requirements (e.g. the runtime). We found that our proposed approach for ASM, LM-ASM, outperforms sequential computation of scores and in-memory data structures in terms of runtime, while losing almost none of the translation quality.

As part of future work, we plan to investigate alternatives for generating the relevance assessments based on the BLEU score obtained when using a potentially relevant sentence as translation template. This could lead to similarity metrics and IR models that better approximate the MT quality.

Acknowledgments This research is supported by the Science Foundation Ireland (Grant 07/CE/I1142) as part of the Centre for Next Generation Localisation (CNGL) project. The authors express their gratitude to the anonymous reviewers whose feedback helped improve the quality of this paper.

References

Bentley, J. L. and Sedgewick, R. (1997). Fast algorithms for sorting and searching strings. In *SODA '97: Proceedings of the eighth annual ACM-SIAM symposium on discrete algorithms*, pages 360–369, Philadelphia, PA, USA. SIAM.

Berghel, H. and Roach, D. (1996). An extension of Ukkonen's enhanced dynamic programming ASM algorithm. *ACM Trans. Inf. Syst.*, 14(1):94–106.

Boytsov, L. (2011). Indexing methods for approximate dictionary searching: Comparative analysis. *J. Exp. Algorithmics*, 16:1–91.

Burkhard, W. A. and Keller, R. M. (1973). Some approaches to best-match file searching. *Communications of the ACM (CACM)*, 16(4):230–236.

Dandapat, S., Morrissey, S., Way, A., and van Genabith, J. (2012). Combining EBMT, SMT, TM and IR technologies for quality and scale. In *Proceedings of ESIRMT and HyTra*, pages 48–58, Avignon, France. ACL.

Di Nunzio, G. M., Ferro, N., Mandl, T., and Peters, C. (2008). CLEF 2007: Ad hoc track overview. In *Advances in Multilingual and Multimodal Information Retrieval, 8th Workshop of the Cross-Language Evaluation Forum, CLEF 2007*, volume 5152 of *LNCS*, pages 13–32. Springer.

Faulk, R. D. (1964). An inductive approach to language translation. *CACM*, 7(11):647–653.

Gravano, L., Ipeirotis, P. G., Jagadish, H. V., Koudas, N., Muthukrishnan, S., Pietarinen, L., and Srivastava, D. (2001). Using q-grams in a DBMS for approximate string processing. *IEEE Data Eng. Bull.*, 24(4):28–34.

Gusfield, D. (1997). *Algorithms on Strings, Trees and Sequences: Computer Science and Computational Biology*. Cambridge University Press.

Hiemstra, D. (2000). *Using Language Models for Information Retrieval*. PhD thesis, Center of Telematics and Information Technology, AE Enschede.

Hildebrand, A., Eck, M., Vogel, S., and Waibel, A. (2005). Adaptation of the translation model for statistical machine translation based on information retrieval. In *Proceedings of EAMT*, pages 133–142.

Koehn, P. (2004). Statistical Significance Tests for Machine Translation Evaluation. In *EMNLP 2004*, pages 388—395.

Koehn, P. and Senellart, J. (2010). Fast approximate string matching with suffix arrays and A* parsing. In *Meeting of the Association for Machine Translation of the Americas (AMTA)*.

Levenshtein, V. I. (1966). Binary codes capable of correcting deletions, insertions and reversals. *Soviet Physics Doklady*, 10(8):707–710.

Lin, H.-J., Wu, H.-H., and Wang, C.-W. (2011). Music matching based on rough longest common subsequence. *Journal of information science and engineering*, 27:95–110.

Lv, Y. and Zhai, C. (2009). Positional language models for information retrieval. In *Proceedings of SIGIR '09*, pages 299–306.

Manning, C. D., Raghavan, P., and Schütze, H. (2008). *Introduction to Information Retrieval*. Cambridge University Press.

Mongeau, M. and Sankoff, D. (1990). Comparison of musical sequences. *Computers and the Humanities*, 24:161–175.

Nagao, M. (1984). A framework of a mechanical translation between Japanese and English by analogy principle. In *Artificial and human intelligence*, pages 173–180, New York, NY, USA. Elsevier North-Holland.

Navarro, G. (2001). A guided tour to approximate string matching. *ACM Computing Surveys*, 33(1):31–88.

Needleman, S. B. and Wunsch, C. D. (1970). A general method applicable to the search for similarities in the amino acid sequence of two proteins. *Journal of Molecular Biology*, 48(3):443–453.

Papineni, K., Roukos, S., Ward, T., and Zhu, W.-J. (2002). BLEU: a method for automatic evaluation of machine translation. In *ACL '02*, pages 311–318, Stroudsburg, PA, USA. ACL.

Ponte, J. M. and Croft, W. B. (1998). A language modeling approach to information retrieval. In *SIGIR'98*, pages 275–281. ACM.

Robertson, S. E., Walker, S., and Beaulieu, M. (1998). Okapi at TREC-7: Automatic ad hoc, filtering, VLC and interactive track. In *The Seventh Text REtrieval Conference (TREC-7)*, pages 253–264, Gaithersburg, MD, USA. NIST.

Schulz, K. and Mihov, S. (2002). Fast string correction with Levenshtein-automata. *International journal of document analysis and recognition*, 5:67–85.

Sikes, R. (2007). Fuzzy matching in theory and practice. *Multilingual*, 18(8):39–43.

Somers, H. (2003). *An overview of EBMT*. Kluwer. In Michael Carl and Andy Way (eds) Recent advances in Example-Based Machine Translation, Dordrecht.

Ukkonen, E. (1985a). Algorithms for approximate string matching. *Information and Control*, 64(1-3):100–118.

Ukkonen, E. (1985b). Finding approximate patterns in strings. *J. Algorithms*, 6(1):132–137.

Voorhees, E. M. (1999). The TREC-8 question answering track report. In *Proceedings of TREC-8*, pages 77–82.

Wagner, R. A. and Fischer, M. J. (1974). The string-to-string correction problem. *J. ACM*, 21(1):168–173.

Yap, T. K., Frieder, O., and Martino, R. L. (1996). *High performance computational methods for biological sequence analysis*. Kluwer.

Improving Text Normalization Using Character-blocks based Models and System Combination

ChenLi YangLiu
Department of Computer Science
The University of Texas at Dallas
800 W. Campbell Road, Richardson, Texas
{chenli,yangl}@hlt.utdallas.edu

ABSTRACT
There are many abbreviation and non-standard tokens in SMS and Twitter messages. Normalizing these non-standard tokens will ease natural language processing modules for these domains. Recently, character-level machine translation (MT) and sequence labeling methods have been used for this normalization task, and demonstrated competitive performance. In this paper, we propose an approach to segment words into blocks of characters according to their phonetic symbols, and apply MT and sequence labeling models on such block-level. We also propose to combine these methods, as well as with other existing methods, in order to leverage their different strengths. Our experiments show our proposed approach achieved high precision and broad coverage.

TITLE AND ABSTRACT IN CHINESE

使用字符块模型和系统合成提高文字纠错率

在手机短信和Twitter消息中存在许多缩写和非规范的单词。纠正这些非规范的单词会给后续的自然语言处理模型带来便利。近年来，基于字符层面的机器翻译方法和序列标注方法被广泛应用到这个任务中，并且大大提高了纠错率。本文介绍了一种根据单词发音来切割单词的方法，然后把上述的两种模型利用在切割后的字符块层面进行纠错尝试。另外，我们还尝试了将现有的方法和提出的方法进行各种合成以期利用各自的优点来提高最终的效果。最后我们的实验表明我们的方法在对非规范词的纠错召回率和准确率上都有了明显的提高。

KEYWORDS: text normalization, machine translation, CRF, NLP application.

KEYWORDS IN CHINESE: 文本纠错，机器翻译，条件随机场，自然语言处理应用.

Proceedings of COLING 2012: Technical Papers, pages 1587–1602,
COLING 2012, Mumbai, December 2012.

1 Introduction

There has been a rapid increase in social text in the last few years, including the mobile phone text messages (SMS), comments from the social media websites such as Facebook and Twitter, and real-time communication platforms like MSN and Gtalk. For example, by July 2012[1], Twitter has more than 500 million users, and more than 340 million new tweets are sent everyday. This trend attracts a lot of research in order to mine valuable information and knowledge from this data. Unfortunately, traditional NLP tools sometimes perform poorly when processing this kind of text. One of reasons is that social text is very informal, and contains many misspelled words, abbreviations and many other non-standard tokens. There are different efforts to develop robust language processing techniques for this domain. One is to develop NLP tools that are designed specially for tweets or SMS. For example, the system in (Ritter et al., 2011) reduced the errors of POS tagging by 41% compared with Stanford POS Tagger, and by 22% in parsing tweets compared with OpenNLP chunker tool. Another way is to perform some preprocessing on the original informal text, such as normalization, such that the resulting text is easier for the subsequent modules. The task of normalization is to automatically convert the non-standard tokens into corresponding standard words. Intuitively this will ease subsequent language processing modules in this domain. For example, if '2mr' is converted to 'tomorrow', a text-to-speech system will know how to pronounce it, a POS tagger can label it correctly, and an information extraction system can identify it as a time expression.

Non-standard tokens are formed in different ways in social text, and because of different reasons, including factors such as length limitation, individual's writing style, need to convey emotional or other information. A lot of research has been conducted for text normalization. Some are designed to handle one type of non-standard words, such as deletion (e.g., 'abt' for 'about') (Pennell and Liu, 2010); some can deal with different kinds of non-standard tokens (Liu et al., 2011). Various modeling approaches have been explored for this problem. In particular, a character-level MT method (Pennell and Liu, 2011) and sequence labeling approach (Pennell and Liu, 2010) were used recently and showed good performance. Their essential thought is to make alignment between characters in non-standard tokens and standard words, and predict the probability of the candidate standard words given the non-standard tokens.

In this paper, we propose to use character blocks for alignment, rather than single characters, but adopt the same framework as the above character based methods. Since some characters appear together frequently (e.g., 'ght', 'tion', and 'ure'), we expect that keeping such combinations in the alignment is better than splitting into individual characters in improving system precision (yielding fewer candidates but more accurate). To generate the character blocks, we resort to the word's phonetic symbol and some common pronunciation rules. This results in much better alignment for the MT and sequence labeling methods. In addition, we use two other normalization methods: spell checker (Damerau, 1964) and character-level two-step MT model (Li and Liu, 2012). We further propose to combine these different systems to take advantage of each individual system's strength. For example, the spell checking model is good at correcting simple errors such as 'talkin' (meaning 'talking'), 'freind' (meaning 'friend'), and 'tgether' (meaning 'together'); character-level two-step MT is designed to process the non-standard tokens generated by phoneme similarity like 'ate' (meaning 'eight' in certain context); character blocks level MT and sequence labeling complement in improving the precision on transferring non-standard tokens that have high grapheme similarity with possible standard

[1] http://en.wikipedia.org/wiki/Twitter

words. Results on one public test set show that our proposed individual systems and combined systems perform better than prior work.

2 Related Work

Text normalization has been well studied in text-to-speech field. See (Sproat et al., 2001) for a good report of this problem. Recently, much research has been done on this problem for social text domain, which has many abbreviations or non-standard tokens. A simple approach for normalization would be applying traditional spell checking model, which is usually based on edit distance (Damerau, 1964; Levenshtein, 1966). This method works well for non-standard tokens that are generated by a few simple operations such as substitution, insertion, and deletion of letters of the standard words. However, it cannot handle words such as 'ate' (meaning 'eight'), where non-standard tokens are created based on the phoneme similarity with the standard words, or 'bday' (meaning 'birthday'), which involves too many operations.

Another popular and successful line of work in normalization adopts a noisy channel model. For a non-standard token A, this method finds the most possible standard word S based on Bayes rule: $argmaxP(S|A) = argmaxp(A|S)*p(S)$. Different methods have been used to compute p(A|S). (Pennell and Liu, 2010) used a CRF sequence modeling approach for deletion-based abbreviation. (Liu et al., 2011) further extended this work – they also used CRF framework, but considered more types of nonstandard words without explicit pre-categorization for non-standard tokens. They also used a set of automatically collected training data. (Liu et al., 2012) continued their work by adopting visual prime approach to improve the coverage of candidate list. (Xue et al., 2011) modeled p(A|S) by computing the grapheme and phoneme similarity and then combined those results with context channel and acronym channel.

Some research also utilized the noisy channel model at sentence level. For informal text T and its possible standard form S, using a noisy channel model: $argmaxP(S|T) = argmaxP(T|S)P(S)$, where P(S) is a normal word-based language model, and P(T|S) is an error model. (Choudhury et al., 2007) used hidden Markov model to simulate SMS messages generation, considering the non-standard tokens in input sentence as emission state in HMM and labeling results are possible candidates. (Cook and Stevenson, 2009) extended the work by adding several more subsystems in the error model according to the most common non-standard tokens' formation process. (Wong and Xia, 2008) used the noisy channel model to handle the normalization of Chinese chat language based on a specific property of Chinese language – the similarity of two Chinese characters is measured based on the initial (shengmu) and final (yunmu) pinyin.

Machine translation is another commonly chosen method for text normalization. (Aw et al., 2006) treated SMS as another language, then machine translation techniques are used to translate this 'foreign language' to regular English. (Contractor et al., 2010) used an MT model as well but the focus of his work is to generate an unsupervised method to clean noisy text in this domain. (Pennell and Liu, 2011) firstly introduced an MT method at the character-level for normalization. In fact, this is another way to compute the word-level $p(A|S)$ mentioned before. (Li and Liu, 2012) extended this character-level MT by leveraging phonetic symbol, translating nonstandard words to phones first, and then phones to standard words. They called it character-level two-step MT and showed better overall coverage of candidates than the one-step MT. (Kobus et al., 2008) tackled normalization through an ASR-like system based on the fact that text messages can be represented by phonetic symbols. They decoded SMS message through a nondeterministic phonemic transduction.

For the normalization task for sentences (or messages), a system needs to first identify words that need to be normalized. This is typically done by simply checking whether a word is in a given dictionary. (Han and Baldwin, 2011) conducted a pilot study on determining whether an out-of-vocabulary (OOV) word is a non-standard token that needs normalization or it is just a well formed OOV word. Then for those ill-formed OOV words, they used grapheme and phoneme level similarity to generate candidate words.

Most of the above systems are specially designed to tackle certain aspects of the normalization task. Some focus on improving precision and some aim for a broad coverage. For example, character-level MT from (Pennell and Liu, 2011) achieved top-1 precision of 60.39% on the SMS test set from (Choudhury et al., 2007), and the sequence labeling model from (Liu et al., 2011) improved it to 62.05%, but they have limitations on the candidate coverage (the top-10 precision in (Pennell and Liu, 2011) is around 75%). In addition, (Pennell and Liu, 2010) can only handle the abbreviations. The character-level two-step MT model from (Li and Liu, 2012) improved the top-10 coverage to over 80% on the same data, however, its top-1 precision is not much better than previous work. The visual prime approach from (Liu et al., 2012) improves the overall coverage to around 94%, but the online computation is very time consuming since for a non-standard token T, the system needs to compute the visual prime value between T and all the words S that start with the same character as T. In this paper, we propose an enhanced MT and sequence labeling model, which are built at the character-block level. Furthermore, we combine multiple systems. The performance on a public test set shows our system yields better precision and recall/coverage. Compared to the visual prime approach (Liu et al., 2012), our system also has lower computational cost. To the best of our knowledge, we are the first to use word segmentation to obtain phonetically meaningful character blocks in the normalization task, and also first to combine MT and sequence labeling models.

3 Normalization System

For a non-standard token, we use four different subsystems to normalize it. Each system generates a list of candidate standard words. We then use heuristic rules to re-rank the candidates to form a final candidate list. The four subsystems are developed based on previous work, however, one key contribution of ours in this study is that we propose to segment words into several blocks based on the word's pronunciation and use these blocks as the units in the machine translation or sequence labeling approaches. This helps yield better alignment for model training. In this section, we first describe our four normalization subsystems in details, and then the combination rules.

3.1 Character-Block Level MT

3.1.1 Character-level Machine Translation

This method was first used in (Pennell and Liu, 2011). Similar to machine translation for a word sequence (i.e., a sentence), character-level machine translation translates the character sequence as seen in a non-standard token to another character sequence of a proper word. Formally, for a non-standard token $A = a_1a_2a_3...a_n$, the task is to find the most likely standard word $S = s_1s_2s_3...s_m$, where a_i and s_i are the characters in the words. A statistical machine translation method is used for this task:

$$\hat{S} = argmaxP(S|A) = argmaxP(A|S)P(S) = argmaxP(a_1a_2a_3...a_n|s_1s_2s_3...s_m)P(s_1s_2s_3...s_m) \quad (1)$$

where $P(a_1a_2a_3...a_n|s_1s_2s_3...s_m)$ is from a character-level translation model, and $P(s_1s_2s_3...s_m)$ is from a character-level language model. The translation model is trained using a parallel corpus containing pairs of non-standard tokens and standard words, and the character n-gram language model can be trained using an English dictionary. During testing, the translation module generates hypotheses of character sequences for a given non-standard token. An English dictionary can be used to remove candidates that are not in the dictionary and preserve N-best candidates.

3.1.2 Character-block Level Machine Translation

When using the character-level MT normalization method as described above, we find some character alignments between the non-standard tokens and standard words are not correct. Fig 1(a) shows such an incorrect example. The second 'e' in 'yesterday' is aligned to 's' in the non-standard token, rather than to 'null' (i.e., character deletion). This wrong alignment will affect translation model training and subsequently the decoding process. For this example, if 'er' in 'yesterday' can be viewed as a block or segment, it can be easily aligned to 'r' in the non-standard token 'ystrdy'. This can be achieved using pronunciation information – 'er' together is pronounced as 'ɚ' in the above example. Therefore, we propose to segment an English word according to its phonetic symbol or pronunciation, and then use these segments as units in the MT system. For the example of 'yesterday', when considering its pronunciation, it is segmented as 'y e s t er d ay'.

In our experiment, we collected about 60 most common pairs of character-blocks and their corresponding phonetic symbols, and use them for segmentation. Some example pairs are shown in Table 1. Once we perform this new segmentation, the normalization procedure is similar to the character-based one described earlier, except that now the units in the MT system are characters or character blocks. Fig 1 (b) shows the translation alignment when using the character blocks. During testing, the MT system generates sequences of characters or character blocks. Again, we remove the spaces and only keep a word candidate if it is in the dictionary.

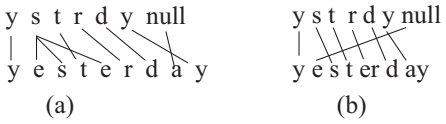


Figure 1: Alignment of 'ystrdy' and 'yesterday' using Character-level MT (a) and Character-block level MT (b)

3.2 Character-Block Level Sequence Labeling

3.2.1 Character-level Sequence Labeling

In (Pennell and Liu, 2010), a CRF sequence modeling approach was used for normalizing deletion-based abbreviation. Given a pair of standard word *W* and its abbreviation *A*, every character in *W* is labeled as 'Y' if it appears in *A*, otherwise 'N'. Fig 2 (a) presents a training instance. A sequence labeling model can be trained using characters as instances, each of which is represented by some features (e.g., character n-grams, position, phonetic information). Next, this model is applied to all the dictionary words. N-best labeling results (i.e., possible abbrevi-

Size of block	Block	Phonetic symbol
2	ea	eɪ,iə,iæ,i
	oo	u,ʊ,ʌ,oʊɑ,oʊə
	ch	k,ʧ
3	ear	ɛər,ɪər,ɜr
	oar	ɔr
	ght	t
4	tion	ʃən
	cial	ʃəl
	sure	ʃər, ʃɜr, ʒər

Table 1: Example pairs of character blocks and corresponding phonetic symbols.

ations) are generated for each word. This forms a lookup table – an abbreviation is associated with all the words that can possibly generate this abbreviation. During testing, for a given abbreviation A, if it is an entry in the lookup table, all of its candidate words will be returned, each with a score $P(A|W) * P(W)$, where $P(A|W)$ is based on the sequence labeling model, and $P(W)$ is from a word-based unigram LM. The best candidate can be obtained from this ranked list. Note if the test abbreviation is not an entry of the lookup table, it means this model fails to normalize this abbreviation token.

(Liu et al., 2011) extended the above model by relaxing the limitation that it only tackles the abbreviation tokens. Rather than labeling 'Y' or 'N' for every character in standard words, they give each character the following 4 types of label: (1) one of the 0-9 digits; (2) one of the 26 characters; (3) null character '_', and (4) letter combination (they used 5 common combinations, such as 'wh', 'ck', and 'ey'). Therefore this system can handle other types of non-standard tokens, such as substitution. An automatic letter level alignment method was used to assign every character in the standard word with a proper label in the non-standard token. Fig 2 (b) presents such a training instance.

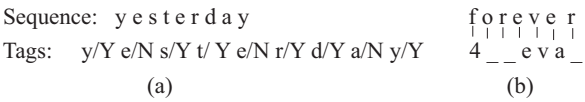


Figure 2: Training examples in sequence labeling models: (a) used in (Pennell and Liu, 2010) and (b) used in (Liu et al., 2011).

3.2.2 Character-Block Level Sequence Labeling

In the above sequence labeling approach, characters are used as instances in the CRF model (although the labels include other symbols or a few letter combination in (Liu et al., 2011)). Similar to the character-based MT method, this can introduce some problems in the alignment. As Fig 2 (b) shows, in the pair of 'forever' and '4eva', letter 'o' and 'r' are aligned to the empty character. Intuitively it is better to align the whole letter chunk 'for' with '4' (actually this is similar to the phrase table in the MT method). Therefore in the sequence labeling model, we

also propose to segment a word into character-blocks rather than single characters, in order to avoid this kind of improper alignments.

Given a training pair (S,T), where T is a non-standard token, and S is the corresponding standard word. First, S is segmented into a sequence of character-blocks and characters, in the same way as in Section 3.1. Then we align it with the characters in non-standard tokens based on the longest common sequence algorithm. At last, we use similar rules as in (Liu et al., 2011) to process the remaining characters between the aligned characters. The only difference is that in non-standard tokens we do not have predefined frequent letter combinations to help decide whether to group a character or character-block with the previous or the following chunk. We put them together with the previous chunk if they have no characters or character-blocks to align in the standard word.

After assigning proper labels for every character and character-block in the standard word, we extract the following features:

- Character-blocks n-gram: c_{-1}, c_0, c_1, $c_{-2}c_{-1}$, $c_{-1}c_0$, c_0c_1, c_1c_2, $c_{-3}c_{-2}c_{-1}$, $c_{-2}c_{-1}c_{-0}$, $c_{-1}c_0c_1$, $c_0c_1c_2$, $c_1c_2c_3$. These are the same as those used (Liu et al., 2011), except that c_i can be character blocks in our work.

- Phonetic features: p_{-1}, p_0, p_1, $p_{-2}p_{-1}$, $p_{-1}p_0$, p_0p_1, p_1p_2, where p_i is the phone corresponding to character or character block c_i. In segmenting words into character-blocks by using phonetic symbols, we notice that for some words the resulting number of blocks is the same as the number of phones. In this case, it is straightforward to align the two sequences (one to one mapping). Otherwise, for the words in the training set, we manually align the two sequences in order to derive the phonetic featuers.

- Syllable features: The website http://www.dict.cn provides syllabification information. We use their syllable segments to assign feature 'B', 'I', and 'L' label for every character-block or character. 'B' means this character or character-block is at the beginning of a syllable, 'I' means inside of a syllable, and 'L' means at the end position of a syllable.

- Word boundary feature: We distinguish the first, last, and inside characters and character-blocks by this feature.

Table 2 shows an example training instance with its features.

Character-block level word	for	e	v	er
Phonetic symbols	fɔr	e	v	ər
Syllable boundary	B	B	I	L
Word boundary	B	I	I	L
Label	4	e	v	a

Table 2: Example of an instance with features in character-block level sequence labeling system.

After training the character-block level CRF model, similar to the previous character-level sequence labeling methods, we apply it to the standard words in the dictionary to create the reverse look-up table, and use it to retrieve candidate words for non-standard tokens during

testing. One thing we need to point it out is that for the words that have different number of character segments and phonetic symbols, we do not include them when building the look up table (since generating high quality phonetic features needs human intervention, which is time consuming).

(Pennell and Liu, 2011),

3.3 Character-level Two-step MT

Li and Liu (2012) introduced a character-level two-step MT method for normalization. Instead of directly translating the character sequence, it first translates the non-standard tokens to possible phonetic sequences, and then to proper words. The purpose of this design is to tackle some difficult cases. For example, for the non-standard word 'rit', it is not easy for the single translation model to convert it to 'right' if there are not enough samples containing 'ght' and 't' in the training pairs. In this case, we can first translate 'rit' to possible phonetic sequences such as 'rait' or 'rIt', and then convert them back to in-vocabulary words. The two steps can both be implemented using machine translation.

In re-implementing this system, we use a different rule to rerank the the candidates compared to that used in (Li and Liu, 2012). For the phonetic sequence generated in the first stage using this method, some can be converted to proper words directly by a simple lookup in the dictionary that consists of word along with their pronunciation (we call these candidates List 1); others have to go through the second-stage translation module to generate possible word candidates (we call these List 2). Our reranking procedure uses a set of heuristic rules to update the probabilities for the candidates in List 2 depending on a candidate's positions in the two lists. For example, one rule is that, if a candidate ranks between 3 and 10 in List 2, and ranks above position 5 in List 1, we change its posterior probability to the same as that of the first candidate's in List 2. Another rule is that, if a candidate ranks between 20 and 40 in List 2 and ranks above position 5 in List 1, we change its posterior probability to the same as that of the 20th candidate in List 2. The essence of this set of rules is to change some candidates' positions to 1, 2, 3, 10 or 20 (we measure the top-N performance for these numbers of candidates). Finally for those candidates with the same posterior probabilities because of the above re-ranking process, we re-rank them again by their longest common sequence with the non-standard tokens.

3.4 Spell Checker

The last normalization subsystem we use is the Jazzy Spell Checker,[2] which is based on edit distance and integrates a phonetic matching algorithm as well.

3.5 System Combination

The above four methods have different strength: Spell checker is very good at correcting simple misspelled words that are formed by a few operations of deletion, insertion or substitution. MT and sequence labeling models have advantages in dealing with more complicated non-standard tokens, regardless of the types. The two-step MT model is designed to tackle the cases when the non-standard tokens have similar pronunciation with the correct answer, but are rather

[2]Jazzy: The java open source spell checker. http://jazzy.sourceforge.net

different on character surface (e.g., 'lrg' and 'large'). In general, the character-block level MT normalization system performs well (as shown later in the experiments).

In order to utilize the advantages of different systems, we propose to combine hypotheses from different systems to obtain the final result. To better evaluate the contributions of these subsystems in the combined system, we examine the combination of different systems:

- Comb1 (two systems): Character-block level MT system + Character-level two-step MT system
- Comb2 (two systems): Character-block level MT system + Character-block level sequence labeling system
- Comb3 (three systems): Character-block level MT system + Character-level two-step MT system + Jazzy Spell Checker
- Comb4 (three systems): Character-block level MT system + Character-block level sequence labeling system + Jazzy Spell Checker
- Comb5 (three systems): Character-block level MT system + Character-level two-step MT system + Character-block level sequence labeling system
- Comb6 (four systems): Character-block level MT system + Character-level two-step MT system + Character-block level sequence labeling system + Jazzy Spell Checker

Since the scores from different systems are in different range and vary greatly, we develop rules to combine them. The combination rules used for these systems are:

- Comb1 and Comb2: These involve two subsystems. The essence of combination rules is to re-rank. The steps we used are:
  (i) create an empty final candidate list;
  (ii) find the common candidates from the two candidate lists (i.e., intersection of the two sets). Since we notice that candidates below position 20 are often not very accurate or not useful for future use (e.g., sentence level normalization/decoding), we decide to obtain the intersection from the top 10 candidates of the two systems' result lists.
  (iii) put elements from the intersection set into the final candidate list, reversely rank them according to the sum of their positions in the two original lists (the smaller, the better). If there is a tie, we re-rank again according to their longest common sequence with the non-standard tokens (the higher, the better).
  (iv) after dealing with the elements in the intersection set, we add the remaining candidates from the two systems into the final list – the first remaining candidate from the character-block level MT system, then the first one from the other system's list, then the second ones from both lists, and so on, until one list is empty and we put all the rest candidates into the final candidate list.
- Comb3 and Comb4: These involve three subsystems and one is Jazzy Spell Checker. We only use the first candidate of the Jazzy spell checker. The combination rule is, if these three subsystems or any of the two subsystems have the same first candidate, select it as the first candidate in the final list, then ignore the Jazzy Spell Checker's result and combine the rest two candidate lists as above. Otherwise, ignore Jazzy Spell Checker's results and just combine the two subsystems as described above for Comb1 and Comb2.

- Comb5: This has three complicated subsystems (not including Jazzy spell checker). The essence of the combination rule is similar to the one used for two subsystems, but here we will consider the intersection of the three systems' top 10 results first, and then three intersection sets from any two subsystems' results. We use similar position information to rank candidates in these common sets. After this step, we first add the remaining candidates from the top 30 results from the sequence labeling system, then the top 30 from character-block lever MT, and at last those from character-level two-step MT.
- Comb6: This includes all the 4 subsystems. If the first candidate of all the systems are the same, or the first candidate of any three sub-systems are the same, choose it as the first candidate in the final list, then ignore the Jazzy Spell Checker's result and combine the rest three candidate lists as above. Otherwise, ignore Jazzy Spell Checker's results and combine the rest three system as Comb5.

Note if any one individual system's result list is empty, the condition is reduced to the combination of fewer individual systems. In addition, no duplicate candidates are included in the final candidate list.

4 Experiments

4.1 Experiment Setup

We use the following data sets. Data set 4 is first used as the training set and the other three as test sets. To evaluate the impact of training size, we use data sets 3 and 4 for training, and evaluate the models on data sets 1 and 2.

- Data 1: 303 pairs of non-standard tokens and standard words collected from SMS in 2007, used in (Choudhury et al., 2007).
- Data 2: 558 pairs of non-standard tokens and standard words collected from 549 tweets in 2010 by (Han and Baldwin, 2011).
- Data 3: 3,998 pairs of non-standard tokens and standard words collected from 6160 tweets between 2009 and 2010 by (Liu et al., 2011). We made another pass of this data set and corrected some annotation.
- Data 4: 2,028 unique pairs of non-standard tokens and standard words, collected from 4,660 Twitter messages, selected from the Edinburgh Twitter Data set collected in Aug 2009 by (Pennell and Liu, 2011).

The dictionary we used is obtained from http://ciba.iciba.com/, which includes 75,262 English word entries and corresponding phonetic symbols (IPA symbols). This is used in various modules in the normalization systems. We segment the English words into character-blocks and use them in both the character-block level MT and character-block sequence labeling model. As described earlier, in order to extract phonetic features for character-block sequence labeling model, we only use the words that have the same number of character-block segments and phones. Therefore, the number of the final standard words used to create the lookup table is 10,105. We choose 10,000-best output in CRF sequence labeling testing. This resulted in about 48,995K non-standard token entries in the lookup table, and the average number of standard words for an entry is about 8. We use CRF++ toolkit for our sequence labeling model.

For the two-step MT system, the training data for the first step is the parallel pairs containing non-standard tokens and their pronunciations (this is the pronunciation for the corresponding normalized words). We obtain this by replacing the standard words in the training set with their phonetic symbols. For the second stage, the training pairs are all the dictionary words and their phonetic symbols.

We train a 5-gram character language model from the Edinburgh Twitter corpus for both character-block level MT model and the second step of the two-step MT method. A phone-based language model is built from the IPAs in the dictionary, and used for the first step translation model in the two-step MT method. SRILM toolkit (Stolcke, 2002) is used to build these language models. We use Moses (Koehn et al., 2007) for all of our experiments. Giza++ (Och and Ney, 2003) is used for automatic word alignment for the three MT systems. We use 8 iterations by IBM model 1, 6 iterations by HMM model, and 6 iterations by IBM model 3 in the alignment. The final score from the translation model is a combination of the four sub-models: phrase translation, language model, reordering model, and the word length penalty weight.

In this study, we only focus on the word-level normalization in all of our experiments. A word-based unigram language model is combined with the normalization score when calculating the final score for a candidate for every individual system except the Jazzy Spell Checker system.

4.2 Experiment Result and Discussion

Tables 3, 4, and 5 present results on test data set 1, 2 and 3, using data set 4 as the training set. Similar to previous work, we evaluate n-best performance (n equals to 1, 3, 10 and 20), and also the candidate coverage (how many test cases' correct answers can be obtained in the final list regardless of its positions). We also include in the table the average number of candidates in the final list to make the coverage result more meaningful. Results in the tables are shown for individual systems and the combined systems, as well as those from previous work.

Among the individual systems, we notice that for data set 1 and 2, character-block level MT system's top one precision outperforms other individual systems, and character-level two-step MT has advantage on the top-n coverage (n from 3) compared with other systems. Character-block level sequence labeling usually has a lower top-1 precision than the two MT models, but its top-20 precision is not much poorer than others (sometimes it is even better). Character-block level sequence labeling can successfully normalize non-standard tokens such as 'gf', 'prof', and 'pro' (to 'girlfriend', 'professor', and 'problem' or 'probably'). However, these cases are very hard for the MT systems to tackle with. MT systems often fail to find correct candidates or sometimes offer no results for these cases. Jazzy spell checker has the worst performance on these two test sets compared to other three systems. In addition, because of its limited number of candidates, its top-n coverage is much lower than others.

The patterns on data set 3 are different. It indicates this data set has different characteristics than the other two. On data set 3, the sequence labeling method performs the best, although its absolute performance is similar to that on the other two data sets. Test data set 3 includes many test cases that the MT system is not good at tackling, such as those examples given above. Because of the poor performance of the MT systems on this data set, there is not much performance difference between them and the Jazzy spell checker.

For all the data sets, we can see that generally the sequence labeling method generates more candidates than others. This is partly because when creating the lookup table, we generated

a large number (10,000) of possible variations for a standard word in the dictionary in order achieve high coverage. Jazzy spell checker has limited number of candidate words, which explains its lower coverage than other methods, even though sometimes it can achieve competitive performance in the top-1 accuracy (e.g., on data set 3). Comparing the character-level MT and character-block level MT, we can see that our proposed method generates fewer number of candidates, but with better accuracy. This will be very beneficial when rescoring these candidates in sentence-level decoding/normalization.

Regarding the combination results, we can see that in general the combined system performs better than the individual systems. Comparing results for comb1 and comb3, and comb2 and comb4, we can see that after adding Jazzy Spell Checker's candidate, the top1 precision has some improvement, but the top-n coverage has no change. This is because we only use the first candidate from the spell checker. It also shows that Jazzy spell checker has some advantages in dealing with some cases and that adding it can yield performance gain. Comparing comb4 and comb6 (the latter includes Jazzy spell checker), we can see that for data set 2, comb4 has better top-1 performance, but not the top-n coverage, whereas on data set 1 and 3, comb6 performs better. We think this might be explained by the relative performance of the character-level two-step MT system. On data set 1 and 3, the character-level two-step MT has better top-n performance (n from 3) than the character-block MT. However, it is different for data set 2. Another important reason for the mixed results for these combined systems is the heuristic rules we use in system combination. We believe better ways of combination can improve performance and make the systems more robust.

Comparing our system performance with other previously reported results, we can see that on data set 1 (which is the most widely used data in previous studies), our system performs the best. On data set 2 and 3, out results are slightly worse than (Liu et al., 2012). One important reason is the training data size – ours is much smaller. However, our system uses fewer candidates and is more efficient than (Liu et al., 2012), which considers all the English words with the same initial character. For the size of the English dictionary we use, which consists of 75,262 words, the average number of candidates would be about 2,894, much higher than the candidates we have in our systems.

For the character-block sequence labeling system, we also performed some manual correction to fix some bad alignment between the character blocks and their phonetic symbol. Results for this modified sequence labeling system and using it in the combination with others are shown at the end of the tables (marked up **). We can see that there is consistent improvement after doing manual correction for the alignment, suggesting the alignment quality is important for model training. This is consistent with the finding that using character blocks outperforms character-based models, since the former has better alignment quality.

Since three of the normalization systems are statistical models, we evaluate the effect of the training data size on the system performance. In this experiment, we also include data set 3 in the labeled training data. We randomly chose some training pairs from it and combine with data set 4, creating training data with 3,000, 4,000 and 5,480 unique pairs respectively (5,480 pairs is the union of data set 3 and 4). Fig 3(a) shows the learning curve evaluated on data set 1 and 2 using these 4 training sizes. This is the result using the combined system 6 involving manual correction. As expected, the performance improves as the training data becomes larger. The pattern suggests that adding more training data may yield further gain. Note that for data set 2, when the training data is 5,480 pairs, it can achieve top one precision of 75.8%, which is

SMS Dataset (303 pairs)	Accuracy					# Cand
	Top1	Top3	Top10	Top20	Coverage	
Character-level MT	65.7	73.7	77	77.3	77.7	12.5
Character-block level MT	67.3	74.7	77.3	78	78	9.2
Character-block level sequence labeling	55	65.3	70.7	72.7	76	46
Character-level two-step MT	64.7	76.3	80.7	81.7	82.4	21.9
Jazzy Spell Checker	43.3	46.7	46.7	46.7	46.7	1.3
Comb6	72.6	83	88.3	90.3	93.7	45.2
Liu et al., 2012	64.36	80.2	89.8	91.8	94.1	n/a
Pennell et al., 2011	60.4	74.6	75.6	75.6	n/a	n/a
Cook et al., 2009	59.4	n/a	83.8	87.8	n/a	n/a
Choundhury et al., 2007	59.9	n/a	84.3	88.7	n/a	n/a
Comb1	68.7	77	82.7	83.3	85	25.7
Comb2	70.7	84	88	91	94	52.7
Comb3	69.7	77.3	82.7	83.3	85	25.7
Comb4	71.3	84	88.3	91	94	52.7
Comb5	73	83	88.3	90.3	93.7	45.2
Character-block level sequence labeling**	58.6	66.7	72.3	74	76.4	59.3
Comb6**	**74.6**	**84.6**	**90.3**	**92**	**94.3**	59.3

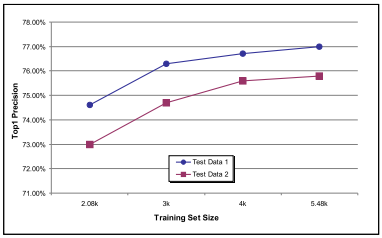
Table 3: Results on Data Set 1. ** means there is manual correction in alignment in the sequence labeling method.

Twitter Dataset (558 pairs)	Accuracy					# Cand
	Top1	Top3	Top10	Top20	Coverage	
Character-level MT	62.2	71.3	75	76.5	77	14.2
Character-block level MT	63.1	72.6	76.2	76.9	77.1	10.7
Character-block level sequence labeling	57.1	67	72	75.6	80.4	62.8
Character-level two-step MT	61.1	71.8	78.3	79.8	81.2	27.5
Jazzy Spell Checker	44.8	50.9	51.4	51.6	51.6	1.5
Comb6	71.7	81.5	86	**89.5**	94.2	38
Liu et al., 2012	**75.69**	n/a	n/a	n/a	n/a	n/a
Han and Baldwin et al., 2011	75.3	n/a	n/a	n/a	n/a	n/a
Comb1	66.2	75	82.4	84.7	86.1	28.9
Comb2	70.6	80.4	84.5	87	92.2	63.9
Comb3	68	75.8	82.4	84.7	86.1	28.9
Comb4	73.1	81.7	84.5	87	92.2	63.9
Comb5	71.1	81.5	86	89.5	94.2	38
Character-block level sequence labeling**	57.4	67	72.5	75.1	80	80.7
Comb6**	73.0	**81.9**	**86.7**	89.2	**94.2**	38.9

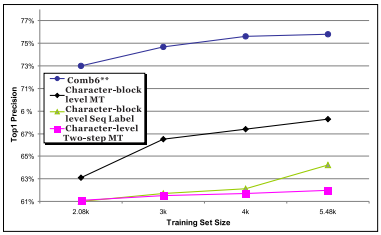
Table 4: Results on Data Set 2. ** means there is manual correction in alignment in the sequence labeling method.

Twitter Dataset (3,998 pairs)	Accuracy					# Cand
	Top1	Top3	Top10	Top20	Coverage	
Character-level MT	49.6	56.8	60.3	63.4	63.7	14.5
Character-block level MT	49.8	57.8	61.2	61.8	62	10.7
Character-block level sequence labeling	54	65.5	74.3	78.3	86.9	52.7
Character-level two-step MT	47.3	60.4	66.3	68.8	69.3	25.8
Jazzy Spell Checker	48.7	54.2	54.7	54.7	54.7	1.6
Comb6	61.4	73.5	82.5	86.3	89.5	40.2
Liu et al., 2012	**69.81**	**82.51**	**92.24**	**93.79**	**95.71**	n/a
Comb1	51	61.7	70	72	73.5	44.3
Comb2	57	70.8	78.9	81.9	86.9	77
Comb3	54	62.2	70	72	73.5	44.3
Comb4	61.4	72	78.9	81.9	86.9	77
Comb5	58.9	73.5	82.5	86.3	89.5	40.2
Character-block level sequence labeling**	58.6	70.2	78.4	81.6	85.5	71.1
Comb6**	62.6	75.1	84	87.5	90.7	45.4

Table 5: Results on Data Set 3. ** means there is manual correction of alignment in the sequence labeling method.



(a) Learning curve of the combination system Comb6** on Data set 1 and 2



(b) Learning curve of individual system and Comb6** on Data set 2

Figure 3: Learning curve for different data sets and differen systems

higher than the state-of-the-art result. We also plot the learning curves for individual systems along with the combined system in Fig 3(b). This is for data set 2. Similar trends are observed for data set 1. It is clear that the combined system outperforms all the individual systems. We can see that the performance of the character-block level MT improves steadily as the training data grows. In contrast, the training size effect on the character-level two-step MT is rather small. This suggests that the two-step MT may not need a large training set, and at the same time this method also has limits and it is hard to make further improvement. Finally for the character-block level sequence labeling, we notice a jump in its performance when training data grows from 4k to 5.48k, which may imply that the added new instances in the last trail are a better match with the test set, and there is still potential for performance gain. More annotated data is needed for a better understanding of the properties of these statistical models.

5 Conclusion and Future Work

In this paper we presented a character-block level MT and sequence labeling method for normalization in social text, and proposed various ways to combine these two novel methods with two other approaches: character-level two-step MT and Jazzy Spell checker. Experiments on several data sets show our methods perform competitively. In particular, on a widely used public test set containing 303 non-standard tokens (Choudhury et al., 2007), our system yields higher 1-best accuracy (74.6%) and better coverage (94.6%) than previous work. Results demonstrate that our proposed character blocks help generate better alignment in the MT and sequence labeling methods, which further improves normalization performance. We believe that our approach is general and applicable to many languages. The idea of aligning standard and non-standard word tokens is not language specific, and we expect it works for other languages, especially for Western European languages that are similar to English in terms of the way of forming non-standard words. Even for languages such as Chinese, there is previous work (Yang et al., 2009) that uses the CRF sequence labeling method for abbreviation detection. In our future work, we plan to first design more robust combination rules to combine different systems (e.g., weighted combination or reranking). Second we will perform sentence level normalization where word n-gram language models will be incorporated to re-rank the candidates.

Acknowledgments

This work is partly supported by DARPA under Contract No. HR0011-12-C-0016. Any opinions expressed in this material are those of the authors and do not necessarily reflect the views of DARPA.

References

Aw, A., Zhang, M., Xiao, J., Su, J., and Su, J. (2006). A phrase-based statistical model for sms text normalization. In *Processing of COLING/ACL*, pages 33–40.

Choudhury, M., Saraf, R., Jain, V., Mukherjee, A., Sarkar, S., and Basu, A. (2007). Investigation and modeling of the structure of texting language. *IJDAR*, 10(3-4):157–174.

Contractor, D., Faruquie, T. A., Subramaniam, L. V., and Subramaniam, L. V. (2010). Unsupervised cleansing of noisy text. In *Proceedings of the 23rd International Conference on Computational Linguistics: Posters, Association for Computational Linguistics*, pages 189–196, Stroudsburg, PA, USA.

Cook, P. and Stevenson, S. (2009). An unsupervised model for text message normalization. In *Proceedings of the Workshop on Computational Approaches to Linguistic Creativity*, pages 71–78, Boulder, Colorado. Association for Computational Linguistics.

Damerau, F. (1964). A technique for computer detection and correction of spelling errors. *Commun. ACM*, 7(3):171–176.

Han, B. and Baldwin, T. (2011). Lexical normalisation of short text messages: Makn sens a #twitter. In *Proceeding of 49th ACL*, pages 368–378, Portland, Oregon, USA.

Kobus, C., Yvon, F., and Damnati, G. (2008). Normalizing sms: are two metaphors better than one. In *Proceedings of 22nd International Conference on Computational Linguistics*, pages 441–448.

Koehn, P., Hoang, H., Birch, A., Callison-Burch, C., Federico, M., Bertoldi, N., Cowan, B., Shen, W., Moran, C., Zens, R., Dyer, C., Bojar, O., Constantin, A., Herbst, E., and Herbst, E. (2007). Moses: Open source toolkit for statistical machine translation. In *Proceedings of the 45th Annual Meeting of the ACL on Interactive Poster and Demonstrartion Sessions*, pages 177–180.

Levenshtein, V. (1966). Binary codes capable of correcting deletions, insertions and reversals. *Soviet Physics Doklady*, 10:707.

Li, C. and Liu, Y. (2012). Normalization of text messages using character- and phone-based machine translation approaches. In *Proceedings of 13th Interspeech*.

Liu, F., Weng, F., and Jiang, X. (2012). A broad-coverage normalization system for social media language. In *Proceedings of the 50th ACL*, pages 1035–1044, Jeju Island, Korea.

Liu, F., Weng, F., Wang, B., and Liu, Y. (2011). Insertion, deletion, or substitution? normalizing text messages without pre-categorization nor supervision. In *Proceedings of the 49th ACL*, pages 71–76, Portland, Oregon, USA.

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

Pennell, D. and Liu, Y. (2010). Normalization of text messages for text-to-speech. In *ICASSP*, pages 4842–4845.

Pennell, D. and Liu, Y. (2011). A character-level machine translation approach for normalization of sms abbreviations. In *Proceedings of 5th International Joint Conference on Natural Language Processing*, pages 974–982, Chiang Mai, Thailand.

Ritter, A., Clark, S., Mausam, and Etzioni, O. (2011). Named entity recognition in tweets: An experimental study. In *Proceedings of 2011 EMNLP*, pages 1524–1534.

Sproat, R., Black, A. W., Chen, S. F., Kumar, S., Ostendorf, M., and Richards, C. (2001). Normalization of non-standard words. *Computer Speech & Language*, 15(3):287–333.

Stolcke, A. (2002). SRILM-an extensible language modeling toolkit. In *Proceedings International Conference on Spoken Language Processing*, pages 257–286.

Toutanova, K. and Moore, R. (2002). Pronunciation modeling for improved spelling correction. In *Proceedings of 40th ACL*, pages 144–151, Philadelphia, Pennsylvania, USA.

Wong, K.-F. and Xia, Y. (2008). Normalization of chinese chat language. *Language Resources and Evaluation*, 42(2):219–242.

Xue, Z., Yin, D., Davison, B. D., and Davison, B. D. (2011). Normalizing microtext. In *Proceedings of 25th AAAI*.

Yang, D., Pan, Y.-C., and Furui, S. (2009). Automatic chinese abbreviation generation using conditional random field. In *HLT-NAACL (Short Papers)*.

Update Summarization Using a Multi-level Hierarchical Dirichlet Process Model

Jiwei Li[1] *Sujian Li*[1] *Xun Wang*[1] *Ye Tian*[1] *Baobao Chang*[1]
(1) Key Laboratory of Computational Linguistics, Peking University, Ministry of Education, CHINA
{bdlijiwei, lisujian, xunwang, ytian, chbb}@pku.edu.cn

ABSTRACT

Update summarization is a new challenge which combines salience ranking with novelty detection. Previous researches usually convert novelty detection to the problem of redundancy removal or salience re-ranking, and seldom explore the birth, splitting, merging and death of aspects for a given topic. In this paper, we borrow the idea of evolutionary clustering and propose a three-level HDP model named h-uHDP, which reveals the diversity and commonality between aspects discovered from two different epochs (i.e. epoch history and epoch update). Specifically, we strengthen modeling the sentence level in the h-uHDP model to adapt to the sentence extraction based framework. Automatic and manual evaluations on TAC data demonstrate the effectiveness of our update summarization algorithm, especially from the novelty criterion.

KEYWORDS : Update summarization, Hierarchical Dirichlet process, Novelty detection.

Proceedings of COLING 2012: Technical Papers, pages 1603–1618,
COLING 2012, Mumbai, December 2012.

1 Introduction

Update summarization aims to generate a short and concise summary for the latest updating topic-related documents (hereafter *update documents* for short), under the assumption that the user has already read the earlier historical documents (*history documents* for short) about the same topic. Recently, there have been many attempts to explore different approaches to generate update summaries. The predominant approaches are mainly built upon the sentence extraction framework.

Update summarization for an evolving topic differs from previous generic summarization for a static topic in that the latter aims to acquire the salient information in one topic, while the former cares for both the salience and the novelty of information. By developing traditional summarization techniques, massive efforts on update summarization have been made to dig out new information (Boudin et al., 2008; Fisher and Boark, 2008; Wan, 2007; Li et al., 2008; Du et al., 2010; Li et al., 2012). The typical examples include the scaled Maximal Marginal Relevance (MMR) algorithm which excludes those sentences similar to the history documents, and some extensions of TextRank such as TimedTextRank (Wan, 2007), PNR^2 (Li et al., 2008), MRSP (Du et al., 2010) which re-rank the salience scores of sentences by employing various kinds of reinforcement between sentences. One problem with these approaches is that they tend to regard update summarization more as a redundancy removal problem than a novelty detection problem. Another problem is that these approaches are mainly based on the computation of lexical similarities between sentences and fail to consider higher level information to avoid semantic redundancy in update summarization.

To solve these two problems, we borrow the techniques of evolutionary clustering which focuses on detecting the dynamics of a given topic. Normally, one topic is described from various specific aspects [1], accompanied with the background information running the whole topic (Chemudugunta et al., 2007; Li et al., 2010). For example, the topic "Quebec independence" may involve the specific aspects including "leader in independence movement", "referendum", "related efforts in independence movement" and so on, while "Quebec" and "independence" are seen as the general background information. The evolving dynamics of a topic is mainly embodied in the birth, splitting, merging and death of the specific aspects (Ren et al., 2008). Then, the commonality and diversity between history documents and update documents can be easily summarized from the aspect level and update summarization is not limited to lexical redundancy removal. Recently, hierarchical Dirichlet process (HDP) (Teh et al., 2006) has been widely used to model the aspects in evolutionary clustering. HDP does not need to predefine the number of clusters, and can be easily and naturally extended to multiple correlated corpora for detecting aspects (Ren et al., 2008; Xu et al., 2008; Zhang et al., 2010; Gao et al., 2011). These distinct advantages make it suitable to update summarization. However, to our best knowledge, no previous work has explored HDP for update summarization.

Aiming at the task of update summarization, in this paper, we develop a novel three-level (i.e. corpus, document set, and document levels) HDP model, called h-uHDP model, which extends the standard HDP to the scenario of two related document sets in different epochs (namely history epoch and update epoch). In h-uHDP, the diversity and connections of aspects between two epochs are naturally modeled: two epochs may share some common aspects; further, some aspects may become outdated while some become popular or some new may appear over time, causing the number of aspects and aspect structures to change at different epochs.

[1] Aspect in this article is usually called *cluster* in evolutionary clustering.

Under the framework of extractive summarization, it is important to acquire the relationship between sentences and aspects for sentence selection. However, in most existing HDP models, the sentence level is disregarded and we cannot directly get the aspect distribution of sentences. Inspired by the progress made in Latent Dirichlet Allocation (LDA) models (Chemudugunta et al., 2007; Li et al., 2010; Delort and Alfonseca, 2012), we newly add the sentence level between the word level and document level in the h-uHDP model. Since neighboring sentences in one document usually talk about one same aspect, we assume that the aspect assignment of each sentence is not conditionally independently. With such assumption, the aspect of each sentence is determined by the aspect distribution of both the document and its neighboring sentences. Our h-uHDP model is capable of mapping multiple levels of information into the latent aspect space.

The rest of this paper is organized as follows. Section 2 discusses the related work on update summarization and evolutionary clustering. Section 3 briefly introduces Dirichlet Process (DP) and Hierarchical Dirichlet Process (HDP). Section 4 presents our proposed aspect model h-uHDP and its inference algorithm. Section 5 addresses the algorithm of update summarization. Section 6 shows the experimental results. Finally, Section 7 concludes the paper.

2 Related work

In this section, we review the related work on update summarization and evolutionary clustering.

2.1 Update summarization

In generic summarization[2], numerous techniques have been developed to measure the salience of sentences and remove the redundancy in summaries, such as the well-known Maximal Marginal Relevance (MMR) (Carbonell and Goldstein, 1998), TextRank (Mihalcea and Tarau, 2004) et al. Some initial work on update summarization inherited the idea of salience ranking in generic summarization and extended the available algorithms to selecting sentences from the newly-coming documents. Boudin et al. (2008) proposed a sentence scoring algorithm derived from MMR and preferred to select those sentences dissimilar to previously read sentences. Fisher and Roark (2008) used a supervised perceptron and simple filtering rules to get the salient sentences for the update documents. Gillick et al. (2008) formulated sentence selection as the problem of integer linear programming and aimed to select a set of sentences that maximize the sum of weights of n-grams covered by the sentences. Adapting the ILP of Gillick et al.(2008), CLASSY by Conroy et al.(2009) seeked to find the sentences that maximize the total approximate oracle scores. Wang and Li (2010) employed an incremental hierarchical clustering algorithm COBWEB to re-organize sentence clusters immediately after new documents/sentences arrive and the most representative sentences for the updated clusters were selected. The graph-based algorithm – TextRank (Mihalcea and Tarau, 2004) has more extensions for update summarization. TimedTextRank by Wan (2007) introduced the time decaying ratio for weighting sentence reinforcement, PNR^2 by Li et al. (2008) added the negative reinforcement between sentences, and MRSP by Du et al. (2010) turned the historical sentences into sink points which are limited their reinforcement with other sentences. Through reinforcement propagation, the salience of sentences in the update documents is influenced by history documents to assure that those sentences with less redundancy with history documents appear in the update summaries. However, they mainly start from the lexical level and cannot explain explicitly what the novel information is.

There are also a few attempts to explore semantic information in update summarization. Steinberger and Jezek (2009) proposed the Iterative Residual Rescaling (IRR) algorithm which

[2] In this paper, generic summarization refers to the non-update summarization.

maps the documents to a set of latent semantic aspects[3]. Then sentences containing novel and significant aspects are then selected for the summary. Inspired by Latent Dirichlet Allocation (LDA) (Blei et al., 2003), Delort and Alfonseca (2012) proposed the DualSum algorithm which designs a nonparametric Bayesian approach to generate four kinds of aspects respectively for background, document, common and novel information. Though these researches have achieved some preliminary findings on exploring semantic information in update summarization, they still cannot present a unified framework to reveal the dynamics of a given topic.

2.2 Evolutional clustering and HDP

Evolutionary clustering is a relatively new research for topic detection, which aims to preserve the smoothness of clustering results over time, while fitting the data of each epoch. The work by Chakrabarti et al. (2006) was probably considered as the first to address the problem of evolutionary clustering. They proposed a general framework of evolutionary clustering and extended two classical clustering algorithms to the evolutionary setting: (1) k-means clustering, and (2) agglomerative hierarchical clustering. Later, Chi et al. (2008) presented two frameworks by incorporating temporal smoothness constraint and applied them on spectral clustering algorithm. While the researches on extending classic clustering algorithms have advanced the literature of evolutionary clustering, they have a very restrictive assumption: the number of clusters over time stays the same. It is clear that this assumption is obviously violated in many real applications.

Recently, HDP has been widely used in evolutionary clustering due to its capability of learning number of clusters automatically and sharing mixture components across different corpora. In HDP, each corpus is modeled by an infinite Dirichlet Process (DP) mixture model, and the infinite set of mixture clusters is shared among all corpora. Sethuraman (1994) gave a stick-breaking constructive definition of DP for arbitrarily measurable base space, which is very useful to model the weight of mixture components in the mixture model. Blackwell and MacQueen (1973) explained DP using the Polya urn scheme, as the predictive distribution of an event is proportional to the frequency of the existing events or to a concentration parameter for an unpresented event. The Polya urn scheme is closely related to the Chinese Restaurant Process (CRP) metaphor, which is applied on HDP demonstrating the 'clustering property' as the 'distribution on partition'. In addition, HDP can also be seen as an LDA-based model, which can automatically and naturally infer the number of clusters from data (Teh et al., 2006). Base on HDP, some algorithms of evolutionary clustering are proposed by incorporating time dependencies, such as DPChain, HDP-EVO, HDP-HMM, dynamic HDP and EvoHDP et al. (Xu et al., 2008; Xu et al., 2008; Ren et al., 2008; Zhang et al., 2010; Gao et al., 2011).

3 DP and HDP

In this section, we briefly introduce Dirichlet Process (DP) and Hierarchical Dirichlet Process (HDP).

A DP can be considered as a distribution of probability measure G. Suppose a finite partition $(T_1,\ldots,T_K)$ in the measure space Θ and a probability distribution G_0 on Θ, we write $G \sim DP(\alpha, G_0)$ if $(G(T_1),\ldots,G(T_K)) \sim Dir(\alpha G_0(T_1),\ldots,\alpha G_0(T_K))$, where α is a positive concentration parameter and G_0 is called a base measure. Sethuraman (1994) showed that a measure G drawn from a DP is discrete by the stick-breaking construction:

[3] aspect is called as topic in the original paper of (Steinberger and Jezek, 2009).

$$\beta_k \sim Beta(1,\alpha),\quad \pi_k = \beta_k \prod_{l=1}^{k-1}(1-\beta_l) = \beta_k(1-\sum_{l=1}^{k-1}\pi_l),\quad \{\phi_k\}_{k=1}^{\infty} \sim G_0,\quad G = \sum_{k=1}^{\infty}\pi_k \delta_{\phi_k} \tag{1}$$

where δ_{ϕ_k} is a probability measure concentrated at ϕ_k. It is important to note that the sequence $\boldsymbol{\pi} = (\pi_k)_{k=1}^{\infty}$ constructed by Eq. (1) satisfies $\sum_{k=1}^{\infty}\pi_k = 1$ with probability 1. For convenience, we write $\boldsymbol{\pi}$~GEM(α)[4] if $\boldsymbol{\pi}$ is a random probability measure defined by Eq. (1). After observing the draws $\theta_1, \theta_2, ..., \theta_{n-1}$ from *G*, the posterior of *G* still satisfies the *DP* distribution:

$$G \mid \theta_1, \theta_2, ..., \theta_{n-1} \sim DP(\alpha + n - 1, \frac{m_k \delta_{\phi_k} + \alpha G_0}{\alpha + n - 1}) \tag{2}$$

where m_k denotes the number of draws taking the value ϕ_k.

A HDP defines a distribution over a set of DPs. In HDP, a global measure G_0 is distributed as a DP with concentration parameter γ and base measure *H*. Then a set of measures $\{G_j\}_{j=1}^{J}$ is drawn independently from a DP with base measure G_0. Such a process is described as:

$$G_0 \sim DP(\gamma, H), \qquad G_j \mid \alpha_0, G_0 \sim DP(\alpha_0, G_0) \tag{3}$$

For each *j*, let $\{\theta_{ji}\}_{i=1}^{n_j}$ be independent and identically distributed (i.i.d.) random variables drawn from G_j. n_j observations $\{x_{ji}\}_{i=1}^{n_j}$ are drawn from the mixture model:

$$\theta_{ji} \sim G_j, \qquad x_{ji} \sim F(x \mid \theta_{ji}) \tag{4}$$

where $F(x|\theta_{ji})$ denotes the distribution of generating x_{ji}. Equations (3) and (4) complete the definition of a HDP mixture model, whose graphical representation is shown in Figure 1(a).

According to Eq. (1), the stick-breaking construction of HDP can be represented as:

$$\boldsymbol{\beta} = (\beta_k)_{k=1}^{\infty} \sim GEM(\gamma),\quad G_0 = \sum_{k=1}^{\infty}\beta_k \delta_{\phi_k},\quad \boldsymbol{\pi_j} = (\pi_{jk})_{k=1}^{\infty} \sim \mathrm{DP}(\alpha_0, \boldsymbol{\beta}),\quad G_j = \sum_{k=1}^{\infty}\pi_{jk}\delta_{\phi_k} \tag{5}$$

and the corresponding graphical model is shown in Fig. 1(b). We can see that HDP can readily be extended to as many levels as are deemed useful. That is, we can obtain a hierarchy of DPs, where the draw from the DP at a given node serves as a base measure for its children (Teh, 2006).

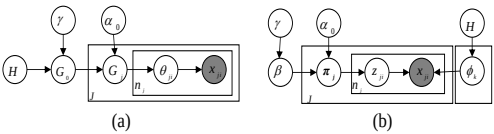


FIGURE 1 – Graphical representation for HDP. (a) original representation. (b) stick-breaking construction

4 h-uHDP model

This section clarifies why and how we propose our improved HDP model (named history-update HDP, *h-uHDP* for short) for the task of update summarization.

In update summarization, a given topic is composed of two document sets (*docset* for short) varying two epochs, namely *history* and *update* epoch. To precisely observe the dynamics of the aspects in one topic, we need to model the aspects over three levels: topic corpus, *docset* at each

[4] GEM stands for Griffiths, Engen, and McCloskey (Teh et al. 2006)

epoch, and document. In such case, we extend the standard HDP to a three level HDP: a set of common aspects on the top level of the hierarchy explicitly address the issue of aspect correspondence between two epochs; the second level is for the aspects at each different epoch, which are considered as the subset of the top level aspects; and the third level is designed for the aspects of each document; the relationship among these three levels of aspects can be obtained through statistical inference. Thus, h-uHDP can naturally model the diversity and connections of aspects between two epochs.

First of all, we introduce some notations in our real data setting of update summarization. We use J^H and J^U to denote the number of documents in the *history* and *update* epochs respectively. For the convenience of description, we use the symbol e in the superscript to denote U or H. Each docset is denoted as $D^e = \{D_j^e\}_{j=1}^{J^e}$ where document D_j^e has N_j^e sentences $\{s_{j,i}^e\}_{i=1}^{N_j^e}$ and the i^{th} sentence in D_j^e has $N_{j,i}^e$ observed word samples $\{x_{j,i,n}^e\}_{n=1}^{N_{j,i}^e}$.

4.1 Model

Our h-uHDP model is an extension of a three-level HDP model which naturally incorporates the levels of corpus, docset and document as shown in Fig. 2. Specifically, we design a two-level HDP respectively for each docset, and these two HDPs share an overall base measure G which is drawn from $DP(\varepsilon,G_0)$ and serves as the overall component bookkeeping for both epochs. We use G^H to denote the global measure for the *history* epoch and call it the *history* global measure. Similarly, G^U is called the *update* global measure. Then, the local measures for each document are denoted as $\{G_j^e\}_{j=1}^{J^e}$, which are drawn from the *history* or *update* global measures. That is, $G_j^e \sim DP(\alpha^e, G^e)$ given $e \in \{H, U\}$.

Then, we introduce the sentence level into the HDP model where each sentence is assigned to one aspect with the consideration of both its neighboring sentences and words contained by this sentence. We use $\theta_{j,i}^e$ to denote the aspect assignment of the i^{th} sentence in D_j^e. There is also a binomial distribution $y \sim binomial(\rho)$, which controls for each sentence how often we encounter a background word, or an aspect word. ρ has a beta prior with parameter β: $\rho \sim beta(\beta)$.

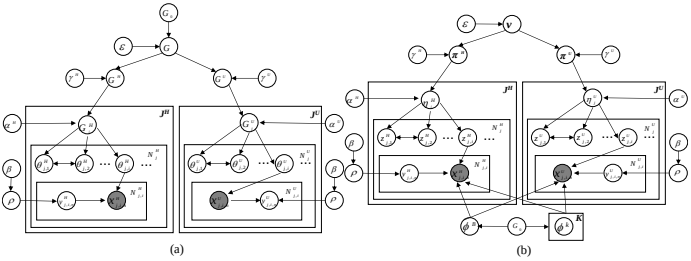


FIGURE 2 –Graphical representation for h-uHDP. (a) original representation. (b) stick-breaking construction

Fig. 2(a) illustrates the graphical representation of h-uHDP model. The generation process of our h-uHDP model is as follows:

1. Draw an overall base measure $G \sim DP(\varepsilon, G_0)$, which denotes the overall aspect distribution for all documents at two epochs.

2. For $e \in \{H, U\}$:

   2.1 Draw the global measure G^e according to the overall measure G. G^e serves as the base measure for each docset. That is, $G^e \sim DP(\gamma^e, G)$.

   2.2 Draw the local measures $\{G_j^e\}_{j=1}^{J^e}$. Each G_j^e for document D_j^e is drawn from the corresponding global measure G^e: $G_j^e \sim DP(\alpha^e, G^e)$.

   2.3 Draw the aspect for sentence $s_{j,i}^e$ according to G_j^e and the aspect assignment of neighboring sentences: $\theta_{j,i}^e \sim g(\theta \mid G_j^e, \boldsymbol{\theta}_{j,-i}^e)$

   2.4 Sample the words $\{x_{j,i,n}^e\}_{n=1}^{N_{j,i}^e}$: $y_{j,i,n}^e \sim binomial(\rho)$, $x_{j,i,n}^e \sim f(x \mid \theta_{j,i}^e, y_{j,i,n}^e)$ where $f(x \mid \theta_{j,i}^e, y_{j,i,n}^e)$ is a distribution parameterized by $\theta_{j,i}^e$ and $y_{j,i,n}^e$.

We can see that the extended three-level HDP model h-uHDP, in fact, considers five levels for a given topic, including word, sentence, document, docset, and corpus. At the same time, aspect assignment dependency between sentences is naturally incorporated in the model.

Next, we will provide the stick-breaking perspective and a Gibbs sampler for model inference.

4.2 The stick-breaking construction

According to the stick-breaking construction of DP, the overall base measure G can be expressed with the following form:

$$G = \sum_{k=1}^{\infty} v_k \delta_{\phi_k}, \quad \boldsymbol{v} \sim GEM(\varepsilon) \tag{6}$$

Then, according to Eq. (5), we can also get the global and local measures with the form as:

$$G^e = \sum_{k=1}^{\infty} \pi_k^e \delta_{\phi_k}, \quad \boldsymbol{\pi}^e \sim DP(\gamma^e, v) \tag{7}$$

and

$$G_j^e = \sum_{k=1}^{\infty} \eta_{jk}^e \delta_{\phi_k}, \ \boldsymbol{\eta}_j^e \sim DP(\alpha^e, \boldsymbol{\pi}^e) \tag{8}$$

As with the standard HDP, we get the stick-breaking construction for h-uHDP, illustrated in Figure 2(b).

Next, we focus on the modeling of the sentence level. $z_{j,i}^e$ is used to indicate the aspect assignment of sentence $s_{j,i}^e$ and the formula of assigning aspect k to $z_{j,i}^e$ is as follows:

$$p(z_{j,i}^e = k \mid \boldsymbol{\phi}, \boldsymbol{z}_{-j,i}^e) \propto f_k(s_{j,i}^e) g(z_{j,i}^e = k \mid \boldsymbol{z}_{-j,i}^e) \tag{9}$$

$$\text{and} \quad g(z_{j,i}^e = k \mid \boldsymbol{z}_{-j,i}^e) = \frac{1}{|N_j^e - 1|} \sum_{d \in [1, N_j^e] - \{i\}} \exp\left\{ \frac{\sigma \cdot \delta(k, z_{j,d}^e)}{|d - i|} \right\} \tag{10}$$

In Formula (9), $f_k(s_{j,i}^e)$ denotes the probability of generating sentence $s_{j,i}^e$ given aspect k and the function $g(\bullet)$ reflects the influence from the neighboring sentences. We use the symbol '-' to denote the exclusion of current sentence or word, and $\boldsymbol{z}_{-j,i}$ means the aspect assignment of all sentences in D_j^e excluding the current sentence. $\delta(k, z_{j,d}^e)$ equals 1 if the aspect assignment of the d^{th} sentence in D_j^e is k, 0 otherwise. The parameter σ (named as *sentence influence factor*) is used to tune the influence from neighboring sentences. Eq. (10) shows that the longer the distance from one sentence to the current sentence, the less the influence that sentence has on the aspect assignment of the current sentence. $y_{j,i,n}^e$ is the indicator variable of word $x_{j,i,n}^e$ and controlled by a binomial distribution with beta prior β. If $y_{j,i,n}^e = 0$, $x_{j,i,n}^e$ is a background word.

If $y^e_{j,i,n} = 1$, $x^e_{j,i,n}$ is an aspect word.

4.3 Inference

For model inference, we use a straightforward Gibbs sampler based on the Chinese Restaurant Franchise (CRF) and the stick-breaking construction. Thus, we begin with an analog of the CRF process for h-uHDP: a document D^e_j corresponds to a *restaurant*, and a sentence $s^e_{j,i}$ corresponds to a customer. Different from the standard HDP, one customer in our model is seen as a family which includes a few persons. Here, we assume that the persons in one family usually have the same preference for one dish at one table except some persons shown no preference for any food. The general *background dish* is assigned to the persons having no preference and a specific dish k is assigned to those persons having preference. The global menu of dishes is denoted by K+1 i.i.d. random variables $\phi_B, \phi_1, \ldots, \phi_K$ distributed according to G_0. We also introduce variables, ψ^e_{jt}, to represent the dish served at table t in restaurant j. To denote the associations among $\theta^e_{j,i}$, ψ^e_{jt} and ϕ_k, we let t^e_{ji} be the index of ψ^e_{jt} associated with $\theta^e_{j,i}$, and let k^e_{jt} be the index of ϕ_k associated with ψ^e_{jt}. In the CRF metaphor, customer $s^e_{j,i}$ sits at table t^e_{ji} while table t in restaurant D^e_j serves dish k^e_{jt}.

We also need a notation for counts. Specifically, we need to record the counts of families, persons and tables. Marginal counts are represented with dots in the subscript. $n^e_{jt\bullet}$ represents the number of families in restaurant j at table t in the corresponding epoch, and $n^e_{j\bullet k}$ represents the number of families in restaurant j eating dish k in the corresponding epoch. The notation m^e_{jk} denotes the number of tables in restaurant j serving dish k in one epoch, $m^e_{j\bullet}$ denotes the number of tables in restaurant j in one epoch, $m^e_{\bullet k}$ denotes the number of tables serving dish k in one epoch, and $m^e_{\bullet\bullet}$ denotes the total number of tables in each epoch. The notation above with removing the superscript represents the corresponding counts in both epochs. For example, $n_{\bullet\bullet k}$ is the total number of customers assigned to aspect k in both epochs and $m_{\bullet k}$ denotes the total number of tables serving dish k in both epochs.

In our implementation, we first sample the index variables t^e_{ji} and k^e_{jt}. Then the $\theta^e_{j,i}$ and ψ^e_{jt} can be reconstructed from their index variables and ϕ_k, which makes the MCMC sampling more efficient (Blei et al., 2006).

Sampling ***t****.* Due to the space limit, we would just show the sampling formula without derivation. The likelihood due to $s^e_{j,i}$ given $t^e_{ji} = t$ for some previously t is $f^{-s^e_{j,i}}_{k^e_{jt}}(s^e_{j,i})$. The likelihood of generating $s^e_{j,i}$ given $t^e_{ji} = t^{new}$ can be calculated by integrating out the possible values of $k_{jt^{new}}$. K^e denotes the set of aspects assigned in current epoch. The prior probability that t^e_{ji} takes on a previously used t is calculated according to $g(z^e_{j,i} \mid \mathbf{z}^e_{\mathbf{-j,i}})$, $f^{-s_{ji}}_{k_{jt}}(s^e_{j,i})$ and $n^e_{jt\bullet}$, whereas the probability that it takes on a new value (i.e. $t^{new} = m^e_{j\bullet} + 1$) is proportional to α^e. Then, the conditional distribution of t^e_{ji} given the rest of the variables is:

$$p(t^e_{ji} = t \mid \mathbf{t}_{-ji}, \mathbf{k}) \propto \begin{cases} g(z^e_{j,i} = k \mid \mathbf{z}^e_{\mathbf{-j,i}}) n^e_{jt\bullet} f^{-s_{ji}}_{k_{jt}}(s^e_{j,i}) & \textbf{\textit{if } } t \textbf{\textit{ previously used}} \\ \alpha^e P(s^e_{j,i} \mid \mathbf{t}_{-ji}, t^e_{ji} = t^{new}, \mathbf{k}) & \textbf{\textit{if }} t = t^{new} \end{cases} \tag{11}$$

and
$$P(s^e_{j,i} \mid \boldsymbol{t}_{-ji}, t_{ji} = t^{new}, \boldsymbol{k}) = \sum_{k \in K^e} g(z^e_{j,i} = k \mid \boldsymbol{z}^e_{-j,i})\left(\frac{m^e_{\bullet k}}{m^e_{\bullet\bullet} + \gamma^e} + \frac{\gamma^e}{m^e_{\bullet\bullet} + \gamma^e} \cdot \frac{m_{\bullet k}}{m_{\bullet\bullet} + \varepsilon}\right) f_k^{-s^e_{j,i}}(s^e_{j,i}) + \sum_{k \in K^{\{H,U\}} - K^e} \frac{\gamma^e}{m^e_{\bullet\bullet} + \gamma^e} \cdot \frac{m_{\bullet k}}{m_{\bullet\bullet} + \varepsilon} f_k^{-s^e_{j,i}}(s^e_{j,i}) + \frac{\gamma^e}{m^e_{\bullet\bullet} + \gamma^e} \cdot \frac{\varepsilon}{m_{\bullet\bullet} + \varepsilon} f_{k^{new}}^{-s_{j,i}}(s^e_{j,i}) \tag{12}$$

where $m^e_{\bullet\bullet}$ denotes the total number of tables in epoch *e* and $K^{\{H,U\}}$ means the set of the aspects available in the two epochs.

According to the distribution $y^e_{j,i,n} \sim binomial(\rho)$ and $\rho \sim beta(\beta)$, we can get the conditional probability of generating sentence $s^e_{j,i}$ given a specified aspect *k*:

$$f_k^{-s^e_{j,i}}(s^e_{j,i}) = \begin{cases} \dfrac{\Gamma(n_{\bullet\bullet k} + V\beta)}{\Gamma(n_{\bullet\bullet k} + \sum_k A^{(k)}_{(s^e_{j,i})} + V\beta)} \prod_{\substack{x^e_{j,i,n} = x^e_{j,i} \\ y^e_{j,i,n} = 1}} \dfrac{\Gamma(E^{(k)}_{(x^e_{j,i,n})} + E^{(s^e_{j,i})}_{(x^e_{j,i,n})} + \beta)}{\Gamma(E^{(k)}_{(x^e_{j,i,n})} + \beta)} & \text{if } k \text{ previously used} \\ \dfrac{\Gamma(V\beta)}{\Gamma(\sum_k A^{(k)}_{(s^e_{j,i})} + V\beta)} \prod_{\substack{x^e_{j,i,n} = x^e_{j,i} \\ y^e_{j,i,n} = 1}} \dfrac{\Gamma(E^{(s^e_{j,i})}_{(x^e_{j,i,n})} + \beta)}{\Gamma(\beta)} & k = k^{new} \end{cases} \tag{13}$$

$A^{(k)}_{(s^e_{j,i})}$ denotes the number of words that belong to aspect *k* in sentence $s^e_{j,i}$. It is obvious that background words do not influence the aspect assignment of sentences. $n_{\bullet\bullet k}$ is the total number of words assigned to aspect *k* in both epochs. *V* represents the size of vocabulary. $E^{(k)}_{(x^e_{j,i,n})}$ denotes the total number of times that word $x^e_{j,i,n}$ belongs to topic *k* in both docsets, and $E^{(s^e_{j,i})}_{(x^e_{j,i,n})}$ denotes the number of times that word $x^e_{j,i,n}$ exists in $s^e_{j,i}$.

If the sampled value of t^e_{ji} is t^{new}, then we can sample $k^e_{jt^{new}}$ according to (12):

$$p(k^e_{jt^{new}} = k \mid \boldsymbol{t}, \boldsymbol{k}^e_{-jt^{new}}) \propto \begin{cases} g(z^e_{j,i} = k \mid \boldsymbol{z}^e_{-j,i})\left(\dfrac{m^e_{\bullet k}}{m^e_{\bullet\bullet} + \gamma^e} + \dfrac{\gamma^e}{m^e_{\bullet\bullet} + \gamma^e} \cdot \dfrac{m_{\bullet k}}{m_{\bullet\bullet} + \varepsilon}\right) f_k^{-s^e_{j,i}}(s^e_{j,i}) & \textbf{if } k \in K^e \\ \dfrac{\gamma^e}{m^e_{\bullet\bullet} + \gamma^e} \cdot \dfrac{m_{\bullet k}}{m_{\bullet\bullet} + \varepsilon} f_k^{-s^e_{j,i}}(s^e_{j,i}) & \textbf{if } k \in K - K^e \\ \dfrac{\gamma^e}{m^e_{\bullet\bullet} + \gamma^e} \dfrac{\varepsilon}{m_{\bullet\bullet} + \varepsilon} f_{k^{new}}^{-s^e_{j,i}}(s^e_{j,i}) & \textbf{if } k = k^{new} \end{cases} \tag{14}$$

All the counts above except $A^{(k)}_{(s^e_{j,i})}$ and $E^{(s^e_{j,i})}_{(x^e_{j,i,n})}$ exclude the current sentence.

Sampling $\boldsymbol{k}$. Because the process of sampling $\boldsymbol{t}$ actually changes the component member of tables, we continue to sample k^e_{jt} for each table. The conditional probability $p(k^e_{jt} = k \mid \boldsymbol{t}, \boldsymbol{k}_{-jt})$ can be calculated similar to Eq. (14) and it is noted that a set of customers (not one custom) at table *t* should be considered.

Sampling $\boldsymbol{y}$. $y^e_{j,i,n}$ determines whether $x^e_{j,i,n}$ is a background word or an aspect word. If $y^e_{j,i,n} = 0$, $x^e_{j,i,n}$ is a background word, otherwise assigned to aspect *k*. we sample $y^e_{j,i,n}$ as:

$$p(y^e_{j,i,n} \mid z^e_{j,i} = k, \boldsymbol{y}_{-i,j,n}) = \begin{cases} \dfrac{C_{(0)} + \beta}{C_{(\bullet)} + 2\beta} \cdot \dfrac{C^{(x^e_{j,i,n})}_{(0)} + \lambda}{C_{(0)} + V\lambda} & \textbf{if } y^e_{j,i,n} = 0 \\ \dfrac{C_{(1)} + \beta}{C_{(\bullet)} + 2\beta} \cdot \dfrac{C^{(x^e_{j,i,n})}_{(k)} + \lambda}{C^{(k)}_{(1)} + V\lambda} & \textbf{if } y^e_{j,i,n} = 1 \end{cases} \tag{15}$$

where $C_{(\bullet)}$ denotes the total number of words in both docsets, $C_{(0)}$ denotes the total number of background words, $C_{(1)}$ denotes the total number of aspect words, and $C^{(k)}_{(1)}$ denotes the total

number of words that are assigned to aspect k. $C_{(0)}^{(x_{j,i,n}^e)}$ represents the number of times that word $x_{j,i,n}^e$ is assigned to background word and $C_{(k)}^{(x_{j,i,n}^e)}$ represents the total number of times that word $x_{j,i,n}^e$ is assigned to aspect k. The base measure G_0 was set a symmetric Dirichlet distribution with parameters λ (e.g. 0.5).

Based on Equations (11), (12) and (14), the aspect assignment probability of each sentence can be calculated as:

$$p(s_{j,i}^e = k \mid \boldsymbol{k}_{-ji}^e) \propto \begin{cases} g(z_{j,i}^e = k \mid \boldsymbol{z}_{-j,i}^e) n_{j\bullet k}^e f_k^{-s_{j,i}^e}(s_{j,i}^e) + \alpha^e g(z_{j,i}^e = k \mid \boldsymbol{z}_{j,i}^e) & \\ \times \left(\frac{m_{\bullet k}^e}{m_{\bullet\bullet}^e + \gamma^e} + \frac{\gamma^e}{m_{\bullet\bullet}^e + \gamma^e} \frac{m_{\bullet k}}{m_{\bullet\bullet} + \varepsilon}\right) f_k^{-s_{j,i}^e}(s_{j,i}^e) & \text{if } k \in K^e \\ \alpha^e \frac{\gamma^e}{m_{\bullet\bullet}^e + \gamma^e} \frac{m_{\bullet k}}{m_{\bullet\bullet} + \varepsilon} f_k^{-s_{j,i}^e}(s_{j,i}^e) & \text{if } k \in K^{(H,U)} - K^e \\ \alpha^e \frac{\gamma^e}{m_{\bullet\bullet}^e + \gamma^e} \frac{\varepsilon}{m_{\bullet\bullet} + \varepsilon} f_{k^{new}}^{-s_{j,i}^e}(s_{j,i}^e) & \text{if } k = k^{new} \end{cases} \tag{16}$$

As for the concentration parameters of h-uHDP, i.e., ε , α^e and γ^e, we sample them from a vague gamma prior which is set to be $Ga(10.0, 1.0)$. The sampling method is the same as that in (Teh et al., 2006).

5 Update Summarization with h-uHDP model

The task of update summarization aims to produce an update summary for the documents in the *update* epoch, assuming that users already read earlier documents in the *history* epoch. That is, we need to boost sentences in *update* epoch that can bring out important and novel information. On one hand, the generated summary should extract the main content in D^U, and on the other hand, the summary should avoid mentioning too much old information in D^H. To care for these two points, we propose a sentence selection strategy based on Kullback-Leibler (KL) divergence, which has been widely used in extractive summarization (Haghighi and Vanderwende, 2009; Mason and Charniak, 2011; Delort and Alfonseca, 2012).

Given the *history* sentence set S^H and the *update* sentence set S^U, we propose a function to score a set of sentences *Sum* which is a subset S^U.

$$Score(Sum) = KL(p_{S^H} \mid\mid p_{Sum}) - \kappa KL(p_{S^U} \mid\mid p_{Sum}) \tag{17}$$

In the equation, the first term means the prize on the divergence from epoch *history* and the second term represents the penalty on the divergence from epoch *update*. The parameter κ (called as *epoch balance factor*) is used to tune the weights of two KL distances. p_{Sum} is the empirical aspect distribution of the candidate summary *Sum*. p_{S^H} and p_{S^U} respectively denote the aspect distribution of S^H and S^U. $KL(p_{S^e} \mid\mid p_{Sum})$ $(e \in \{H, U\})$ represents the KL divergence given by $\sum_{k=1}^{K} p(S^e \mid k) \log \frac{p(S^e \mid k)}{p(Sum \mid k)}$. $p(\cdot \mid k)$ represents the probability distribution of a set of sentences on a specific aspect k, and is calculated based on the aspect assignment probability of each sentence which can be obtained according to Eq. (16).

$$p(S^e \mid k) = \frac{1}{\sum_j N_j^e} \sum_{s \in S^e} p(s = k), \quad p(Sum \mid k) = \frac{1}{\mid Sum \mid} \sum_{s \in Sum} p(s = k) \tag{18}$$

Generally, an optimum update summary should have the aspect distribution which approximates to p_{S^U} as possible and keep far away from the distribution p_{S^H}. Let Sum^* denote the optimum update summary. We can get Sum^* that maximizes the scoring function.

$$Sum^* = \underset{Sum \subseteq S^U \,\&\&\, words(Sum) \le L}{\arg\max} Score(Sum) \tag{19}$$

Since the problem of finding the subset of sentences from a collection that maximize the scoring function is NP-complete, a greedy algorithm is applied by adding sentences one by one. We use *Y* to denote the sentence set which contains the selected summary sentences. The algorithm first initializes *Y* to ϕ and *X* to S^U. During each iteration, we select from *X* one sentence (i.e. s_m) which makes $Score(s_m \cup Y)$ have the highest score. To avoid aspect redundancy in the summary, we also adopt the MMR strategy in the process of sentence selection. That is, for each s_m, we compute the semantic similarity between s_m and each sentence s_t in set *Y* as follows:

$$\cos\_sem(s_m, s_t) = \frac{\sum_k p(s_m \mid k) \cdot p(s_t \mid k)}{\sqrt{\sum_{k=1}^{K} p^2(s_m \mid k)} \cdot \sqrt{\sum_{k=1}^{K} p^2(s_t \mid k)}} \tag{20}$$

6 Experiments

In our experiments, we use four years of TAC (2008-2011) data, which contain 44-48 topics per year. For each topic, two docsets (named docset *H* and *U*) are given to respectively describe the *history* epoch and the *Update* epoch. Table 1 illustrates the number of topics, averaged number of documents per docset, and averaged number of sentences per docset for each year's data.

TAC	**2008**		**2009**		**2010**		**2011**	
Topics #	48		44		46		44	
docset	***H***	***U***	***H***	***U***	***H***	***U***	***H***	***U***
Avg Doc # per docset	10	10	10	10	10	10	10	10
Avg Sen # per docset	236.5	222.4	253.5	228.3	238.6	230.2	208.9	210.5

TABLE 1 – Experiment data (TAC 2008 - 2011).

As for the automatic evaluation of summarization, we still use the widely used ROUGE (Recall-Oriented Understudy for Gisting Evaluation) (Lin and Hovy, 2003) measures, including ROUGE-1, ROUGE-2, and ROUGE-SU4[5] and their corresponding 95% confidential intervals. In order to obtain a more comprehensive measure of summary quality, we also conduct manual evaluation on TAC 2011 dataset with the reference to (Haghighi and Vanderwende, 2009; Celikyilmaz and Hakkani-Tur, 2011; Delort and Alfonseca, 2011).

6.1 Parameter tuning

To get the final update summary using the h-uHDPSum algorithm, we still need to determine two parameters: sentence influence factor σ in Eq. (10) and epoch balance factor κ in Eq. (17). The combination of the two factors makes it hard to find a global optimized solution. So we apply a

[5]Jackknife scoring for ROUGE is used in order to compare with the human summaries.

gradient search strategy. At first, the epoch balance factor κ is fixed to a given value. Then the performance using different values of σ is evaluated. After that, we fix σ with the value which has achieved the best performance, and conduct experiments to find an appropriate value for κ. TAC 2008 and 2009 datasets are used as training data to tune these two parameters.

Firstly, κ is set to the value of 1, i.e. the prize on the divergence from epoch *history* is as important as the penalty on the divergence from epoch *update*. Reviewing Eq. (10), we can see that, the aspect assignment of one sentence is mainly determined by its neighboring sentences when σ is set a large value, whereas the influence from other sentences is not considered at all when σ is set 0. In the first place, we experiment the h-uHDPsum algorithm by setting σ in the range from 0 to 10 with interval of 1. The ROUGE scores drop sharply when σ is set a value larger than 2.0. Next, σ is set in the range from 0.0 to 2.0 with interval of 1.0. Fig. 4 presents the ROUGE-2 and ROUGE-SU4 evaluation results of h-uHDPSum, with regard to different values of σ. We find that the ROUGE scores reach their peak at around 1.0 and drop afterwords. The experimental results conform to our expectation and verify that the h-uHDP model is reasonable by considering the influence among sentences.

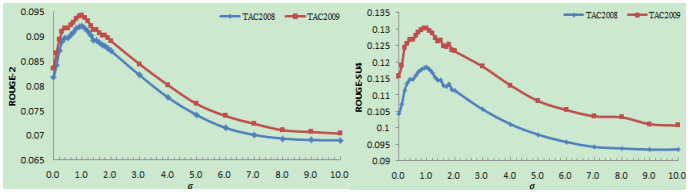


FIGURE 4 – Tuning parameter σ when κ is set to 1.

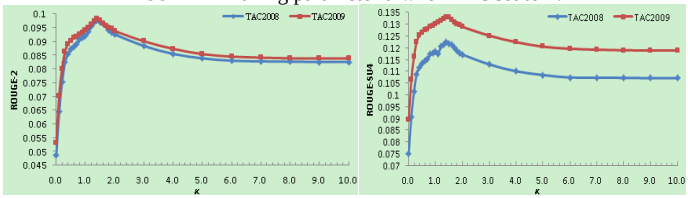


FIGURE 5 – Tuning parameter κ when σ is set to 1.

Next, we fix the sentence influence factor σ at 1.0 and tune the parameter κ. From Eq. (17), we can see that κ is used to balance the prize for the divergence from *history* epoch and penalty on the divergence from the update epoch. When κ is set as the value of 0, the scoring of sentences is only determined by docset *H*. That is, a sentence is likely to be selected into summary, only when it has a large divergence of aspect distribution from docset *H*. When κ is set a larger value, penalty on the divergence from docset *U* is more considered. Similar to the process of tuning σ, the performance using different values of κ ranging from 0 to 10 with interval of 1 is evaluated. We find that the peak performance of κ should be located in the range of [0.0, 2.0]. Thus, we conduct experiments to find an appropriate value for κ in the range from 0.0 to 2.0 with interval of 0.1. Fig. 5 shows the performance of h-uHDPSum with respect to κ. Performance gets better as κ increases from 0 to 1.4, and then declines gently until κ arrives at 3.0. Afterwards, the curve becomes smooth and means that the summarization algorithm is mainly up to docset *U* to decide which sentences to select. Parameters σ and κ are respectively set as 1.0 and 1.4 in the h-uHDPSum algorithm.

6.2 Comparison with other approaches

In this subsection, we compare our h-uHDPSum algorithm with several baseline methods on TAC 2010 and TAC 2011 datasets. One kind of baseline methods consists of the top three performing systems (denoted as SysRank 1st, 2nd and 3rd) on update summarization tasks according to the ROUGE-2 metric on TAC2010 and TAC2011. From Table 3, we can see that our approach obviously outperformed the top three participating systems both on TAC2010 and TAC2011, with respect to the ROUGE-2 and ROUGE-SU4 scores along with the corresponding 95% confidence intervals.

	TAC2010		TAC2011	
	ROUGE-2	**ROUGE-SU4**	**ROUGE-2**	**ROUGE-SU4**
h-uHDPSum	**0.0857(0.0784-0.0930)**	**0.1255(0.1182-0.1328)**	**0.1017(0.0910-0.1034)**	**0.1364(0.1265-0.1473)**
SysRank 1st	0.0799(0.0747-0.0851)	0.1198(0.1154-0.1244)	0.0959(0.0894-0.1029)	0.1309(0.1251-0.1366)
SysRank 2nd	0.0790(0.0740-0.0842)	0.1187(0.1142-0.1234)	0.0924(0.0857-0.0993)	0.1274(0.1217-0.1334)
SysRank 3rd	0.0729(0.0682-0.0779)	0.1080(0.1041-0.1122)	0.0863(0.0808-0.0920)	0.1280(0.1229-0.1330)
h-uHDPSum-noBG	0.0812(0.0767-0.0852)	0.1199(0.1120-0.1278)	0.0931(0.0874-0.0988)	0.1310(0.1258-0.1362)
2LevHDPSum	0.0780(0.0709-0.0851)	0.1171(0.1110-0.1232)	0.0917(0.0863-0.0971)	0.1315(0.1227-0.1401)
HDPSum	0.0708(0.0631-0.0785)	0.1091(0.1014-0.1176)	0.0842(0.0782-0.0902)	0.1218(0.1163-0.1273)
2LevLDASum	0.0720(0.0687-0.0793)	0.1152(0.1067-0.1237)	0.0879(0.0831-0.0925)	0.1310(0.1255-0.1366)
LDASum	0.0649(0.0594-0.0704)	0.1027(0.0936-0.1118)	0.0767(0.0704-0.0830)	0.1074(0.1015-0.1134)

TABLE 2 – Performance Comparison on TAC2010 and TAC2011.

To illustrate the effectiveness of our aspect modeling technique, we provide five other baseline systems which adopt different aspect modeling techniques. The systems *h-uHDPSum-noBG* and *2LevHDPSum* can be seen the simplified versions of *h-uHDPSum*. *h-uHDPSum-noBG* is the same as *h-uHDPSum* except that the general background information is not considered, whereas *2LevHDPSum* is a two-level (i.e. document level and sentence level) HDP model where the docset level is removed. At the same time, we implement one standard HDP model for comparison. As shown in Table 2, *h-uHDPSum* is better than both *h-uHDPSum-noBG*, *2LevHDPSum* and *HDPSum*, which verifies that the identification of background words or the introduction of the docset level can promote the performance of update summarization. Even without consideration of the background information, we can see the *h-uHDPSum-noBG* approach can be comparable to the best participating system of TAC evaluations. In addition, to compare with another popular modeling technique - Latent Dirichlet Allocation (LDA), we design a two-level LDA-based system *2LevLDASum* and a standard LDA-based system *LDASum*. Both *2LevLDASum* and *2LevHDPSum* are similar as possible beyond the distinction that *2LevLDASum* assume a fixed finite number of aspects[6] while *2LevHDPSum* does not. We can see that *2LevHDPSum* is better than *2LevLDASum* and *HDPSum* better than *LDASum* in performance. This can be easily explained, novel aspects can be automatically detected and the aspect number is determined naturally in HDP-based models. In contrast, how to determine the aspect number in the LDA-based models is still an open problem. This is also the reason why we select HDP as the foundation of our aspect modeling technique.

6.3 Manual evaluation

In order to obtain a more accurate measure of summary quality, manual evaluation is required. In this section, we compare our *h-uHDPSum* approach with *2LevLDASum* and the best participating

[6] In our experiments, the aspect number is set as 10, 20, 30 and 40 respectively and we select the best performed result with the aspect number as 20.

system (*Peer 43*). Similar to the manual evaluation in TAC, human assessors assign a score to each summary with respect to each of the following four criteria: 1) Overall responsiveness (overall performance in terms of content and fluency), 2) Focus (containing less irrelevant details), 3) Novelty (containing novel information beyond docset *H*), 4) Non-redundancy (repeating less the same information). The score is an integer between 1 (very poor) and 5 (very good). We randomly select 28 topics from TAC 2011 data and assign each topic to three different assessors[7]. In Table 3, the left four columns report the average scores of each criterion for the three systems. The experimental results indicate that *h-uHDPSum* is significantly better than both *Peer 43* and *2LevLDASum* (based on paired t-test with p-value < 0.01).

Simultaneously, a fairly standard approach for manual evaluation is conducted through pairwise comparison (Haghighi and Vanderwende, 2009; Celikyilmaz and Hakkani-Tur, 2011). According to the rating scores, each pair of summaries is judged which one is better under each criterion. If two summaries have the same score, they are judged a tie (of the equal quality). We record the times of 'winning' (having a higher score) and tie for each system. In Table 3, the right six columns show the evaluation results in frequencies respectively for *h-uHDPSum* vs. *Peer 43*, and *h-uHDPSum* vs. *2LevLDASum*. The experimental results also indicate that *h-uHDPSum* is significantly better than both *Peer 43* and *2LevLDASum*. We also observe that the winning times of *h-uHDPSum* under the novelty criterion is much more than those under the other criteria. This indicates that our approach can exhibit a clear advantage of promoting the novelty performance in update summaries.

				h-uHDPSum vs. *Peer 43*			*h-uHDPSum* vs. *2LevLDASum*		
	h-uHDPSum	*Peer 43*	*2LevLDASum*	*h-uHDP.*	Tie	*Peer 43*	*h-uHDP.*	Tie	*2LevLDA.*
Overall	**3. 94**	3. 65	3. 56	**31**	39	14	**50**	19	15
Focus	**4. 06**	3. 75	3. 65	**36**	26	22	**33**	48	3
Novelty	**4. 17**	3. 69	3. 67	**52**	13	19	**62**	6	16
Non-redund.	**4. 18**	3. 91	3. 98	**41**	22	21	**36**	44	4

TABLE 3 – Results of manual evaluation on TAC2011.

Conclusion

In this paper, we propose a novel approach based on a three-level HDP model h-uHDP for update summarization. The h-uHDP model can detect the birth, splitting, merging and death of specific aspects and the general background information for a given topic. Under the sentence extraction based framework of summarization, we especially strengthen modeling the sentence level in h-uHDP, where the aspect assignment of each sentence is influenced by its neighboring sentences. Based on h-uHDP, we propose a sentence selection strategy adopting KL divergence, which cares for both salience and novelty of sentences. Automatic and manual evaluations on TAC data illustrate that our approach obviously outperforms the state-of-the-art approaches.

Acknowledgments

The research work described in this paper has been partially supported by NSFC grants (No.90920011 and No.61273278), NSSFC grant (No: 10CYY023), National Key Technology R&D Program (No: 2011BAH10B04-03), and National High Technology R&D Program (No. 2012AA011101). We also thank the three anonymous reviewers for their helpful comments. Corresponding authors: Sujian Li (lisujian@pku.edu.cn) and Baobao Chang (chbb@pku.edu.cn).

[7] Each topic includes three update summaries generated by the three systems. A total of 1008 (28topics * 3systems*3persons*4criteria) scores need to be ranked. Six assessors participate the scoring.

References

Blackwell, D. and MacQueen J. B. (1973). Ferguson distributions via Polya urn schemes. The Annals of statistics, 1: 353-355.

Blei, D., Andrew, Y. N. and Jordan, M. L. (2003). Latent Dirichlet Allocation, in *Journal of Machine Learning Research*, vol 3: 993-1022.

Boudin, F., El-Beze, M. and Torres-Moreno, J. (2008). A scalable MMR approach to sentence scoring for multi-document update summarization. In *COLING 2008*, volume: Posters, pages 23–26.

Carbonell, J. and Goldstein, J. (1998). The use of MMR, diversity-based reranking for reordering documents and producing summaries. In *Proc. of the 21st SIGIR*, pages 335-336.

Celikyilmaz, A. and Hakkani-Tur, D. (2011). Discovery of topically coherent sentences for extractive summarization. In *Proc. of the 49th ACL*, pages 491-499.

Chakrabarti, D., Kumar, R. and Tomkins, A. (2006). Evolutionary clustering, in *Proc. of KDD2006*, pages 554‑560.

Chemudugunta, C., Smyth, P. and Steyvers, M. (2007). Modeling general and specific aspects of documents with a probabilistic topic model. In *Neural Information Processing Systems*. pages 241-248.

Chi, Y., Song, X., Zhou, D., Hino, K. and Tseng, B. L. (2007). Evolutionary spectral clustering by incorporating temporal smoothness, In *Proc. of KDD 2007*, pages 153‑162.

Delort, J. and Alfonseca, E. (2012). DualSum: a topic-Model based approach for update summarization, In *Proc. of the 13th EACL*, pages 214-223.

Du, P., Guo, J., Zhang, J. and Cheng, X. (2010). Manifold ranking with sink points for update summarization. In *Proc. of CIKM'2010*, pages 1757-1760.

John, C. M., Judith, S. D. and Dianne, O. P. (2009). Summarization and metrics. In *Proc. of TAC'09.*

Fisher, S. and Roark, B. (2008). Query-focused supervised sentence ranking for update summaries. In *Proc. of 1st Text Analysis Conference, TAC-2008*.

Gao, Z. J., Song, Y. and Liu, S. (2011). Tracking and connecting topics via incremental hierarchical Dirichlet Processes, In *ICDE 2011*. pages 1056-1061.

Gillick, D., Favre, B., and Hakkani-Tur, D. (2008). The ICSI summarization system at TAC 2008. In *Proc. Of TAC'08.*

Griffiths, T. L., Steyvers, M., Blei, D. M. and Tenenbaum, J. B. (2005). Integrating topics and syntax. In *Neural Information Processing Systems*, vol 17, pages 537-544.

Gruber, A., Weiss, Y. and Rosen-Zvi, M. (2007). Hidden topic Markov models. In *Proc. of the conference on Artificial Intelligence and Statistics.*

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

Li, P., Jiang, J and Wang, Y. (2010). Generating templates of entity summaries with an entity-aspect model and pattern mining, in *ACL 2010*, pages. 640-649.

Li, W., Wei, F., Lu, Q. and He, Y. (2008). *PNR*^{2}*:* ranking sentences with positive and negative reinforcement for query-oriented update summarization. *In Proc. of COLING '08*. Vol 1, pages 489–496.

Li, X., Du, L., and Shen, Y. (2012). Update summarization via graph-based sentence ranking, *IEEE Transactions on Knowledge and Data Engineering*.

Lin, C. Y. and Hovy, E. H. (2003). Automatic evaluation of summaries using N-gram co-occurrence statistics. *In Proc. of HLT-NAACL 200*3, pages 71-78.

Mason, R. and Charniak, E. (2011). Extractive multi-document summaries should explicitly not contain document-specific content. In *Proc. of WASDGML'11*, pages 49-54.

Mihalcea, R. and Tarau, P. (2004). TextRank: bringing order into texts. In *Proc. of EMNLP '04.*

Ren, L., Dunson, D. B. and Carin, L. (2008). The dynamic hierarchical Dirichlet process. In *ICML' 08*, pages: 824- 831.

Sethuraman J. (1994). A constructive definition of Dirichlet priors. In *Statistica Sinaca*, Vol 2 pages: 639-650.

Shen, C. and Li, T. (2010). Multi-document summarization via the minimum dominating set. *In COLING '10*, pages 984–992.

Steinberger, J. and Jezek, K. (2009). Update summarization based on novel topic distribution. *In Proceedings of DocEng'09*. pages 205-213.

Teh, Y., Jordan, M., Beal, M., and D. Blei. (2006). Hierarchical Dirichlet processes, *Journal of the American Statistical Association*, vol. 101, pages 1566‑1581.

Wan, X. (2007). TimedTextRank: Adding the Temporal Dimension to Multi-Document Summarization. *In Proc. of SIGIR'07*, pages 867-868.

Wang, D. and Li, T. (2010). Document update summarization using incremental hierarchical clustering. In *CIKM'10*, pages 279 ‑ 288.

Xu, T., Zhang, Z. M., Yu, P. S. and Long. B. (2008). Dirichlet process based evolutionary clustering. In *ICDM'08*. pages 648-657.

Xu, T., Zhang, Z. M., Yu, P. S., and Long. B. (2008). Evolutionary clustering by hierarchical Dirichlet process with hidden Markov state. In *ICDM'08*. pages 658-667.

Zhang, J., Song, Y., Zhang, C. and Liu, S. (2010). Evolutionary hierarchical Dirichlet processes for multiple correlated time-varying corpora, in *KDD* 2010. pages: 1079‑1088.

Employing Morphological Structures and Sememes for Chinese Event Extraction

LI Pei Feng and ZHOU Guo Dong[*]
School of Computer Science & Technology, Soochow University, Suzhou, China, 215006
{pfli, gdzhou}@suda.edu.cn

ABSTRACT

Current Chinese event extraction systems suffer much from the low recall due to unknown triggers. To resolve this problem, this paper firstly introduces morphological structures to better represent the compositional semantics inside Chinese triggers and then proposes a mechanism to automatically identify the head morpheme (either verb or noun) as the governing sememe of a trigger. Finally, it proposes a mechanism of combining the morphological structures and sememes of Chinese words to infer unknown triggers to improve the recall of the Chinese event extraction system. Evaluation on the ACE 2005 Chinese corpus justifies the effectiveness of our approach over a state-of-the-art system.

形态结构和义原在中文事件抽取中的应用

由于存在大量未知的触发词，当前的中文事件抽取系统受限于它的低召回率。为了解决这个问题，本文首先引入形态结构来更好地表示隐含在中文触发词内部的组合语义，然后提出了一个自动识别触发词中作为支配义原的核心词素（动词或名词）的机制。最后，本文提出了一个结合了中文词语的形态结构和义原去推测未知触发词的方法，用于提高中文事件抽取系统的召回率。在ACE 2005中文语料上的实验验证了我们方法的性能超越了目前最好的中文事件抽取系统。

KEYWORDS: Chinese event extraction; Morphological structure; Governing sememe; Trigger identification; Head morpheme.
Keywords in L2: 中文事件抽取; 形态结构; 支配语义; 触发词识别; 核心语素

[*] Corresponding author

Proceedings of COLING 2012: Technical Papers, pages 1619–1634,
COLING 2012, Mumbai, December 2012.

1 Introduction

As a compromise to natural language understanding, Information Extraction (IE) aims to extract structured information (e.g., entities, relations and events) from a text. Event extraction, a classic subtask in IE, is to recognize event trigger mentions of a predefined event type and their participants and attributes. While most studies in the literature focus on English event extraction, there are few successful stories concerning Chinese event extraction due to the special characteristics and challenges in Chinese language. Even with ground truth entities, times and values, the performance of most Chinese event extraction systems is much lower than that of English ones.

For Chinese event extraction, unknown triggers (a trigger in the test set doesn't occur in the training set and otherwise, a known trigger.) and word segmentation errors are two major reasons for the low performance, particularly the recall. The statistics on the ACE 2005 Chinese and English corpora (Li et al., 2012) shows that these two cases cover almost 30% of Chinese trigger mentions while this figure reduces to only about 9% in English. Besides, given the same number of event mentions, there are about 30% more different triggers in Chinese than those in English. This amplifies the problem. Therefore, trigger identification becomes a key to the success of Chinese event extraction.

Currently, there mainly exist two major mechanisms to solve this problem. The first one is to expand the triggers using predefined or automatically-clustered synsets, a common mechanism widely used in various NLP applications. The problem with this mechanism is that it fails to consider the sense shifting of a word in difficult contexts and thus may introduce too many pseudo triggers and harm the precision. This largely limits the contribution of this mechanism (Chen and Ji, 2009b; Ji, 2009; Qin et al. 2010). For example, as a trigger of the *Start-position* event, "担任" has more than five senses (e.g., serve as, bear, engage, do, etc.) and only one of them (serve as) can trigger a *Start-position* event. Take following two sentences as samples:

*(E1) 我们将**承担**所有本公司的费用。 (We will bear all the expenses for our company.)*

*(E2) 他将在IBM**从事**科学研究工作。 (He will engage in scientific research in IBM.)*

Although "承担" (bear) and "从事" (engage) are two synonyms of "担任", they do not trigger the *Start-position* event but any other events.

The second one is to expand the triggers using the compositional semantics inside Chinese words. The intuition is that if a Chinese word contains more than one character, and its meaning can be often inferred from the meanings of its component characters (Yuan, 1998). For example, Li et al. (2012) infer the semantics of a verb (most triggers in Chinese events are verbs) from its basic single-character verb (BV) and significantly improve the F1-measure, largely due to the dramatic increase in the recall. The problem with Li et al. (2012) is that they extract all single-character verbs contained in triggers as BVs (e.g., "担" (undertake, verb) and "任" (serve as, verb) are treated as two BVs for "担任" (serve as)). Therefore, pseudo triggers are much introduced. This severely harms the precision. Take the following sentence as a sample:

*(E3) 所有的公司员工**信任**他们的董事长。（All employees **trust** their chairman.）*

Although "信任" (trust) and "担任" have the same BV ("任") and the same verb structure (*verb+BV*), "信任"(trust) does not trigger the *Start-position* event but any other events.

Further analysis indicates that above two mechanisms are quite complementary. For example, we can find out that if we introduce the semantic similarity into the compositional semantics, "信任"(trust) in (E3) will not be expanded as a trigger for the *Start-position* event because of its different sense from "担任"(serve as), while if we introduce the compositional semantics into the semantic similarity, "从事"(engage) in (E2) will be filtered out from the trigger list of the *Start-position* event since it doesn't have the same BV as "担任" (serve as). However, a more refined mechanism is required to filter out "承担"(bear) in (E1).

In this paper, we first introduce the more general morphological structures in Chinese triggers, in place of verb structures in Li et al. (2102), to better represent the compositional semantics inside Chinese words and then propose a mechanism to automatically identify the head morpheme (either verb or noun) as the governing sememe of a trigger based on its morphological structure. The intuition behind is that the head morpheme can better represent the semantics of a Chinese word than the combination of all its component BVs. Finally, we propose a mechanism of combining the morphological structures and sememes of Chinese words to infer unknown triggers. Evaluation on the ACE 2005 Chinese corpus justifies the appropriateness of our approach.

To better understand the Chinese event extraction task as defined in ACE evaluations, where an event is defined as a specific occurrence involving participants, we list some ACE terminologies:

- **Event mention**: a phrase or sentence within which an event is described;
- **Trigger**: the main word that most clearly expresses the occurrence of an event, so recognizing an event can be recast as identifying a corresponding trigger;
- **Trigger mention:** a reference to a trigger.
- **Trigger type/Event type**: the type of an event;
- **Argument**: the entity mentions involved in an event;
- **Argument role**: the relation of an argument to an event where it participates.

In particular, the event extraction task is divided into four components:

- **Trigger identification**: to distinguish true trigger mentions from pseudo trigger mentions;
- **Event type determination**: to classify trigger mentions by event types;
- **Argument identification**: to distinguish true arguments from pseudo arguments;
- **Argument role determination**: to classify arguments by argument roles.

The rest of this paper is organized as follows. Section 2 overviews the related work. Section 3 describes various morphological structures in Chinese words and proposes a mechanism for determining the morphological structure and head morpheme in a Chinese trigger. Section 4 proposes an algorithm to infer unknown triggers on their morphological structures and sememes. Section 5 presents the experimental results. Finally, we conclude the paper with future work.

2 Related work

In the literature, most of existing studies on event extraction concern English and can be classified into either pattern-based (e.g., Riloff, 1996; Yangarber et al., 2000; Stevenson and Greenwood, 2005; Shinyama and Sekine, 2006; Patwardhan and Riloff, 2007; Chambers and

Jurafsky, 2011) or classifier-based (e.g., Grishman et al., 2005; Ahn, 2006; Hardy et al., 2006; Maslennikov and Chua, 2007; Ji and Grishman, 2008; Patwardhan and Riloff, 2009; Liao and Grishman, 2010; & 2011; Hong et al., 2011; Lu and Roth, 2012; Llorens et al., 2012). In particular, while earlier studies focus on sentence-level extraction, later ones turn to employ global information.

Compared with tremendous work on English event extraction, there are only a few studies on Chinese event extraction with focus on either feature engineering or trigger expansion, under the same framework as English event extraction.

On feature engineering, Tan et al. (2008) first employ a local feature selection method to ensure the performance of trigger classification and then apply multiple levels of patterns to improve the coverage in argument classification. Fu et al. (2010) apply a feature weighting scheme to re-weight various features for trigger identification and event type determination. Chen and Ji (2009b) apply various kinds of lexical, syntactic and semantic features to address the special issues in Chinese. Li et al. (2012) extend Chen and Ji (2009b) with more refined features and additional dependency and semantic role features.

On trigger expansion, Chen and Ji (2009a) propose a bootstrapping framework to exploit extra information captured by an English event extraction system. Ji (2009) first extracts some cross-lingual predicate clusters using bilingual parallel corpora and a cross-lingual information extraction system, and then employs the derived clusters to expand the triggers. Qin et al. (2010) employ a semantic dictionary "TongYiCi Ciling (expanded version)" to expand triggers for Chinese event type determination. Li et al. (2012) propose an inference mechanism to infer new triggers by employing the verb structures to explore the compositional semantics inside Chinese triggers (verbs only) and achieve the state-of-the-art performance of 67.4% in F1-measure on the ACE 2005 Chinese corpus, ignoring the post-processing – discourse consistency.

3 Morphological structures and head morphemes inside Chinese triggers

In this section, we introduce various morphological structures to better represent the compositional semantics inside Chinese triggers and then propose two mechanisms to identify the morphological structures and the head morpheme in Chinese triggers respectively.

3.1 Compositional semantics and morphological structures in Chinese words

Both in English and Chinese languages, a word is composed of one or more characters. However, a component character in English is just the basic unit to form a word instead of a semantic unit. In comparison, almost all Chinese characters have their own meanings and are called morpheme (or single-morpheme word), the minimal meaningful unit in Chinese language. If a Chinese word contains more than one character, its meaning can be often interpreted in terms of its composite characters/morphemes. This more fine-grained semantics are the compositional semantics inside Chinese words namely. Actually, it is also a normal way to understand a new Chinese word in everyday life for a Chinese native speaker.

Without doubt, a general method to represent the compositional semantics inside Chinese words is to systematically explore the morphological structures in Chinese words since it is the nature of compound words. Morphological structures in Chinese words are the word-building process to form the morphemes into words and are formulated by three major processes: compounding, affixation, and conversion. Compounding is a process, by which two or more morphemes are composed together to form a compound word. Affixation is a morphological process to add

grammatical or lexical information to a base form. By the conversion process, a word is changed from one part-of-speech (POS) into another without the addition or deletion of any morphemes. Compounding is the most productive way to compose a Chinese word while affixation is the most popular way to construct an English word. Affixation also is used widely in Chinese, but its prefix or suffix doesn't have the meaning and can be always omitted (e.g., "老虎" (tiger) and "虎" (tiger) have the same meaning.). As for conversion, it's really not a way to construct a word and just represents the fact that some words have more than one tense.

3.2 Morphological structures in Chinese triggers

Since almost all triggers in Chinese events are verbs and nouns, we focus on the morphological structures of Chinese verbs and nouns. Actually, statistics on the ACE 2005 Chinese corpus shows that 95% of triggers are either verbs or verbal nouns and just nearly 5% are pure nouns (e.g., "公开信" (open letter), "大会" (plenary session)). In ACE 2005 English corpus, there are some adjectives triggering an event of special type. However, no adjective acts as a trigger in the ACE 2005 Chinese corpus for the special characteristics in Chinese language. Besides, almost 95% of triggers in the training set just contain one or two morphemes, so this paper only considers the one-morpheme and two-morpheme triggers of verbs and nouns.

There are two type words in Chinese triggers: single-morpheme words and compound words. Single-morpheme word just contains one morpheme. Sometimes, a single-morpheme word maybe is composed by more than one character, such as the transliterated word. But it doesn't occur in Chinese triggers and we disregard them in this paper. So there is only one morphological structure concerning a single-morpheme trigger:

Single-Morpheme Structure: Single-morpheme trigger whose POS is a verb or a noun (e.g., "死" (die), "去" (go), "信" (letter), etc.).

Compounding is the most productive way to compose a Chinese trigger. In this paper we define five types (similar to (Chang, 1995)) of the morphological structures in Chinese triggers based on the relations between their morphemes.

Coordinative Structure: The two morphemes of a trigger play coordinative role. For example, "合" (combine) and "并" (merge) are coordinative in trigger "合并" (merge).

Modifier-Head Structure: The modified morpheme follows the modifying one in a trigger. For example, "婚" (marry) is modified by "新" (new) in trigger " 新婚" (newly-married).

Subject-Predicate Structure: One morpheme is the subject and the other one tells something about the subject. This structure is like a subject-predicate sentence condensed in a trigger. For example, "身" (body) is a subject of predicate "亡" (die) in trigger "身亡" (die).

Predicate-Object Structure: The first morpheme (predicate) governs the second one (object) in a trigger. For example, "业" (business) serves as the object of predicate "开" (start) in trigger "开业" (start business).

Predicate-Complement Structure: The first morpheme is a predicate and the second one interprets the first one from different aspects (e.g., direction, result and tense) in a trigger. For example, morpheme"入" (into) expresses the direction of action "进" (go) in trigger "进入" (go into).

3.3 Determining the morphological structure in a Chinese trigger

A general method to determine the morphological structures in Chinese triggers is to first annotate some instances manually and then train a classifier. Alternatively, a simple way is employed in this paper to determine the morphological structures in Chinese triggers via their POS structures, due to our finding that the morphological structures in Chinese triggers can be inferred from their POS structures. Following are the inference rules employed in this paper for different morphological structures:

Single-Morpheme Structure: For a single-morpheme trigger whose POS is a noun or a verb, its morphological structure is *Single-Morpheme*. The statistics on the training set shows that this inference rule covers almost 100% of cases given correct POSs.

Predicate-Complement Structure: If the POS structure of a trigger is (*verb + preposition*) or *(verb + auxiliary*), its morphological structure is *Predicate-Complement*. The statistics on the training set shows that this inference rule covers almost 100% of cases given correct POSs.

Predicate-Object Structure: If the POS structure of a trigger is (*verb + noun)*, its morphological structure is *Predicate-Object*. The statistics on the training set shows that this inference rule covers almost 100% of cases given correct POSs.

Coordinative Structure: If the POS structure of a trigger is (*verb + verb*) (e.g., "捐/VV 赠/VV" (donate), "购/VV 买/VV" (buy), etc.), its morphological structure is *Coordinative*. The statistics on the training set shows that this inference rule covers almost 98% of cases given correct POSs. The only exception to this inference rule is that it ignores those triggers whose POS structure is (*noun + noun*), This happens in Chinese triggers, though seldom. In such cases, i.e. if the POS structure of a trigger is (*noun + noun*), its morphological structure can be either *Modifier-Head* or *Coordinative* (e.g., "婚/NN 姻/NN" (marriage)).

Modifier-Head Structure: The morphological structure of a trigger is *Modifier-Head*, *if* its POS structure is one of following four structures: 1) (*adjective + verb*); 2) (*adjective + noun*); 3) (*noun + noun*); 4) (*noun + verb*). The statistics on the training set shows that this inference rule covers almost 96% of cases given correct POSs. The only exceptions to this inference rule are that if the POS structure of a trigger is (*noun + noun*) or (*noun + verb*), its morphological structure can also be *Coordinative* or *Subject-Predicate*, respectively.

Subject-Predicate Structure: Our exploration on the ACE 2005 Chinese corpus shows that only one trigger (i.e. "身亡" (die)) has the *Subject-Predicate* structure. Therefore, we ignore this structure.

Structure	% Trigger mentions
Single-Morpheme	19.1
Coordinative	46.3
Modifier-Head	13.3
Predicate-Object	11.4
Predicate-Complement	8.7
Words (length>=3)	1.2

TABLE 1 – Distribution of different morphological structures in Chinese trigger mentions

To obtain the POS structures of Chinese triggers, we split all triggers into characters and employ a Chinese POS tool – ICTCLAS to tag their POSs. Table 1 shows the distribution of the morphological structures in Chinese triggers in the training set, extracted using above inference rules. Random manual evaluation of 1000 instances shows that our inference rules achieve the accuracy of more than 91% given automatically-tagged POSs.

3.4 Identifying head morpheme in Chinese triggers

Normally, almost all Chinese verbs or nouns contain one morpheme as the governing semantic element, called Head Morpheme (HM), to construct a word and the semantics of such a word thus can be inferred from its HM. Since the semantics of a Chinese trigger can be often inferred from its HM, it's natural to infer unknown triggers via HMs. For example, given verb "死" (die) as HM in trigger "烧死" (burn to death, trigger of the *Die* event) whose morphological structure is *Coordinative*, it is reasonable to infer "砸死" (crush to death), "炸死" (burst to death), "闷死" (stifle to death) to be triggers of the same event, due to their same HM and morphological structure as "烧死".

Li et al. (2012) regards all single-character verbs contained in triggers as BVs and use them to infer unknown triggers. It may introduce many pseudo triggers into candidates and harm the precision for that loose constraint. For example, the morphological structure of "烧死" is *Coordinative*, and "烧" (burn) and "死" (die) are two single-morpheme verbs in it. Following Li's inference rule, all words including BV "烧" or "死" are regarded as triggers if their verb structures are (*BV* + *verb*) or (*verb* + *BV*). Hence, some pseudo triggers, such as "烧烤" (barbecue), "烧焊" (weld), "烧制" (fire), etc., would be expanded to be triggers.

Besides, a noun may be a HM to infer new triggers. For example, given "信" (letter) as the HM in trigger "私信"(private letter, trigger of *Phone-Write* event) whose morphological structure is *Modifier-Head*. It's correctly to infer those words (e.g., "贺信" (congratulatory letter), "密信" (secret letter), etc.) with the HM "信" (letter) and the morphological structure *Modifier-Head*, as triggers.

Therefore, how to identify the HM in a Chinese trigger becomes the key to infer unknown triggers. Table 2 shows our automatic mechanism to identify HM, where *LM(w)* and *RM(w)* are used to obtain the left and right morphemes from one-morpheme or two-morphemes word *w* respectively.

Structure	Inferences to select HM
Single-morpheme	*tr*
Coordinative	LM(*tr*): **if** SSIM*(tr,* LM*(tr))* > *α* RM(*tr*): **if** SSIM*(tr,* RM*(tr))* > *α*
Modifier-Head	RM(*tr*)
Predicate-Object	RM(*tr*)
Predicate-Complement	LM(*tr*)

TABLE 2 – Inferences on different morphological structures to extract HMs

For a trigger whose morphological structure is *Single-morpheme*, *Predicate-Complement* or *Modifier-Head*, it's easy to identify its HM from the relationship between its two morphemes. If the structure of a trigger is *Predicate-Object*, we select the noun (object) as HM because it better represents the semantics of the trigger than the predicate, i.e. the governing semantic element

always comes from the object. However, without additional information, it's hard to select HM from a trigger whose morphological structure is *Coordinative*. For example, given the trigger "访问" (visit) whose morphological structure is *Coordinative*, its two component morphemes, "访" (visit) and "问" (ask), have their own semantics respectively. Fortunately, we can find out that morpheme "访" (visit) has the same meaning as trigger "访问" (visit). So an effective way to identify HM in a trigger with the *Coordinative* structure is via the semantic similarity (SemSim).

In this paper, we employ HowNet[1] (Dong and Dong, 2006) to obtain the semantics of Chinese words. Similar to Wordnet in English, HowNet is a structured Chinese lexical semantic resource. In HowNet, sememe is a basic semantic unit and represents the meaning of a word. In total, about 2200 sememes are used to define 95000 Chinese words. In this paper, the governing sememe is introduced to recognize HMs from those triggers with the *Coordinative* structure. That is, if a morpheme represents the governing sememe, it is recognized as HM of that trigger. Following Liu and Li (2002), function *SemSim(x, y)* is used to calculate the semantic similarity between the sememes of the trigger *x* and its morpheme *y* as follow:

$$SemSim(x, y) = \frac{\phi}{Dis(x, y) + \phi} \tag{1}$$

where *Dis(x,y)* is the distance between the sememe of *x* and *y* in HowNet's sememe hierarchical architecture, and ϕ is an adjustable parameter and assigned 0.75 following Liu and Li (2002).

4 Inferring unknown triggers on HMs and sememes

To better represent the compositional semantics inside Chinese words and filter out more pseudo triggers, we introduce the morphological structures and sememes of Chinese words to infer unknown triggers. The methodology is shown as follows: 1) following the principle of compositional semantics, we extract these one-morpheme or two-morpheme words in the test set as candidates when they contain at least one HM and their POS are nouns or verbs; 2) according to the morphological structure of each candidate word, we applied different inferences to choose unknown triggers. We implement an algorithm to determine whether a candidate is an unknown trigger and the input and output are shown as follows:

Input: *HMs* $\leftarrow$ the set of all HMs extracting from the training set

$$candidates \leftarrow \{w \mid \begin{array}{l}(LM(w) \in HMs \vee (RM(w) \in HMs) \\ \wedge (POS(w) = noun \vee POS(w) = verb)\} \\ \wedge MPRO(MORPH(w), HM(w)) \geq 0\end{array}$$

$$triggerwords \leftarrow \phi$$

Output: *triggerwords* : the set of unknown triggers accepted by our algorithm

POS(w) and *HM(w)* are applied to get the POS of word *w* and obtain the HM in word *w* respectively. *MPRO(ms, hm)* is defined to compute the conditional probability of a trigger when it contains a HM *hm* and its morphological structure is *ms*. *MORPH(w)* is used to get the morphological structure of word *w*.

For each candidate word *w* in *candidates*, we apply following inferences to distinguish the true unknown triggers from the pseudo ones according to the morphological structure and sememe.

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

Single-Morpheme: These expanding single-morpheme words are those HMs in two-morpheme triggers. So we apply a simple constraint to determine whether or not it's an unknown trigger:

$$\underset{m_i \in S}{MAX}(SemSim(w, m_i)) = 1 \tag{2}$$

where S is the set of triggers in the training set which contain word w. If the maximum score of the semantic similarity between these triggers and word w is equal to 1, we accept it.

Predicate-Complement: The first morpheme is usually a verb, so the sememe of word w always is similar to the sememe of its first morpheme. The constraint for *Predicate-Complement* structure is:

$$LM(w) \in S_{sm} \cup S_{pc} \tag{3}$$

where S_{sm} is the set of triggers in the training set whose structures are *Single-morpheme* while S_{pc} is the set of left morphemes of triggers in the training set whose structures are *Predicate-Complement*.

Predicate-Object: for a word w whose morphological structure is *Predicate-Object*, we regard it as the unknown trigger following two conditions to constrain its two morphemes:

$$RM(w) \in HMs \tag{4}$$

$$\underset{m_i \in SW}{MAX}(SemSim(LM(w), m_i)) \geq \beta \tag{5}$$

where SW is the set of predicates in the similar triggers[2] of word w. For example, if there are two triggers "离职" (resign) and "辞职" (resign), and their HMs are "职" (job) too. For a candidate "免职" (resign), its morphological structure is as same as the above two and its HM also is "职" (job). We call them similar triggers and calculate the similarities between "免"(dismiss) and the predicates ("离" (leave), "辞" (dismiss)) in its similar triggers in the training set.

Modifier-Head: The first morpheme of word w modifies the second one, so that the semantics of word w comes from its second morpheme. We apply following rules based on POS consistency and semantic similarity.

$$RM(w) \in HMs \tag{6}$$

$$POS(LM(w)) \in \{POS(l) \mid COM(l,b) \in S_{mh} \wedge b \in HMs\} \tag{7}$$

$$\underset{m_i \in S}{MAX}(SemSim(w, m_i)) = 1 \tag{8}$$

where S_{mh} is the set of triggers in the training set whose structures are *Modifier-Head* and *COM(l,b)* is to combine morpheme l and b to be a two-morpheme word. Otherwise, S is the set of those triggers which contain word w.

Coordinative: Since the two composite morphemes of word w are homogeneous and its semantics is flexible and maybe comes from the combination of its two morphemes or one of its morpheme. We calculate the average score of the similarities to infer trigger of this type:

[2] Similar triggers are those triggers with the same morphological structure and the same BM in the training set.

$$\frac{\sum_{co \in \{w, LM(w), RM(w)\}} \underset{m_i \in SC}{MAX}(SemSim(co, m_i))}{3} > \lambda \quad (9)$$

where

$$SC \in \{s | MORPH(s) = MORPH(w) \wedge (LM(w) \in HMs \wedge LM(s) = LM(w) \vee RM(w) \in HMs \wedge RM(w) = RM(s))\} \quad (10)$$

where *SC* is the set of triggers in the training set with following two constraints: 1) their morphological structures are *Coordinative*; 2) their left/right morphemes and the left/right morpheme of word *w* are the same HM.

5 Experimentation and discussion

In this section, we evaluate our mechanism of combining the morphological structures and sememes of Chinese words in inferring unknown triggers and report the experimental results on trigger identification and its application to overall Chinese event extraction.

5.1 Experimental setting and baseline

We use a state-of-the-art Chinese event extraction system (Li et al., 2012) as one of our baselines which consists of four typical components (trigger identification, event type determination, argument identification and argument role determination) in a pipeline way. During testing, each word in the test set is first scanned for instances of known triggers from the training set and then scanned by employing the compositional semantics inside Chinese triggers to infer instances of unknown triggers. When an instance is found, the trigger identifier is applied to distinguish those true trigger mentions from pseudo ones. If true, the event type determiner is then applied to recognize its event type. For any entity mention in a sentence which is identified as an event, the argument identifier is employed to assign its possible arguments afterwards. Finally, the argument role determiner is introduced to assign a role to each argument.

Besides, we adopt the same experimental setting as Li et al. (2012). The ACE 2005 Chinese corpus (only the training data is available) is used in all our experiments. The corpus contains 633 Chinese documents annotated with 8 predefined event types and 33 predefined event subtypes[3]. We randomly select 567 documents as the training set and the remaining 66 documents as the test set. Besides, we reserve 33 documents in the training set as the development set and follow the setting of ACE diagnostic tasks and use the ground truth entities, times and values for our training and testing. As for evaluation, we also follow the standards as defined in Li et al (2012):

- A trigger is **correctly** identified if its position in the document matches a reference trigger;
- An event type is **correctly** determined if the trigger's event type and position in the document match a reference trigger;
- An argument is **correctly** identified if its involved event type and position in the document match any of the reference argument mentions;
- An argument role is **correctly** determined if its involved event type, position in the document, and role match any of the reference argument mentions.

[3] Similar to previous studies, we treat these subtypes simply as 33 separate event types and do not consider the hierarchical structure among them.

Finally, all the sentences in the corpus are divided into words using a word segmentation tool (ICTCLAS) with all entities annotated in the corpus kept. Besides, we use Berkeley Parser and Stanford Parser to create the constituent and dependency parse trees. We use N-gram features and employ the ME model[4] to train individual component classifiers.

5.2 Results on identifying HMs and unknown triggers

As the key to infer unknown triggers, Table 3 shows the performance of HM identification. For evaluation, the HMs of all the known triggers in the ACE 2005 Chinese corpus are manually labeled by three annotators and we accept those morphemes as HMs when at least two annotators agree on them. The thresholds α is fine-tuned to 0.85 using the development set. Compared to Li et al. (2012), our approach can improve the F1-measure by 6.9%, largely due to the dramatic increase in Precision of 15.8%. Li et al. (2012) extracted all single-character verbs as BVs, so their Recall is higher than that of ours. Otherwise, we extract 30 single-morpheme nouns as HMs and 73% of them occur in the gold set while this number in Li et al. (2012) is 0.

System	#BV/HMs	P(%)	R(%)	F1
Li et al. (2012)	361	64.3	88.5	74.5
Ours	266	80.1	82.1	81.4

TABLE 3– Performance of the HM identification (#Gold: 262)

We apply the mechanism of combining the morphological structures and sememes of Chinese words (CMS) to infer unknown triggers. The thresholds β and λ are fine-tuned to 0.7 using the development set. Following Li et al. (2012), we also apply the non-trigger filtering rule in our system and just filter out those candidates which occur as pseudo triggers more than 5 times in the training set. So we obtain a candidate set of words including known triggers in the training set and those unknown triggers identified by our mechanism. Manual inspection shows that 62 words are inferred as unknown triggers, among which 69.4% are true triggers.

To verify the effectiveness of our mechanism, we extract those trigger mentions from the test set when they are instances of known triggers from the training set or unknown triggers extracted by CMS. Table 4 shows the results of our CMS and two baseline systems in inferring unknown trigger mentions. Here, Baseline-1 (Chen and Ji (2009b)) just extracts those trigger mentions occurring in the training data while Baseline-2 (Li, et al., 2012) infers unknown trigger mentions based on the compositional semantics and verb structures of Chinese words.

System	#True trigger mentions	#Pseudo trigger mentions
Baseline-1	266	629
Baseline-2	302	444
CMS	326	508
Gold	367	-

Table 4 – Impact of combining the morphological structure and sememe of Chinese words in inferring unknown triggers

Compared with Baseline-1 and Baseline-2, our mechanism recovers 16.3% (60) and 6.5% (24) of true trigger mentions respectively. This improvement mainly comes from two factors. The first one is that we introduce those nouns to be HMs and almost 20% of the true unknown triggers

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

(e.g., “失**业**” (lose one’s job), “出**境**” (leave the country)) are extracted. The second one is that our mechanism filters out more pseudo trigger mentions due to the contribution of combining the morphological structures and sememes of Chinese words. For example, Baseline-2 will infer “**调**频” (frequency adjustment) “妨**害**” (impair) to be triggers due to “调” (adjust) and “害” (harm) are BVs and their syntactic structures are (*BV+noun*) and (*verb+BV*) respectively. On the contrary, our mechanism will filter out “**调**频” since its structure is *Modifier-Head* and the head morpheme “频” (frequency) doesn’t appear in HMs while “妨**害**” will also be ignored because its sememe is not similar to any known triggers with the same HM “害” (harm). It justifies the effectiveness of our mechanism to combine the morphological structures and sememes of Chinese words in recovering true triggers.

Otherwise, some triggers in the training set are seldom used as trigger mentions. We also applied above mechanism to filter out those triggers. Table 4 shows that almost 28% of pseudo trigger mentions is filtered out, so the number of pseudo trigger mentions is reduced to 508.

5.3 Results on trigger identification and overall Chinese event extraction

There are too many pseudo trigger mentions showed in Table 4 by using our mechanism to infer unknown triggers and extract trigger mentions from the test set, so we introduce a ME-based trigger identifier to distinguish the true trigger mentions from the pseudo ones as previous works.

Table 5 shows the contribution of our mechanism to trigger identification on the held-out test set. Compared to Baseline-1, our approach can dramatically improve the F1-measure by 10.0%, with a big gain of 17.8% in Recall and a small loss of 1.8% in Precision. It further proves the effectiveness of the compositional semantics in inferring Chinese unknown triggers. Compared to the state-of-the-art system *(*Baseline-2), our approach also enhances F1-measure by 4.1%, largely due to a dramatic increase of 7.7% in Recall. It also justifies that the morphological structures of Chinese words are more effective than the verb structures when they are employed to infer unknown triggers. Besides, these results also show that introducing sememes of Chinese words into our mechanism is a helpful way to filter out those pseudo triggers.

We also employ the mechanism of discourse consistency (Li et al., 2012) to improve the Precision and our results show that our approach achieves 79.4%, 69.2% and 73.9% in F1-measure, Precision and Recall respectively and it outperforms Li et al. (2012) by 3.4% and 5.7% in F1-measure and Recall, with a small loss of 0.1% in Precision.

System	Trigger identification		
	P(%)	R(%)	F1
Baseline-1	75.2	52.0	61.5
Baseline-2 (Li et al. (2012))	73.5	62.1	67.4
CMS	73.4	**69.8**	71.5
Baseline-2+ Discourse consistency	79.3	63.5	70.5
CMS + Discourse consistency	**79.4**	69.2	**73.9**

Table 5 – Contribution to Chinese trigger identification

Table 6 shows the contribution of trigger identification to overall event extraction on the held-out test set. Compared to Baseline-2, we can find that our approach can improve the F1-measure for event type determination by 4.0%, argument identification by 3.3% and argument role determination (i.e. overall event extraction) by 2.9%, largely due to the dramatic increase in

Recall of 7.4%, 6.1% and 5.6%. These results also ensure the importance of trigger identification in Chinese event extraction.

System	Event type determination			Argument identification			Argument role determination		
	P(%)	R(%)	F1	P(%)	R(%)	F1	P(%)	R(%)	F1
Baseline-1	70.3	49.0	57.8	58.4	42.7	49.3	55.2	38.6	45.4
Baseline-2	70.2	59.1	64.2	58.0	48.9	53.0	54.7	44.5	49.1
CMS	**69.9**	**66.5**	**68.2**	**57.6**	**55.0**	**56.3**	**54.1**	**50.1**	**52.0**

Table 6 – Contribution to Overall Chinese event extraction

5.4 Discussion

Through manual inspection, we find that many remaining errors are related to three aspects. The first one is that almost 4.7% of trigger mentions in the test set doesn't have a morpheme appeared in the set of HMs. For example, there are so many ways to hurt a human to express an *injure* event and just a few of triggers or its HMs occurred in the training set. The second one comes from the errors in POS tagging in the verb structures of triggers and constituent parse tree. Almost all errors in determining morphological structures are come form those wrong POSs, especially those single-morpheme triggers, with the wrong POS in the parse tree will be ignored in inferring unknown triggers. The last one is the low quality of the annotated event corpus and many event mentions are missed. Those un-annotated true mentions would make the classifier confuse to distinguish true event mentions from pseudo ones. We look into those pseudo trigger mentions which are classified as true ones by the ME classifier and find out almost 20% of them maybe are true ones by our knowledge.

In order to evaluate the effect of the training set size on the performance, we modify the proportion of the training set to the test set from 9:1 to 1:9. Fig. 2 shows the percentages of true trigger mentions extracted by our baseline and our CMS. From Figure 1, we can find out that our mechanism can extract much more true trigger mentions than that of the baseline, especially for a smaller training set. When the proportion of the training set to the test set is set to 1:9, our mechanism can extract 67.5% of true trigger mentions while the figure drops to 43.3% in our baseline. This justifies that our mechanism can be well applied to minimally-supervised event extraction.

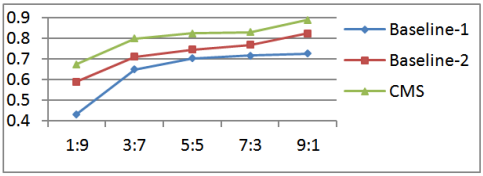


FIGURE 1 – The percentages of extracting true trigger mentions on different proportions of the training set to the test set

Compared to Li et al. (2012), There are three contributions in our work: 1) we use the morphological structure to better represent the compositional semantics inside Chinese triggers; 2) we introduce a mechanism to identify HMs in triggers automatically and those HMs can be verbs or nouns; 3) we propose a mechanism of combining the morphological structures and sememes of

Chinese words to extract unknown triggers. The results show that our mechanism outperforms the state-of-the-art system.

Conclusion

To address the special characteristics of Chinese event extraction and extract more true trigger mentions, this paper presents a novel approach to Chinese trigger identification which combines the morphological structures and sememes of Chinese words to infer unknown triggers. The experimental results show that our approach can significantly improve the performance of the Chinese event extraction system, especially Chinese trigger identification in Recall. In future work, we will focus on how to apply the mechanism of compositional semantics to unsupervised or minimally supervised event extraction system and improve their performance.

Acknowledgments

The authors would like to thank three anonymous reviewers for their comments on this paper. This research was supported by the National Natural Science Foundation of China under Grant Nos. 61070123 and 61273320, the National 863 Project of China under Grant No. 2012AA011102.

REFERENCES

Ahn, D. (2006). The stages of event extraction. In *Proceedings of the Workshop on Annotating and Reasoning about Time and Events (ARTE 2006)*, pages 1-8.

Chambers, N. and Jurafsky, D. (2011). Template-based information extraction without the templates. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics (ACL 2011)*, pages 976–986.

Chang, S. M. (1995). A study on Mandarin compounds. Hsinchu: National Tsing Hua University.

Chen, Z. and Ji, H. (2009a). Can one language bootstrap the other: a case study on event extraction. In *Proceedings of the NAACL HLT 2009 Workshop on Semi-Supervised Learning for Natural Language Processing (SemiSupLearn 2009)*, pages 66-74.

Chen, Z. and Ji, H. (2009b). Language specific issue and feature exploration in Chinese event extraction. In *Proceedings of Human Language Technologies: the 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL 2009)*, pages 209-212.

Dong Z. D. and Dong Q. (2006). HowNet and the computation of meaning. World Scientific Pub Co. Inc.

Finkel, J., Grenager, T. and Manning, C. (2005). Incorporating non-local information into information extraction systems by Gibbs sampling. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics (ACL 2005)*, pages 363-370.

Fu, J. F., Liu, Z. T., Zhong, Z. M. and Shan, J. F. (2010). Chinese event extraction based on feature weighting. *Information Technology Journal*, 9: 184-187.

Gupta, P. and Ji, H. (2009). Predicting unknown time arguments based on cross-event propagation. In *Proceedings of the ACL-IJCNLP 2009 Conference Short Papers (ACLShort 2009)*, pages 369-272.

Grishman, R., Westbrook, D. and Meyers, A. (2005). NYU's English ACE 2005 system description. In *Proceedings of ACE 2005 Evaluation Workshop (ACE workshop 2005)*.

Hardy, H., Kanchakouskaya, V. and Strzalkowski, T. (2006). Automatic event classification

using surface text features. In *Proceedings of the Twenty-first National Conference on Artificial Intelligence Workshop on Event Extraction and Synthesis (AAAI workshop 2006)*, pages 36-41.

Hong, Y., Zhang, J., Ma, B., Yao, J. M., Zhou, G. G. and Zhu, Q. M. (2011). Using cross-entity inference to improve event extraction. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL-HLT 2011)*, pages 1127-1136.

Ji, H. (2009). Cross-lingual predicate cluster acquisition to improve bilingual event extraction by inductive learning. In *Proceedings of the Workshop on Unsupervised and Minimally Supervised Learning of Lexical Semantics (UMSLLS 2009)*, pages 27-35.

Ji, H. and Grishman, R. (2008). Refining event extraction through cross-document inference. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL-HLT 2008)*, pages 254-262.

Li, J. H., Zhou, G. D. and Ng, H. T. (2010). Joint syntactic and semantic parsing of Chinese. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics (ACL 2010)*, pages 1108-1117.

Li, P. F., Zhou G. D., Zhu Q. M. and Hou L. B. (2012). Employing compositional semantics and discourse consistency in Chinese event extraction. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL 2012)*, pages 1006-1016.

Liao, S. S. and Grishman, R. (2010). Using document level cross-event inference to improve event extraction. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics (ACL 2010)*, pages 789-797.

Liao, S. S. and Grishman, R. (2011). Using prediction from sentential scope to build a pseudo co-testing learner for event extraction. In *Proceedings of the 5th International Joint Conference on Natural Language Processing (IJCNLP 2011)*, pages 714–722.

Llorens, H., Saquete, E., Navarro-Colorado, B. (2012). Applying semantic knowledge to the automatic processing of temporal expressions and events. *Information Processing & Management*.

Lu, W. and Roth, D. (2012). Automatic event extraction with structured preference modeling. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL 2012)*, pages 835–844.

Maslennikov, M. and Chua, T. (2007). A multi resolution framework for information extraction from free text. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics (ACL 2007)*, pages 592-599.

Patwardhan, S. and Riloff, E. (2007). Effective information extraction with semantic affinity patterns and relevant regions. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-coNLL 2007)*, pages 717-727.

Patwardhan, S. and Riloff, E. (2009). A unified model of phrasal and sentential evidence for information extraction. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing (EMNLP 2009)*, pages 151-160.

Qin, B., Zhao, Y. Y., Ding, X., Liu, T. and Zhai, G. F. (2011). Event type recognition based on trigger expansion. *Tsinghua Science and Technology*. 15(3): 251-258.

Riloff, E. (1996). Automatically generating extraction patterns from untagged text. In *Proceedings of the Thirteenth National Conference on Articial Intelligence*, pages 1044–1049.

Shinyama, Y. and Sekine, S. (2006). Preemptive information extraction using unrestricted relation discovery. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics (NAACL-HLT 2006)*, pages 304–311.

Stevenson, M. and Greenwood, M. (2005). A semantic approach to IE pattern induction. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL 2005)*, pages 379–386.

Tan, H., Zhao, T., Zheng, J. (2008). Identification of Chinese event and their argument roles. In *Proceedings of the 2008 IEEE 8th International Conference on Computer and Information Technology Workshops (CITWORKSHOPS 2008)*, pages 14-19.

Yangarber, R., Grishman, R., Tapanainen, P. and Huttunen, S. (2000). Automatic acquisition of domain knowledge for information extraction. In *Proceedings of the 18th International Conference on Computational Linguistics (ACL 2000)*, pages 940–946.

Yuan, M. L. (1998). Studies on valency in modern Chinese. Chinese Commerce and Trade Press, Beijing, China.

Joint Modeling of Trigger Identification and Event Type Determination in Chinese Event Extraction

LI Pei Feng, ZHU Qiao Ming, DIAO Hong Jun and ZHOU Guo Dong[*]
School of Computer Science & Technology, Soochow University, Suzhou, China, 215006
{pfli, qmzhu, hjdiao, gdzhou}@suda.edu.cn

ABSTRACT

Currently, Chinese event extraction systems suffer much from the low quality of annotated event corpora and the high ratio of pseudo trigger mentions to true ones. To resolve these two issues, this paper proposes a joint model of trigger identification and event type determination. Besides, several trigger filtering schemas are introduced to filter out those pseudo trigger mentions as many as possible. Evaluation on the ACE 2005 Chinese corpus justifies the effectiveness of our approach over a strong baseline.

一个应用于中文事件抽取的事件触发词识别和类型判别联合模型

当前，有2个问题困扰着中文事件抽取系统：低质量的事件标记语料库和假事件触发词相对于真事件触发词的高比例。为了解决以上2个问题，本文提出了一个结合事件触发词识别和事件类型判别的联合模型。另外，几个触发词过滤模式同样被引入本系统用于过滤掉尽可能多的假触发词实例。在ACE2005中文语料上的测试结果表明，本文的方法和基准系统相比具有更高的性能。

KEYWORDS: Joint modeling, Event type determination, Trigger identification, Trigger filtering.
KEYWORDS IN L2: 联合模型, 事件类型判别, 触发词识别, 触发词过滤

[*]Corresponding author

Proceedings of COLING 2012: Technical Papers, pages 1635–1652,
COLING 2012, Mumbai, December 2012.

1 Introduction

Information extraction (IE) is a task of extracting structured information (e.g. entities, relations and events) from the text. As a critical part of IE, event extraction is to identify trigger mentions of a predefined event type, and their participants and attributes. It can be typically divided into four components: trigger identification, event type determination, argument identification and argument role determination. Due to the central role of the contained events in a text, it is critical to mine their semantics in order to understand a text. Unfortunately, event extraction has been proven its performance is still very low.

In the literature, most studies focus on English event extraction and have achieved certain success (e.g., Grishman et al., 2005; Ahn, 2006; Patwardhan and Riloff, 2009; Hong et al., 2011; Lu and Roth, 2012; Llorens et al., 2012). However, there are few successful stories regarding Chinese event extraction due to the special characteristics in Chinese trigger identification. Besides unknown triggers[1] and word segmentation errors (Li et al., 2012), the low quality of annotated corpora and the high ratio of pseudo trigger mentions to true ones are also blamed for the low performance of Chinese event extraction.

To examine the low quality of annotated corpora in Chinese event extraction, we take the ACE (Automatic Content Extraction) 2005 Chinese corpus (with 8 types and 33 subtypes of events), one of the most popular corpora in event extraction, as an example. In particular, we randomly select 33 documents from the training set and ask two human annotators to manually tag event mentions and their types following the definition of the ACE 2005 corpus. Here, human annotator 1 is a first year postgraduate student with no background in Chinese event extraction while human annotator 2 is a third year postgraduate student working on Chinese event extraction. Table 1 justifies the difficulty of Chinese event extraction, particularly for trigger identification and event type determination, even for a well-educated human being. As shown in Table 1, the IAA (Inter-Annotator Agreement) on both trigger identification and event type determination is well below 50%. Even so, it is not surprising since the IAA on trigger identification on the ACE 2005 English corpus is only about 40% (Ji and Grishman, 2008).

Performance / Human	Trigger identification			Event type determination		
	P%	R%	F1	P%	R%	F1
annotator1 (blind)	63.3	62.9	63.1	61.7	59.5	60.6
annotator2 (familiar)	72.6	74.3	73.4	69.1	70.2	69.6
Inter-Annotator Agreement	45.8	42.9	44.3	45.3	42.5	43.8

TABLE 1 – Low quality of human annotation in the ACE 2005 Chinese corpus

Detailed analysis shows that one major reason for the low quality of human annotation is due to the difficulty of following the specified annotation guidelines, as mentioned in Ji and Grishman (2008). To better justify this issue, we randomly select 20 triggers and extract all the sentences which contain those triggers from the training set. Our exploration shows that although almost all the annotated trigger mentions are true ones, ensuring the reliability of the annotated trigger mentions, many true trigger mentions, e.g., those with exactly the same constituent or dependency structure as annotated ones, are not annotated, accounting for about 10% of trigger mentions. Table 2 shows the statistics.

[1] A trigger word/phrase occurring in the training data is called a known trigger and otherwise, an unknown trigger.

#Triggers	#Sentences	#Annotated trigger mentions	#un-annotated trigger mentions
20	452	198	23

TABLE 2 – Statistics of annotated vs. un-annotated trigger mentions in the ACE 2005 Chinese corpus

Take following two sentences as examples:

*(E1) 3 名抗议者在**冲突**中受伤。(Annotated trigger mention)*

*(Three protestors were injured in the **conflict**.)*

*(E2) 双方各有数人在**冲突**中受伤。(Un-annotated trigger mention)*

*(Several people from both sides were injured in the **conflict**.)*

Although the two examples are similar, "冲突" (conflict) in example (E1) is annotated as a trigger mention of the *Conflict* event while the one in example (E2) is not annotated. With the extreme example of "战争" (war), as the trigger of the *Conflict* event, among 11 trigger mentions concerned with "朝鲜战争" (Korean war) and "海湾战争" (gulf war), four of them are annotated as *Conflict* event while the others are ignored. Those un-annotated true trigger mentions would make the classifier difficult to distinguish true trigger mentions from pseudo ones.

For the high ratio of pseudo trigger mentions to true ones, Table 3 shows top 5 imbalanced triggers from the training set of the ACE 2005 Chinese corpus and justifies the difficulty for a classifier to identify a true trigger mention, especially for those of a particular event type, which appears only a few times in the training set.

Trigger[2]	#True trigger mentions	#Pseudo trigger mentions
投资 (invest)	1	67
建设 (set up)	1	66
取得 (obtain)	1	52
发 (provide)	1	36
给 (give)	2	64

TABLE 3 – Top 5 triggers with the highest ratios of pseudo trigger mentions to true ones in the ACE 2005 Chinese corpus

Recently, Li et al. (2012) justified that trigger identification was most critical for the performance of Chinese event extraction. In this paper, we also focus on trigger identification and its impact on overall Chinese event extraction.

In order to address the above-mentioned two critical issues in Chinese event extraction, this paper proposes a joint model of trigger identification and event type determination to improve the performance of trigger identification and overall Chinese event extraction. Besides, several trigger filtering schemas are introduced to filter out those pseudo trigger mentions as many as possible.

The rest of this paper is organized as follows. Section 2 overviews the related work. Section 3

[2] Most Chinese words have more than one sense. Here, we just give the one when it acts as a trigger.

describes the joint model of trigger identification and event type determination. Section 4 introduces those trigger filtering schemas. Section 5 evaluates our approach and shows its effectiveness over a strong baseline. Section 6 concludes the paper with future work.

2 Related work

To better understand the Chinese event extraction task as defined in ACE, where an event is defined as a specific occurrence involving participants, we list some ACE terminologies:

Event mention: a phrase or sentence within which an event is described, including a trigger and its arguments.

Trigger: the main word which most clearly expresses the occurrence of an event, so recognizing an event can be recast as identifying a corresponding trigger.

Trigger mention: a reference to a trigger word.

Trigger type/Event type: the type of an event.

Argument: the entity mentions involved in an event.

Argument role: the relation of an argument to an event where it participates.

In the literature, almost all the existing studies on event extraction are concerned with English. While earlier studies focus on sentence-level extraction (Grishman et al., 2005; Ahn, 2006; Hardy et al., 2006), later ones turn to employ high-level information, such as document (Maslennikov and Chua, 2007; Finkel et al., 2005; Patwardhan and Riloff, 2009), cross-document (Ji and Grishman, 2008), cross-event (Gupta and Ji, 2009; Liao and Grishman, 2010) and cross-entity (Hong et al., 2011) information.

2.1 Chinese event extraction

Compared with tremendous efforts in English event extraction, there are only a few studies on Chinese event extraction.

Some studies focused on feature selection. Tan et al. (2008) used a local feature selection method to ensure the performance of trigger classification and applied multiple levels of patterns to improve their coverage in argument classification. Fu et al. (2010) applied a feature weighting algorithm to re-weight various features extracted for trigger identification and event type determination. Chen and Ji (2009b) applied various kinds of lexical, syntactic and semantic features to address the special issues in Chinese. They also constructed a global errata table to record the inconsistency in the training set and used it to correct the inconsistency in the test set.

The other studies focused on automatic expansion of event triggers to improve the recall. Chen and Ji (2009a) proposed a bootstrapping framework, which exploited extra information captured by an English event extraction system. Ji (2009) first extracts some cross-lingual predicate clusters using bilingual parallel corpora and a cross-lingual information extraction system, and then employs the derived clusters to expand the triggers. Qin et al. (2010) described a method to expand the event triggers for Chinese event type determination based on a Chinese semantic dictionary "TongYiCi CiLin (expansion version)". Li et al. (2012) proposed a novel inference mechanism to infer new trigger words by employing compositional semantics inside Chinese triggers. Their system achieved the state-of-the-art performance of 67.4 units in F1-measure on the ACE 2005 Chinese corpus, ignoring the post-processing – discourse consistency.

2.2 Joint modeling

While a pipeline model may suffer from the errors propagated from upstream tasks, a joint model can benefit from the close interaction between two or more tasks: it not only allows the uncertainty about one task to be carried forward to next ones but also allows useful information from one task to be carried backward to previous ones. Recently, joint modeling has been widely attempted in various NLP tasks, such as joint named entity recognition and syntactic parsing (Finkel and Manning, 2009), joint syntactic parsing and semantic role labeling (Li et al., 2010), joint anaphoricity and coreference determination (Denis and Baldridge, 2007; Iida and Poesio, 2011).

In the event extraction task, only a few studies are concerned with joint modeling, mostly in the bio-molecular domain. Riedel et al. (2009) used Markov Logic as a general purpose framework for jointly modeling the complete bio-molecular event structure for a given sentence. Poon and Vanderwende (2010) also adopted Markov Logic for bio-molecular event extraction in jointly predicting events and their arguments. Riedel and McCallum (2011) presented three joint models for bio-molecular event extraction. While the first model jointly predicts triggers and their arguments and the second model enforces additional constraints that ensure the consistency between events in hierarchical regulation structures, the third model integrates the first one and the second one in explicitly capturing the interaction of various arguments in the same event. Do et al. (2012) constructed a timeline of events mentioned in a given text which proposed a joint inference module that enforced global coherency constraints on the final outputs of the two pairwise classifiers, one between event mentions and time intervals, and one between event mentions themselves.

Our joint model is inspired by both Roth and Yih (2004) on joint named entity recognition and relation extraction and Denis and Baldridge (2007) on joint anaphoricity determination and coreference resolution. However, as far as we know, there are no successful models for jointly solving Chinese trigger identification and event type determination.

2.3 Trigger filtering

With the high ratio of pseudo trigger mentions to true ones, it is natural to filter out those unlikely trigger mentions in a preprocessing step. Basically, the general purpose for instance filtering is to reduce the class distribution imbalance by discarding harmful or superfluous instances.

In the literature, instance filtering has been widely employed in various NLP tasks. As for event extraction, there are also a few relevant studies. Patwardhan and Riloff (2009) first applied a self-trained relevant sentence classifier to identify relevant regions and split all candidate sentences into two sets: relevant and irrelevant sentences. Then, they used a pattern-based classifier to recognize events from those relevant sentences and a SVM-based classifier to recognize events from those irrelevant sentences. Landeghem et al. (2009) provided a negative-instances filter to check whether the length of the sub-sentence spanned by a candidate event does not exceed a certain value. Landeghem et al. (2010) further designed a false-positive filter using specific categories of relations to serve as negative indicators in Bio-NLP. Liao and Grishman (2010) applied a pseudo co-testing algorithm based on various criteria, such as informativeness, representativeness and diversity of the sentence, to filter out those pseudo samples to reduce annotation labour in event corpus annotation.

3 Joint modeling of trigger identification and event type determination

In this section, an ILP (Integer Logic Programming) -based inference framework is proposed to jointly model trigger identification and event type determination in reducing the influence of un-annotated true trigger mentions in the ACE 2005 Chinese corpus. Besides, a CRF (Conditional Random Field) model is applied as a supplement to the ME model to capture local sequential information around a trigger mention in trigger identification.

3.1 Joint inference of trigger identification and event type determination

As mentioned in Section 1, many true trigger mentions in the ACE 2005 Chinese corpus are not annotated. When training a classifier to identify trigger mentions, these un-annotated true trigger mentions in the training set will be extracted as negative samples. This will make the trigger identifier wrongly classify many true trigger mentions as pseudo ones, resulting in low recall in trigger identification. On the contrary, without the interference of these un-annotated true trigger mentions, the event type determiner has the higher probability of recognizing these annotated true trigger mentions as some kinds of events. This indicates the necessity and potential of jointly modeling for trigger identification and event type determination.

Besides, although the ME (Maximum-Entropy) model has been widely used in various subtasks of event extraction and achieved certain success in capturing the global information around a trigger mention, our experimentation shows that it suffers from low precision in trigger identification. To overcome this problem, a CRF model is introduced in trigger identification to capture the local sequential information. Our preliminary experimentation shows that the CRF model is much complementary to the ME model in trigger identification.

In our joint model, an ILP-based inference framework is introduced to integrate two trigger identifiers and one event type determiner. Figure 1 shows the ILP-based inference framework, which integrates a CRF-based trigger identifier (CRF_I) with an ME-based trigger identifier (ME_I) and an ME-based event type determiner (ME_D). The features used by ME_D and ME_I are as same as Li et al. (2012).

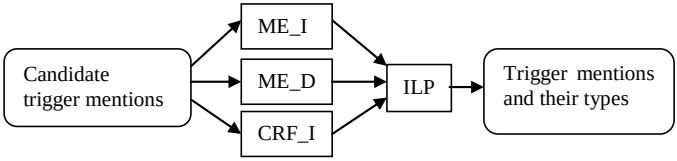


FIGURE 1 – Joint modeling of trigger identification and event type determination

ILP is a mathematical method for constraint-based inference to find the optimal values for a set of variables that minimize an objective loss function in satisfying a certain number of constraints. In the literature, ILP has been widely used in various NLP tasks (e.g., Roth and Yih, 2004; Barzilay and Lapata, 2006; Iida and Poesio, 2011; Do et al., 2012) in combining multiple classifiers, where the traditional pipeline architecture is not appropriate.

We assume $p_{ME\_I}(EVENT|Tr_{i,j})$ is the probability of ME_I identifying a trigger mention as a true one, where $Tr_{i,j}$ is the *jth* mention of the *ith* trigger word in a discourse, and $p_{ME\_d}(R_k\ |Tr_{i,j})$ *is* the probability of ME_D determining a trigger mention as an event of type R_k. Like Roth and Yih (2004), we define following assignment costs:

$$c^{I}_{<i,j>} = -\log(p_{ME\_I}(EVENT \mid Tr_{i,j})) \quad (1)$$

$$\bar{c}^{I}_{<i,j>} = -\log(1 - p_{ME\_I}(EVENT \mid Tr_{i,j})) \quad (2)$$

$$c^{D}_{<i,j>} = -\log(p_{ME\_D}(R_k \mid Tr_{i,j})) \quad (3)$$

$$\bar{c}^{D}_{<i,j,k>} = -\log(1 - p_{ME\_D}(R_k \mid Tr_{i,j})) \quad (4)$$

where $c^{I}_{<i,j>}$ is the cost of $Tr_{i,j}$ as an event trigger mention while $\bar{c}^{I}_{<i,j>}$ is the cost of $Tr_{i,j}$ not as an event trigger mention; $c^{D}_{<i,j>}$ is the cost of $Tr_{i,j}$ as an event trigger mention of type R_k while $\bar{c}^{D}_{<i,j,k>}$ is the cost of $Tr_{i,j}$ not as an event trigger mention of type R_k.

Besides, we use indicator variable $x_{<i,j>}$ that is set to 1 if $Tr_{i,j}$ is an event mention, and 0 otherwise. Similar to $x_{<i,j>}$, we use another indicator variable $y_{<i,j,k>}$ that is set to 1 if $Tr_{i,j}$ is an event mention of type R_k, and 0 otherwise. Finally, the objective function of the ILP-based inference framework can be represented as follows, where D is the set of trigger words in a discourse and M_i is the set of all mentions with the same *ith* trigger word,

$$\begin{aligned} \min \sum_{i \in D} \sum_{j \in M_i} (c^{I}_{<i,j>} x_{<i,j>} + \bar{c}^{I}_{<i,j>} (1 - x_{<i,j>}) \\ + \sum_{i \in D} \sum_{j \in M_i} \sum_{1 \le k \le 33} c^{D}_{<i,j,k>} y_{<i,j,k>} + \bar{c}^{D}_{<i,j,k>} (1 - y_{<i,j,k>}) \end{aligned} \quad (5)$$

Subject to

$$x_{<i,j>} \in \{0,1\} \qquad \forall i \in D \wedge j \in M_i \quad (6)$$

$$y_{<i,j,k>} \in \{0,1\} \qquad \forall i \in D \wedge j \in M_i \wedge 1 \le k \le 33 \quad (7)$$

To enforce consistency, we add further constraints:

(C1) Event type constraint: if a trigger mention $Tr_{i,j}$ belongs to event type R_k, it must be a true trigger mention.

$$x_{<i,j>} \ge y_{<i,j,k>} \qquad \forall i \in D \wedge j \in M_i \wedge 1 \le k \le 33 \quad (8)$$

(C2) True trigger mention constraint: if a mention $Tr_{i,j}$ is a true trigger mention, it must belong to only one event type R_k.

$$x_{<i,j>} = \sum_{1 \le k \le 33} y_{<i,j,k>} \qquad \forall i \in D \wedge j \in M_i \quad (9)$$

(C3) Discourse consistency: all trigger mentions which have the same trigger word must have the same event type in a discourse, or all of them aren't true trigger mentions.

$$x_{<i,j>} = x_{<i,l>} \qquad \forall i \in D \wedge j, l \in M_i \quad (10)$$

As a discourse-driven language, the syntax of Chinese is not as strict as English and very often we need to count on the discourse-level information to understand the meaning of a Chinese sentence. As for an event, a trigger may appear many times in a discourse and a trigger is considered discourse-consistent when all its appearances have the same event type. The statistics on the training sets of both the ACE 2005 Chinese and English corpora shows that within a discourse, there is a strong consistency in both Chinese and English between trigger mentions: if one instance of a word is a trigger, all the other instances in the same discourse will be a trigger

of the same event type with a very high probability (> 90% in Chinese).

(C4) Different event types for trigger mentions in a clause: Different trigger mentions in a clause must have different event types.

$$y_{<i,j,k>} + y_{<r,t,k>} < 2 \quad \forall Tr_{i,j} \in cl_a \wedge Tr_{r,t} \in cl_a \wedge i,r \in D \wedge j \in M_i \wedge t \in M_r \wedge 1 \le k \le 33 \wedge Tr_{i,j} \neq Tr_{r,t} \quad (11)$$

where cl_a is the set of words in clause a.

For example, trigger mentions "暴力" (violence, *Conflict* event) and "冲突" (conflict, *Conflict* event) may occur together to form a phrase "暴力冲突" and we should identify them as one *Conflict* event instead of two.

(C5) Cross-event constraint: Those events with high probability of co-occurring in a discourse must have same indicator values (event or non-event).

$$y_{<i,j,k>} = y_{<r,t,k'>} \quad \forall i,r \in D \wedge j \in M_i \wedge t \in M_r \wedge <k,k'> \in O \wedge 1 \le k \le 33 \wedge 1 \le k' \le 33 \quad (12)$$

where O is the set of event type pairs with high probability of co-occurring[3] in the training set.

As mentioned in Liao et al. (2010), there are strong correlations among event types in a document. We also find out that some events have a high probability of co-occurring in a discourse. For examples, if there is a *Die* event in a discourse, there is more than 70% probability that an *Attack* event also appears in the same discourse.

3.2 CRF-based trigger identification

CRF is a conditional sequence model which represents the probability of a hidden state sequence given some observations. It is a popular and efficient machine learning technique for supervised sequence labeling and has been applied to many NLP tasks.

We choose CRF due to its ability of capturing the local information around a trigger mention. For this purpose, we build a separate character-based trigger identifier and use the CRF model to label each character with a tag indicating whether it is out of a given trigger (O), the beginning of the trigger (B) or a part of the trigger except the beginning one (I). In this way, our CRF-based trigger identifier performs sequential labeling by assigning each character one of the three tags and a character assigned with tag B is concatenated with following characters with tag I to form a trigger. For example, example (E1) can be labelled as follows and ***冲突*** is identified as a trigger.

*(E3) 3/O 名/O 抗/O 议/O 者/O 在/O **冲/B 突/I** 中/O 受/O伤/O 。/O*

To achieve high precision as much as possible, we just use the character itself and characters around it as features. For each character c_i, assuming its 5-windows characters are c_{i-2} c_{i-1} c_i c_{i+1} c_{i+2}, our CRF-based trigger identifier adopts following features: c_{i-2}, c_{i-1}, c_i, c_{i+1}, c_{i+2}, $c_{i-1}c_i$, c_ic_{i+1}, $c_{i-2}c_{i-1}c_i$, $c_{i-1}c_ic_{i+1}$, $c_ic_{i+1}c_{i+2}$.

Our preliminary experimentation shows that the CRF model achieves high precision and is much complementary to the ME model in trigger identification. In this paper, the CRF-based trigger identifier is included into the ILP-based inference framework by introducing one more constraint.

(C6) CRF trigger constraint: due to high precision of the CRF model, we include a simple inference rule in our joint model:

[3] The threshold of the probability of the event type pair is fine-tuned to 0.70 using the development set.

$$x_{<i,j>} = 1 \quad \text{if the CRF model identifies } Tr_{i,j} \text{ as an event} \tag{13}$$

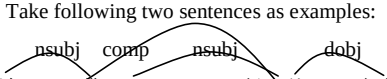
4 Trigger filtering

In this section, we firstly introduce two trigger filtering mechanisms: dependency-based inference mechanism and divide-and-conquer mechanism, to remove pseudo trigger mentions. Then we provide another trigger filtering mechanism by employing both local and global discrimination to filter out those un-annotated true trigger mentions.

4.1 Dependency-based inference mechanism

Some single-character trigger words are very ambiguous, e.g. with more than one senses and POSs (part of speeches), and it is hard to tell their true trigger mentions from their pseudo ones. For example, "到" (come) has 246 mentions (not including those words containing "到") corresponding to 6 senses and 3 POSs, and only 84 of them are true trigger mentions. In this paper, according to the total number of senses and POSs, we select 32 top ambiguous single-character words, such as "到", "往", "并", and employ a dependency-based inference mechanism to filter out their occurrences as pseudo trigger mentions.

In a sentence, there normally exists strong structural dependency between a trigger and its arguments. Take following two sentences as examples:



(E4) 朱婉清/PER 带 儿子/PER 已经 到 了 纽约/GPE。

(Zhu Wanqing has arrived New York with her son.)

(E5) 我/PER 找 不 到 工作。

(I can't find a job.)

(E4) is a positive example where "到" (arrive) is a trigger mention of the *Movement* event and there is a strong structural dependency between the trigger and its arguments, while example (E5) is a negative example where "到" is not a *Movement* event and there is no obvious dependency between the trigger and subject "我" (I).

In this paper, we adopt Markov Logic Network (MLN) to determine whether a single-character word is a trigger mention or not. For this purpose, we construct two inference formulas based on the dependency and POS information as follows, similar to Poon and Vanderwende (2010):

$$Token(i,+w) \wedge Pos(i, +o) \Rightarrow Event(i)$$

$$Token(j,+w) \wedge Dep(i, j, +r) \Rightarrow Event(i)$$

where

Token(i,w): whether token *i* has word *w*;

Pos(i, o): whether token *i* has POS *o*;

Dep(i, j, r): whether there is a dependency edge from *i* to *j* with relation *r* or relation path *r* (e.g., ccomp->nsubj, pp->pobj);

Event (i): whether token *i* is an event trigger mention.

Here, notation "+" signifies that the MLN contains an instance of the formula, with a separate weight, which is learnt from the training set. In particular, the open-source *Alchemy* package[4] is employed for learning and inference. Like Pooh and Vanderwende (2010), we use the Stochastic Gradient Descent (SGD) to learn weights and introduce MC-SAT, a slice sampling Markov chain Monte Carlo algorithm, to make the inference. To obtain a final assignment, we set the query atoms with probability no less than 0.3 (fine-tuned to maximize F1 on the development set) to true and the rest to false, in order to keep true trigger mentions.

4.2 Divide-and-conquer mechanism

Trigger mentions with high ratios of pseudo trigger mentions to true ones are treated differently from those with low ratios, using a threshold θ[5]. For the former, two patterns are applied to filter those high-unlikely trigger mentions as follows, while for the later, we adopt a ME classifier to filter out pseudo trigger mentions as many as possible.

(P1) <entity type of subject> <trigger[6]*>*

(P2) <trigger> <entity type of direct/indirect object>

where the subject and object must be the arguments of that event.

4.3 Local and global discrimination

Like the representativeness of an event, the *discrimination* considers the distributional similarity of a pseudo trigger mention against those true trigger mentions. If a pseudo trigger mention is similar to one of those true trigger mentions, it should be filtered out from the set of pseudo trigger mentions due to its low *discrimination*; otherwise, it should be kept in the set of pseudo trigger mentions due to its high *discrimination*. For an un-annotated true trigger mention which is extracted as a pseudo trigger mention, it tends to have the large distributional similarity with those true trigger mentions and should be filtered from the set of pseudo trigger mentions due to its low discrimination.

Normally, the distribution of events of a particular type is not balanced. For example, in the ACE 2005 Chinese corpus, *Movement* events occur most frequently in the training set with 701 times and occupy 22.0% of all event occurrences, while for the 10 least frequently-occurring event types (e.g. *Execute*, *Delare-Bankruptcy*, *Divorce*, etc.), each of them only occupies less than 1%.

To well address the above phenomenon, this paper introduces two types of discrimination, local discrimination *local_d* and global discrimination *global_d*, to filter out those un-annotated true trigger mentions and reduce their negative impact on trigger identification.

On the one hand, the local discrimination measures the similarity between a particular pseudo trigger mention and all of true trigger mentions with the same trigger word (shorted as STMs). In our case, each trigger mention is represented as a vector of features and the cosine similarity is applied to measure the similarity between a pseudo trigger mention and each STM.

If the pseudo trigger mention is similar to one STM, their similarity will be high. Instead of calculating the average similarity, we calculate the maximum similarity to identify whether the pseudo trigger mention should be filtered out:

[4] http://alchemy.cs.washington.edu/

[5] Threshold θ is fine-tuned to 3, using the development set.

[6] We just extract those trigger mentions with POS verb.

$$local\_d(\vec{p}) = \underset{i \le i \le n}{Max}(sim(\vec{p}, \vec{m_i})) \quad (14)$$

where n is the number of STMs for the pseudo trigger mention p and $sim(\vec{p}, \vec{m_i})$ is the cosine similarity between p and its STM m_i.

On the other hand, the global discrimination comes from the probability of a pseudo trigger mention belonging to the set of true trigger mentions in the training set. While the local discrimination measures the distance between a pseudo trigger mention and those STMs, the global discrimination calculates the distance between a pseudo trigger mention and all the true trigger mentions. In this paper, we use the probability from the event type determiner to calculate the global discrimination:

$$global\_d(\vec{p}) = \underset{1 \le i \le m}{Max}(p(R_i \mid \vec{p})) \quad (15)$$

where m is the number of event types, R_i is the type of a particular event and $p(R_i \mid \vec{p})$ is the probability from the event type determiner.

Given local and global discrimination, the final *discrimination* is calculated via linear interpolation.

$$\text{discrimination}(\vec{p}) = 1 - (\alpha * global\_d(\vec{p}) + (1-\alpha) * local\_d(\vec{p})) \quad (16)$$

where coefficient α ($0 \le \alpha \le 1$) is fine-tuned to 0.75 on the development set and all trigger mentions whose discrimination values are lower than 0.1 are filtered out.

5 Experimentation and discussion

In this section, we evaluate our trigger filtering mechanisms and joint model in Chinese trigger identification and its application to overall Chinese event extraction.

5.1 Experimental setting

For fair comparison, we use the state-of-the-art Chinese event extraction system, as described in Li et al. (2012), as our baseline[7], which consists of four typical components, trigger identification, event type determination, argument identification and argument role determination, and works in a pipeline way. During testing, each word in the test set is first scanned for instances of known triggers from the training set and then scanned by employing the compositional semantics inside Chinese triggers to infer instances of unknown triggers. When an instance is found, the trigger identifier is applied to distinguish those true trigger mentions from pseudo ones. If true, the event type determiner is then applied to recognize its event type. For any entity mention in a sentence which is identified as an event, the argument identifier is employed to assign its possible arguments afterwards. Finally, the argument role determiner is introduced to assign a role to each argument.

Besides, we adopt the same experimental setting as Li et al. (2012) and all the evaluations are done on the ACE 2005 Chinese corpus (only the training data is available), which contains 633 Chinese documents annotated with 8 predefined event types and 33 predefined event subtypes. Similar to previous studies, we treat these subtypes simply as 33 separate event types and do not

[7] To simplify the experiments, the baseline only contains compositional semantics in Li et al. (2012).

consider the hierarchical structure among them. Particularly, we randomly select 567 documents as the training set and the remaining 66 documents as the test set. Besides, we reserve 33 documents in the training set as the development set and follow the setting of ACE diagnostic tasks and use the ground truth entities, times and values for our training and testing. As for evaluation, we also follow the standards as defined in Li et al (2012):

- A trigger is **correctly** identified if its position in the document matches a reference trigger;
- An event type is **correctly** determined if the trigger's event type and position in the document match a reference trigger;
- An argument is **correctly** identified if its involved event type and position in the document match any of the reference argument mentions;
- An argument role is **correctly** determined if its involved event type, position in the document, and role match any of the reference argument mentions.

Finally, all the sentences in the corpus are divided into words using a Chinese word segmentation tool (ICTCLAS[8]) with all entities annotated in the corpus kept. Besides, we use Berkeley Parser[9] and Stanford Parser[10] to create the constituent and dependency parse trees and employ the ME model[11] to train individual component classifiers.

5.2 Trigger filtering

Table 4 shows the impact of the three trigger filtering mechanisms in Chinese event extraction on the held-out test set. From Table 4, we can find out that our trigger filtering mechanisms enhance the F1-measures of trigger identification, event type determination, argument identification and argument role determination by 2.5, 2.7, 2.6 and 2.3 units, respectively. It justifies the effectiveness of our trigger filtering mechanisms in addressing the low quality of the ACE 2005 Chinese corpus.

Performance / System	Trigger identification			Event type determination			Argument identification			Argument role determination		
	P%	R%	F1	P%	R%	F1	P%	R%	F1	P%	R%	F1
Baseline	73.5	62.1	67.4	70.2	59.1	64.2	58.0	48.9	53.0	54.7	44.5	49.1
+DepInference	75.4	61.9	68.0	71.9	59.1	64.9	59.8	48.9	53.8	56.1	44.5	49.6
+D&C	**76.7**	62.9	69.2	**73.1**	59.9	65.6	**60.7**	49.7	54.7	**57.2**	45.4	50.6
+L&G	75.0	**65.4**	**69.9**	71.6	**62.7**	**66.9**	59.5	**52.1**	**55.6**	56.1	**47.5**	**51.4**

TABLE 4 – Contribution of trigger filtering to Chinese event extraction (incremental)

Detailed analysis shows that

- The dependency-based inference (**DepInference**) filters out 8.5% of candidate trigger mentions and 98.8% of them are pseudo ones. As a result, Table 4 shows that this inference improves the precision by 1.9 units for trigger identification with only a slight loss of 0.2 units in the recall. Given the fact that only 20% trigger mentions are single-character words

[8] http://ictclas.org/
[9] http://code.google.com/p/berkeleyparser/
[10] http://nlp.stanford.edu/software/lex-parser.shtml
[11] http://mallet.cs.umass.edu/

and some of them can be distinguished by the trigger identifier, this justifies the effectiveness of the dependency-based inference in filtering out those pseudo single-character trigger mentions.

- While the divide-and-conquer mechanism **(D&C)** filters out 20.3% of candidate trigger mentions, it is surprising that less than 6% of filtered trigger mentions are true ones. As a result, the divide-and-conquer mechanism much improves the F1-measure, precision and recall by 1.3, 1.0 and 1.2 units respectively. Our exploration also shows that our two simple patterns can recover almost 40% of the filtered true trigger mentions.
- The local and global discrimination (**L&D**) improves the recall by 2.5 units with a loss of 1.7 units in precision. When coefficient α is fine-tuned to 0.75 and the threshold of $\text{discrimination}(\vec{p})$ is fine-tuned to 0.1, 5.8% of pseudo trigger mentions are filtered out, in which almost 80% of them are un-annotated true trigger mentions.

5.3 Joint modeling

Table 5 shows the contribution of both Trigger Filtering (**TF**) and Joint Modeling (**JM**) of trigger identification and event type determination to overall Chinese event extraction on the held-out test set. Table 5 indicates that our approach can improve the F1-measures of trigger identification, event type determination, argument identification and argument role determination (i.e. overall event extraction) by 5.7, 6.0, 5.1 and 4.8 units, respectively, largely due to the dramatic increase in recall of 9.8, 9.8, 8.3 and 7.1 units respectively.

Performance / System	Trigger identification			Event type determination			Argument identification			Argument role determination		
	P%	R%	F1	P%	R%	F1	P%	R%	F1	P%	R%	F1
CRF	**83.7**	43.3	57.1									
Baseline	73.5	62.1	67.4	70.2	59.1	64.2	58.0	48.9	53.0	54.7	44.5	49.1
+JM(w/o CRF)	73.1	68.1	70.5	69.9	65.1	67.4	57.8	53.9	55.8	54.6	49.1	51.7
+JM(w/ CRF)	73.0	70.0	71.5	69.9	67.0	68.4	57.8	55.6	56.7	54.6	50.6	52.5
+TF+JM(w/ CRF)	74.4	**71.9**	**73.1**	**71.4**	**68.9**	**70.2**	**59.1**	**57.2**	**58.1**	**55.8**	**52.1**	**53.9**

Table 5 – Contribution of joint modeling to Chinese event extraction

Table 5 also shows that

- For trigger identification, the ILP-based joint model (w/o CRF) improves the F1-measure by 3.1 units due to a big gain of 6.0 units in recall and a small loss of 0.4 units in precision. This result indicates that event type determination can much help trigger identification to improve its performance. This justifies the effectiveness of our ILP-based joint model. As for the loss in precision, it's not surprising that more pseudo trigger mentions tend to be wrongly recognized as true ones since our goal of various constraints in the ILP-based joint model is to identify those true trigger mentions as many as possible.
- Further inclusion of the CRF model as a constraint in the joint model improves the F1-measure of trigger identification by 4.1 units due to a big gain of 7.9 units in recall and a small loss of 0.5 units in precision. Our experimentation also shows that the CRF model is much complementary to the ME model in trigger identification with a high precision of 83.7

units and a low recall of 43.3 units, and the new constraint helps bring back 1.9% of true trigger mentions.

- We also apply the postprocessing mechanism of discourse consistency in Li et al. (2012) to both the baseline and our approach, and their improvement of F1-measues in trigger identification are 3.1 units and 2.4 units respectively. The reason for the loss of our approach is that our three trigger filtering mechanisms reduce the probability of the consistency in a discourse for those filtered pseudo trigger mentions.
- Finally, our trigger filtering schema and joint modeling of trigger identification and event type determination together significantly improve the recall for all of four components in Chinese event extraction with a decent gain in precision.

5.4 Discussion

From Table 5, we can also find out that the performance gaps between trigger identification and event type determination are rather small in all settings (2.9~3.2 units in F1-measures). The fact is that, even if we just assign the type with the highest prior probability to all true trigger mentions, the accuracy can still reach more than 90%. This indicates the importance of trigger identification in overall Chinese event extraction.

Normally in a pipeline system, the improvement in event type determination is always lower than that in trigger identification due to the pipeline nature (i.e. propagated errors from the upstream processes). However, Table 5 shows that our improvement in F1-measure for event type determination is higher than that for trigger identification. This is due to joint modeling of these two components in well capturing the interaction between them.

Conclusion

In order to address the special characteristics of Chinese event extraction, this paper presents a joint model to better integrate trigger identification and event type determination. Besides, several trigger filtering mechanisms are proposed to reduce the influence of those un-annotated true trigger mentions in the corpus as many as possible. The experimental results show that our approach can significantly improve the performance of Chinese trigger identification and overall Chinese event extraction.

Besides those un-annotated true trigger mentions, which much encumber the performance of trigger identification and overall event extraction, we find that 9.7% of the pseudo trigger mentions in the ACE 2005 Chinese corpus are actually true ones. Therefore, a natural extension of this work is to explore some effective methods to recover those pseudo-annotated true trigger mentions. Moreover, encouraged by the success of the ILP-based joint model, we will further explore more on this joint model and more effective joint models to event extraction.

Acknowledgments

The authors would like to thank three anonymous reviewers for their comments on this paper. This research was supported by the National Natural Science Foundation of China under Grant Nos. 61070123, 61272260 and 61273320, the National 863 Project of China under Grant No. 2012AA011102.

REFERENCES

Ahn, D. (2006). The stages of event extraction. In *Proceedings of the Workshop on Annotating*

and Reasoning about Time and Events (ARTE 2006), pages 1-8.

Barzilay, R. and Lapata, M. (2006). Aggregation via set partitioning for natural language generation. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics (NAACL 2006)*, pages 359-366.

Chen, Z. and Ji, H. (2009a). Can one language bootstrap the other: a case study on event extraction. In *Proceedings of the NAACL HLT 2009 Workshop on Semi-Supervised Learning for Natural Language Processing (SemiSupLearn 2009)*, pages 66-74.

Chen, Z. and Ji, H. (2009b). Language specific issue and feature exploration in Chinese event extraction. In *Proceedings of Human Language Technologies: the 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics* (NAACL 2009), pages 209-212.

Denis, P. and Baldridge, J. (2007). Joint determination of anaphoricity and coreference resolution using integer programming. In *Proceedings of Human Language Technologies: The Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL-HLT 2007)*, pages 236–243.

Do, Q. X., Lu, W., and Roth, D. (2012). Joint inference for event timeline construction. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL 2012)*, pages 677-687.

Finkel, J., Grenager, T. and Manning, C. (2005). Incorporating non-local information into information extraction systems by Gibbs sampling. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics (ACL 2005)*, pages 363-370.

Finkel, J. and Manning, C. (2009). Nested named entity recognition. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing (EMNLP 2009)*, pages 141-150.

Fu, J. F., Liu, Z. T., Zhong, Z. M. and Shan, J. F. (2010). Chinese event extraction based on feature weighting. *Information Technology Journal*, 9: 184-187.

Gupta, P. and Ji, H. (2009). Predicting unknown time arguments based on cross-event propagation. In *Proceedings of the ACL-IJCNLP 2009 Conference Short Papers* (ACLShort 2009), pages 369-272.

Grishman, R., Westbrook, D. and Meyers, A. (2005). NYU's English ACE 2005 system description. In *Proceedings of ACE 2005 Evaluation Workshop (ACE workshop 2005)*.

Hardy, H., Kanchakouskaya, V. and Strzalkowski, T. (2006). Automatic event classification using surface text features. In *Proceedings of the Twenty-first National Conference on Artificial Intelligence Workshop on Event Extraction and Synthesis (AAAI workshop 2006)*, pages 36-41.

Hong, Y., Zhang, J., Ma, B., Yao, J. M., Zhou, G. G. and Zhu, Q. M. (2011). Using cross-entity inference to improve event extraction. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL-HLT 2011)*, pages 1127-1136.

Iida, R. and Poesio, M. (2011). A cross-lingual ILP solution to zero anaphora resolution. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL-HLT 2011)*, pages 804-813.

Ji, H. (2009). Cross-lingual predicate cluster acquisition to improve bilingual event extraction by

inductive learning. In *Proceedings of the Workshop on Unsupervised and Minimally Supervised Learning of Lexical Semantics (UMSLLS 2009)*, pages 27-35.

Ji, H. and Grishman, R. (2008). Refining event extraction through cross-document inference. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL-HLT 2008)*, pages 254-262.

Landeghem, S. V., Saeys Y., Baets, B. D. and Peer, Y. V. (2009). Analyzing text in search of bio-molecular events: a high-precision machine learning framework. In *Proceedings of the Workshop on Current Trends in Biomedical Natural Language Processing: Shared Task (BioNLP 2009)*, pages 128-136.

Landeghem, S. V., Pyysalo, S., Ohta, T. and Peer, Y. V. (2010). Integration of static relations to enhance event extraction from text. In *Proceedings of the 2010 Workshop on Biomedical Natural Language Processing (BioNLP 2010)*, pages 144-152.

Li, J. H., Zhou, G. D. and Ng, H. T. (2010). Joint syntactic and semantic parsing of Chinese. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics (ACL 2010)*, pages 1108-1117.

Li, P. F., Zhou G. D., Zhu Q. M. and Hou L. B. (2012). Employing compositional semantics and discourse consistency in Chinese event extraction. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL 2012)*, pages 1006-1016.

Liao, S. S. and Grishman, R. (2010). Using document level cross-event inference to improve event extraction. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics (ACL 2010)*, pages 789-797.

Liao, S. S. and Grishman, R. (2011). Using prediction from sentential scope to build a pseudo co-testing learner for event extraction. In *Proceedings of the 5th International Joint Conference on Natural Language Processing (IJCNLP 2011)*, pages 714–722.

Llorens, H., Saquete, E., Navarro-Colorado, B. (2012). Applying semantic knowledge to the automatic processing of temporal expressions and events. *Information Processing & Management*.

Lu, W. and Roth, D. (2012). Automatic event extraction with structured preference modeling. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics* (ACL 2012), pages 835–844.

Maslennikov, M. and Chua, T. (2007). A multi resolution framework for information extraction from free text. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics (ACL 2007)*, pages 592-599.

Patwardhan, S. and Riloff, E. (2007). Effective information extraction with semantic affinity patterns and relevant regions. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-coNLL 2007)*, pages 717-727.

Patwardhan, S. and Riloff, E. (2009). A unified model of phrasal and sentential evidence for information extraction. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing (EMNLP 2009)*, pages 151-160.

Poon, H. and Vanderwende, L. (2010). Joint inference for knowledge extraction from biomedical literature. In *Proceedings of Human Language Technologies: the 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL-HLT 2010)*,

pages 813-821.

Qin, B., Zhao, Y. Y., Ding, X., Liu, T. and Zhai, G. F. (2011). Event type recognition based on trigger expansion. *Tsinghua Science and Technology*. 15(3): 251-258.

Riedel, S., Chun, H. W., Takagi, T. and Tsujii, J. (2009). A Markov logic approach to bio-molecular event extraction. In *Proceedings of the Workshop on Current Trends in Biomedical Natural Language Processing: Shared Task (BioNLP 2009)*, pages 41-49.

Riedel, S. and McCallum, A. (2011). Fast and robust joint models for biomedical event extraction. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing (EMNLP 2011)*, pages 1-12.

Roth, D. and Yih, W. (2004). A linear programming formulation for global inference in natural language tasks. In *Proceedings of the 8th Conference on Natural Language Learning (CoNLL 2004)*, pages 1-8.

Tan, H., Zhao, T., Zheng, J. (2008). Identification of Chinese event and their argument roles. In *Proceedings of the 2008 IEEE 8th International Conference on Computer and Information Technology Workshops (CITWORKSHOPS 2008)*, pages 14-19.

Integrating Surface and Abstract Features for Robust Cross-Domain Chinese Word Segmentation

LI Xiaoqing[1], WANG Kun[1], ZONG Chengqing[1] and SU Keh-Yih[2]
(1) National Laboratory of Pattern Recognition, Institute of Automation,
Chinese Academy of Sciences, Beijing, China
(2) Behavior Design Corporation, Taiwan
{xqli, kunwang, cqzong}@nlpr.ia.ac.cn, kysu@bdc.com.tw

ABSTRACT

Current character-based approaches are not robust for cross domain Chinese word segmentation. In this paper, we alleviate this problem by deriving a novel enhanced character-based generative model with a new abstract aggregate candidate-feature, which indicates if the given candidate prefers the corresponding position-tag of the longest dictionary matching word. Since the distribution of the proposed feature is invariant across domains, our model thus possesses better generalization ability. Open tests on CIPS-SIGHAN-2010 show that the enhanced generative model achieves robust cross-domain performance for various OOV coverage rates and obtains the best performance on three out of four domains. The enhanced generative model is then further integrated with a discriminative model which also utilizes dictionary information. This integrated model is shown to be either superior or comparable to all other models reported in the literature on every domain of this task.

KEYWORDS : Chinese Word Segmentation, Cross-Domain, Robust Feature, Utilize Dictionary

Proceedings of COLING 2012: Technical Papers, pages 1653–1670,
COLING 2012, Mumbai, December 2012.

1 Introduction

As words are the basic units for text analysis, Chinese word segmentation (CWS) is critical for many Chinese NLP tasks such as parsing. However, current state-of-the-art character-based approaches fail to give robust performance for cross-domain tests (Gao and Vogel, 2010; Huang et al., 2010; Jiang and Dong, 2010; Wang et al., 2012, Sun et al., 2012), despite their acceptable performance for in-domain tests. For example, with the same PKU-News training corpus, the best system in SIGHAN-2005 (Emerson, 2005) achieved a high in-domain performance (96.9% in F-score) in open[1] tests, while the best cross-domain performance for the medicine domain only achieved 93.8%[2] in CIPS-SIGHAN-2010 (Zhao and Liu, 2010) open tests.

Their poor performances result not only from the fact that out-domains have higher out-of-vocabulary (OOV) rates (please refer Table 1 in Section 3.1), but also from the fact that various domains frequently possess different tag distributions for the same character, especially for those out-domain OOV words (which are also technical terms most of times). For example, "酸" (acid) is a typical suffix of medical terms, such as "尿酸" (uric acid) and "氨基酸" (amino acid), and is frequently tagged as "E" in the medicine domain. However, it is usually a single-character word which means 'sour' in general text, and should be tagged as "S". As a result, those surface features (such as character n-grams) adopted in the approaches mentioned above have more difficulty in identifying the correct position-tag, which is a member of {**B**eginning, **M**iddle, **E**nd, **S**ingleton}(Xue, 2003), of a character within an OOV word in this case. Since many technical OOV words appear in the out-domain text, the performance thus degrades sharply. For instance, when we test the main stream character-based discriminative approach for medicine domain in CIPS-SIGHAN-2010 contest, 27.7% of the wrong segmented words are OOV terms.

On the other hand, unlike surface features, the distribution of the proposed abstract feature, which checks if the given candidate prefers the corresponding position-tag of the longest dictionary matching word, is almost invariant across different domains. Enhanced discriminative approaches (Low et al., 2005; Zhao et al., 2010) which adopt this abstract feature thus show better generalization capability in our cross-domain tests.

To adopt the above matching-longest-word feature, a dictionary is required. It is well known that the domain dictionary provides direct and reliable hints for deciding the position-tags of characters within a covered OOV word. Furthermore, unlike named entity, technical terms of a specific domain usually can be considerably covered by a domain dictionary. Since domain dictionaries are frequently available for NLP related projects (e.g., technical manual translation), they can thus provide big help in real applications.

However, the above dictionary related feature utilized in discriminative approaches (Low et al., 2005; Zhao et al., 2010) cannot be directly adopted by a generative model[3]. Therefore, we propose a new tag-matching-status feature for checking if the selected position-tag matches the longest dictionary-matching-word, and derive a novel enhanced character-based generative model. The proposed feature not only induces an additional probabilistic factor but also possesses richer information in comparison with the one adopted in previous works.

[1] Unlike close test, the open test can use any language resource, not restricted to training data only.
[2] 93.8% only corresponds to 23.7% sentence accuracy rate (average 23.3 words per sentence in the medicine corpus), evaluated from its associated 94.0% recall-rate, while 96.9% corresponds to 46.9% sentence accuracy.
[3] It is required by the integrated approach given at Section 4.2.

Several factors that might affect the performance of the new model are studied in this paper: including the context information, the OOV coverage rate of the dictionary, and the weight of the new factor in the model. We evaluated our final system on the CIPS-SIGHAN-2010 Bakeoff data. The obtained results not only convincingly demonstrate the effectiveness of the proposed model for cross-domain CWS, but also achieve the best performance on 3 out of 4 domains in the open test. Afterwards, the proposed enhanced generative model is integrated with another enhanced discriminative model (Low et al., 2005) to further improve the performance, and achieves the best performance on all the tested corpora.

The remainder of this paper is organized as: Section 2 discusses how to incorporate dictionary information and section 3 describes the proposed models. Empirical results and error analysis are presented in section 4 and 5. Section 6 reviews the related work.

2 Dictionary related features

2.1 Word-ID or Word-Matching-Indicator?

Given a dictionary, there are two kinds of features that can be utilized: word-ID, which are binary features that fire only when the word matches one specific word entry, and Word-Matching-Indicator (e.g. TM defined in Section 2.3), which checks the relationship between the assigned position tag of the current character and the dictionary words within local context. Since the statistics of OOV words can never be learnt from the training corpus, the approaches that adopt word-ID as features (Zhang and Clark, 2007; Sun, 2010; Zhang and Clark, 2011) cannot really utilize the information of the OOV words kept in the dictionary. On the contrary, the word-matching-indicator is applicable for both IV and OOV words kept in the dictionary. This feature thus provides valuable information for those OOV words covered by the dictionary. Therefore, based on the positions of those dictionary matching words, two dictionary-related features (i.e., Dictionary Coverage Status and Tag Matching Status, to be specified later) are proposed in this paper, and they will be incorporated into the character-based generative model.

2.2 Dictionary Coverage Status

Let c_i be the i-th character in a given sentence. To check whether there are ambiguities with those dictionary matching words at c_i (and what kind of ambiguities it has), we propose the Dictionary Coverage Status feature, which is a member of {No-Dictionary-Word, No-Ambiguity, Crossed-Ambiguity[4], Included-Ambiguity, Mixed-Ambiguity} that are defined below. This status depends only on the given sentence and the dictionary, and is irrelevant to the position tag assigned to the character. Let D be the given dictionary which only contains multi-character words, and $c_{[i:j]}$ denotes the string from c_i to c_j (including c_j), then the conditions for "Included-Ambiguity" and "Crossed-Ambiguity" are defined below.

(A) Conditions for Included-Ambiguity (IA):

(1) Both c_i and c_{i+1} will be assigned "IA" if they meet the following condition (Figure 1(a)):

$$\exists j,l>0,k\geq j:\{c_{[i-j:i]},c_{[i-k:i+l]}\}\subseteq D;$$

(2) Both c_{i-1} and c_i will be assigned "IA" if they meet the following condition (Figure 1(b)):

$$\exists j,k>0,l\geq j:\{c_{[i:i+j]},c_{[i-k:i+l]}\}\subseteq D.$$

[4] Please note that ambiguity status is traditionally defined on words, but ours is defined on characters.

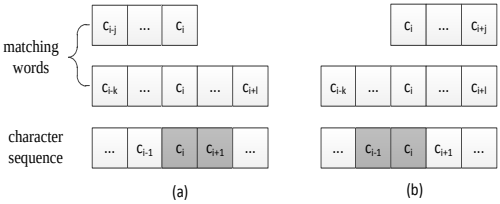


FIGURE 1 – Cases for Included-Ambiguous characters (marked in grey)

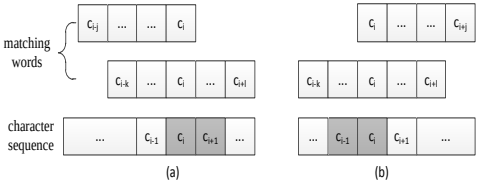


FIGURE 2 – Cases for Crossed-Ambiguous characters (marked in grey)

(B) Conditions for Crossed-Ambiguity (CA):

(1) Both c_i and c_{i+1} will be assigned "CA" if they meet the following condition (see Figure 2(a)):

$$\exists j,l > 0, 0 \le k < j : \{c_{[i-j:i]}, c_{[i-k:i+l]}\} \subseteq D ;$$

(2) Both c_{i-1} and c_i will be assigned "CA" if they meet the following condition (Figure 2(b)):

$$\exists j,k > 0, 0 \le l < j : \{c_{[i:i+j]}, c_{[i-k:i+l]}\} \subseteq D .$$

Then Dictionary Coverage Status at c_i (denoted by DC_i) can be decided as follows:

$$DC_i = \begin{cases} \text{No-Dictionary-Word, if no matching word is found;} \\ \text{Included-Ambiguity, if only (A) is satisfied;} \\ \text{Crossed-Ambiguity, if only (B) is satisfied;} \\ \text{Mixed-Ambiguity, if both (A) and (B) are satisfied;} \\ \text{No-Ambiguity, otherwise.} \end{cases}$$

The above definition implicitly implies that a character which possesses the same position-tag for all associated dictionary matching words will be assigned "No-Ambiguity".

For example, given a character sequence "大学生物" (university biology) and a set of dictionary-matching-words {"大学" (university), "大学生" (undergraduate)}, for characters '学' and "生", condition (A.1) is satisfied , but condition (B) is not; therefore, DC_2 and DC_3 should be set to "Included-Ambiguity". On the other hand, if the dictionary-matching-words are {"大学生", "生物" (biology)}, then condition (B.1) is satisfied, but condition (A) is not; DC_2 and DC_3 thus

should be set to "Crossed-Ambiguity". However, if we have all the three matching words {"大学", "大学生", "生物"}, then both condition (A.1) and condition (B.1) are satisfied; therefore, DC_2 and DC_3 should be set to "Mixed-Ambiguity" in this case. Furthermore, if the matching words are {"大学", "生物"}, then DC_2 and DC_3 would be "No-Ambiguity". Last, DC_1 and DC_4 are always "No-Ambiguity" for all above three different cases.

2.3 Tag Matching Status

To indicate the relationship between the tag assigned to c_i and those dictionary matching words which cover c_i, we introduce the Tag Matching Status feature (the abstract aggregate candidate-feature, and is abbreviated as **TM** from now on), which is a member of {Following-Longest-Word, Only-Following-Shorter-Word, Not-Following-Any-Word, Inapplicable} that are defined below. Denote the set of dictionary matching words that begin with c_i as $D_B = \{c_{[i:j]} \mid c_{[i:j]} \in D\}$, the set of dictionary matching words that enclose c_i as $D_M = \{c_{[j:k]} \mid c_{[j:k]} \in D, j < i < k\}$, and the set of dictionary matching words that end with c_i as $D_E = \{c_{[j:i]} \mid c_{[j:i]} \in D\}$. If c_i is tagged as t_i, then TM_i can be decided as follows:

(1) If $D_B \bigcup D_M \bigcup D_E = \varnothing$, which indicates that this character is not covered by any dictionary word, then TM_i is set to "Inapplicable".

(2) If $D_{t_i} = \varnothing$, and $D_B \bigcup D_M \bigcup D_E \neq \varnothing$ (where D_{t_i} is the set of dictionary matching words corresponding to t_i; for example, D_{t_i} will be D_B, if $t_i = B$. Please note that $D_S = \varnothing$, since the adopted dictionary only contains multi-character words), which indicates that the assigned tag does not follow any dictionary matching word, then TM_i is set to "Not-Following-Any-Word";

(3) If $\forall w \in (D_B \bigcup D_M \bigcup D_E), \exists w' \in D_{t_i} : \mathrm{len}(w') \geq \mathrm{len}(w)$, then TM_i is set to "Following-Longest-Word". It indicates that the assigned tag matches the corresponding position-tag of the longest dictionary matching word at that character;

(4) Otherwise, TM_i is set to "Only-Following-Shorter-Word". It indicates that the assigned tag does not match the corresponding position-tag of the longest dictionary matching word at that character, but matches that of some shorter words.

For example, when we consider the second character '学' in the sequence "大学生" and assume that the dictionary matching words are {"大学", "大学生"}: if the tag assigned to '学' is "M", then TM_2 will be "Following-Longest-Word"; if it is "E", then TM_2 will be "Only-Following-Shorter-Word"; if it is "B" or "S", TM_2 would be "Not-Following-Any-Word". Therefore, this candidate-feature is associated with each candidate of the position-tag. However, if no dictionary word covers this character, then TM_2 will be set to "Inapplicable" regardless of which tag is assigned to '学' (i.e., we do not want to disturb the original model in this case).

3 Proposed models

3.1 Enhanced generative model

Wang et al. (2009) proposed a character-based generative model for CWS, which is able to handle the dependency of character-bigrams within words and thus give a good balance for the performance of IV words and OOV words. Their approach adopts the character-tag-pair trigram model, and obtains the desired position-tag sequence $\tilde{t}_1^n$ as follows:

$$\bar{t}_1^n \equiv \arg\max_{t_1^n} \prod_{i=1}^{n} P([c,t]_i \mid [c,t]_{i-2}^{i-1}) \quad (1)$$

where $[c,t]_1^n$ is the associated character-tag-pair sequence for the given character sequence c_1^n.

To alleviate the data sparseness problem, we pre-convert the given character string into its corresponding unit string before segmentation, where the unit denotes a FN-String (which is a string mixed with foreign/Arabic/selected-punctuations [5] characters, such as "HTML5", "www.google.com", etc.), a CN-String (which is a string of **C**hinese **N**umbers consisting of Chinese digit-characters such as "六五七七二" (65772), etc.), or a single Chinese character. Therefore, a unit is either a Chinese character or a foreign/numerical expression as defined above, and is represented by either a Chinese-Character-ID or a Meta-Type-ID (i.e., FN-type or CN-type) in the model.

After the character string has been pre-converted into the unit string, we incorporate the dictionary related features proposed in Section 2 into the generative model and re-formulate it as follows:

$$\begin{aligned} \bar{t}_1^n &\equiv \arg\max_{t_1^n} P(t_1^n, TM_1^n \mid u_1^n, DW_1^n) \\ &= \arg\max_{t_1^n} \left[\frac{P([u,t,TM]_1^n \mid DW_1^n)}{P(u_1^n \mid DW_1^n)} \right] \\ &= \arg\max_{t_1^n} P([u,t,TM]_1^n \mid DW_1^n) \end{aligned} \quad (2)$$

where DW_i denotes the set of dictionary words that cover u_i (i-th unit). $P([u,t,TM]_1^n \mid DW_1^n)$ is then approximated by $\prod_{i=1}^{n} P([u,t,TM]_i \mid [u,t,TM]_{i-2}^{i-1}, DW_i)$ and its associated factor is further derived as follows:

$$\begin{aligned} & P([u,t,TM]_i \mid [u,t,TM]_{i-2}^{i-1}, DW_i) \\ &= P(TM_i \mid [u,t]_{i-2}^{i}, TM_{i-2}^{i-1}, DW_i) \times P([u,t]_i \mid [u,t,TM]_{i-2}^{i-1}, DW_i) \\ &\approx P(TM_i \mid MWL_i, DC_i, u_{i-2}^{i}) \times P([u,t]_i \mid [u,t]_{i-2}^{i-1}) \end{aligned} \quad (3)$$

where MWL_i denotes the Maximum Word Length of the words that cover u_i, DC_i (Dictionary Coverage Status) and TM_i (Tag Matching Status) are defined at Section 2.2 and Section 2.3, respectively. The first term $P(TM_i \mid MWL_i, DC_i, u_{i-2}^{i})$ is the tag matching factor, which is mainly introduced to give guidance in the case that the second term $P([u,t]_i \mid [u,t]_{i-2}^{i-1})$ (the original generative model) cannot give reliable prediction when the associated character-tag bigram is unseen in the training corpus.

Equation (3) weighs the tag matching factor and the character-tag trigram factor equally. However, it is reasonable to expect that they should be weighted differently according to their contribution. We thus combine these two factors via log-linear interpolation, which is shown as follows:

[5] Selected punctuations include those members from {'+', '-', '*', '/', '.', '%', '@'}.

$$\begin{aligned}\text{Score}(t_i) = \alpha \times \log P([u,t]_i \mid [u,t]_{i-2}^{i-1}) \\ + (1-\alpha) \log P(TM_i \mid MWL_i, DC_i, u_{i-2}^{i})\end{aligned} \quad (4)$$

Where α is the weighting coefficient to be decided from the development set, and $0 < \alpha < 1.0$.

3.2 Enhanced integrated model

To incorporate the dictionary information into the discriminative approach, Low et al. (2005) added two additional features (which are features (d) and (e) in the following list) to the widely adopted primitive templates described in (Ng and Low, 2004), and used them to enhance the original discriminative model (Xue, 2003):

(a) C_n $(n = -2, -1, 0, 1, 2)$; (d) MWL_0, t'_0;
(b) $C_n C_{n+1}$ $(n = -2, -1, 0, 1)$; (e) $C_n t'_0$ $(n = -1, 0, 1)$.
(c) $C_{-1} C_1$;

Let W denote the longest dictionary word that covers c_0, then MWL_0 denotes the length of W, and t'_0 denotes the corresponding tag of c_0 in W.

Since the enhanced generative model cannot utilize the features from future context, which is a common drawback of generative approaches (Wang et al., 2010), following the approach of (Wang et al., 2011), we further integrate the enhanced generative model with the above enhanced discriminative model via log-linear interpolation, shown as follows:

$$\begin{aligned}\text{Score}(t_i) = \beta \times [\alpha \times \log P([u,t]_i \mid [u,t]_{i-2}^{i-1}) \\ + (1-\alpha) \log P(TM_i \mid MWL_i, DC_i, u_{i-2}^{i})] \\ + (1-\beta) \times \log(P(t_i \mid u_{i-2}^{i+2}, MWL_i, t'_i)\end{aligned} \quad (5)$$

Where α and β are two weighting coefficients to be decided from the development set, and $0 < \alpha, \beta < 1.0$.

4 Experiments

All experiments are conducted on the corpora provided by SIGHAN-2005 (Emerson, 2005) and CIPS-SIGHAN-2010[6] (Zhao and Liu, 2010). Both the in-domain test and cross-domain tests are trained on PKU-News[7] from CIPS-SIGHAN-2010. The PKU-News testing corpus of SIGHAN-2005 is adopted for in-domain test, while the corpora of CIPS-SIGHAN-2010 are used for cross-domain tests. There are four different domains: Literature (denoted as **Lit.**), Computer (**Cmp.**), Medicine (**Med.**) and Finance (**Fin.**). To obtain the weights of different factors in Equations (4) and (5), we randomly selected 1% from the original training corpus as the development set, and

[6] CIPS-SIGHAN-2010 is the first cross-domain Chinese Word Segmentation (CWS) bake-off competition which involves 18 systems. To our knowledge, this data-set is the most well-known and widely adopted (also publically available) one for cross-domain CWS test.
[7] The PKU-News training data of CIPS-SIGHAN-2010 is the same with the PKU training data of SIGHAN-2005.

regard the remaining part as the new training set. The updated corpora statistics are shown at TABLE 1.

Corpora	Domain	Characters	Tokens	Word Types	OOV Rate
Training	News	1,820,456	1,109,947	55,303	N/A
Develop.	News	99,381	60,585	11,216	0.028
Testing	News	172,733	104,372	13,148	0.058
	Lit.	50,637	35,736	6,364	0.069
	Cmp.	53,382	35,319	4,150	0.152
	Med.	50,969	31,490	5,076	0.110
	Fin.	53,253	33,028	4,918	0.087

TABLE 1 – Corpus statistics for CIPS-SIGHAN-2010

Besides, the SRI Language Modelling Toolkit (SRILM)[8] (Stolcke, 2002) is used to train $P([u,t]_i | [u,t]_{i-2}^{i-1})$ with the modified Kneser-Ney smoothing method (Chen and Goodman, 1996). Also, the Factored Language Model in SRILM is adopted to train $P_{BF}(TM_i | MWL_i, DC_i, u_{i-2}^i)$, and $P_{BF}(TM_i | MWL_i, DC_i, u_{i-2}^i)$ sequentially back-off to $P_{GT}(TM_i | MWL_i, DC_i)$, where the subscripts "BF" and "GT" denote back-off and Good-Turing estimations, respectively. For the discriminative approach, the ME Package[9] provided by Zhang Le is adopted for training. Last, all adjustable weights are first selected only based on the development set. Those obtained optimal values are then fixed and applied to the testing-set.

4.1 Enhanced generative model

4.1.1 Effect of having character context

It is observed that tag matching status is less reliable when there are several dictionary matching words for the given character. Therefore, the character context is introduced to help disambiguate them. To show the influence of character context information, we test two different enhanced generative models. The first one adopts the factor $P(TM_i | MWL_i, DC_i)$ (denoted as G1), and the second one adopts the factor $P(TM_i | MWL_i, DC_i, u_{i-2}^i)$ (denoted as G2; with character context). In addition, the performance of the original generative model (denoted as B for baseline) is also shown for comparison. The dictionary adopted here contains all the training words and the testing words excluding the named entities (which are usually not covered by domain dictionaries).

TABLE 2 shows that character context effectively improves the performance for four out of five domains in F-score (except the Computer domain), which mainly resultes from the inconsistent segmentation criteria between the News training set and the computer testing set. For example, 18.3% of "就是" (just like) occurrences are segmented as a single word, and 81.7% of them are segmented into two single-character words "就" (just) and "是" (be) under similar context in the training set. However, this string is always treated as a single word in the Computer testing set. When the context is not considered, G1 prefers the longer word and gives correct answer for all the occurrences in the Computer corpus. While the context is considered, G2 will prefer to follow those occurrences in the training set and thus give wrong result most of the time.

[8] http://www.speech.sri.com/projects/srilm/
[9] http://homepages.inf.ed.ac.uk/lzhang10/maxent.html

Model	News	Lit.	Cmp.	Med.	Fin.	OA
B	0.952	0.928	0.929	0.904	0.950	0.939
G1	0.959	0.951	0.969	0.963	0.971	0.962
G2	0.968	0.953	0.966	0.964	0.973	0.965

TABLE 2 – Effect of adopting character context in F-score (testing-set)

Corpus	Character-Tag-Bigram	B	G1	G2
News	seen	0.018	0.030	0.016
	unseen	0.183	0.045	0.045
Lit.	seen	0.028	0.030	0.022
	unseen	0.203	0.046	0.047
Cmp.	Seen	0.026	0.017	0.015
	unseen	0.183	0.014	0.016
Med.	Seen	0.024	0.016	0.012
	unseen	0.251	0.022	0.023
Fin.	Seen	0.019	0.018	0.011
	unseen	0.155	0.015	0.017

TABLE 3 –Tagging error rates for seen/unseen cases in the testing-set

Overall, G2 outperforms B and G1 by 2.6% and 0.3%, respectively. This phenomenon can be explained by classifying the tagging errors into two groups according to whether the associated character-tag bigram is seen or not in the training set. TABLE 3 shows that the original character-tag trigram factor (i.e., B) works well when the bigram is seen, but it performs poorly when this bigram is unseen. On the other hand, G1 (without context) mainly boosts the performance for unseen cases. However, G2 (with context) also boosts the performance for seen cases. Therefore, it will not let the newly added tag-matching factor contaminate the original trigram model when associated character-tag bigrams are seen. The above observations hold for both development-set and testing-set. G2 is thus adopted for the enhanced generative model and integrated model.

4.1.2 Effect of dictionary coverage rate

Since no dictionary can cover all OOV words for real applications, we would like to know how this enhanced generative model performs under different dictionary coverage rates. We extract two dictionaries: the first one (D1) includes all the training words; and the second one (D2) contains all the OOV words in the testing set (excluding named entities). TABLE 4 gives the results for various combinations of D1 and D2 with $\alpha = 0.5$, where the first row "None" denotes that no dictionary (even D1) is adopted; also, the last column "OA" gives the overall performance of various domains (except News).

It can be seen that the improvements with the dictionary information from the training set are not obvious (None vs. D1). The reason is that the original character-tag trigram model already handles IV words well enough and the information of IV words seems redundant to this model. However, when the dictionary starts to cover OOV words, the performance rises sharply according to the OOV coverage rate. Anyway, the enhanced model always outperforms the original model even when the dictionary only covers a few OOV words.

Dict.	News	Lit.	Cmp.	Med.	Fin.	OA
None	0.952	0.928	0.929	0.904	0.950	0.928
D1	0.953	0.930	0.929	0.907	0.951	0.929
+20%D2	0.955	0.933	0.934	0.919	0.955	0.935
+40%D2	0.958	0.939	0.940	0.931	0.959	0.942
+60%D2	0.961	0.943	0.949	0.941	0.964	0.949
+80%D2	0.964	0.948	0.958	0.952	0.968	0.956
+D2	**0.968**	**0.953**	**0.966**	**0.964**	**0.973**	**0.964**

TABLE 4 – F-score versus different OOV coverage rates for the enhanced character-based generative model (testing-set). OA: overall performance of those four cross-domains. Boldface indicates the best result under each column.

Dict.	News	Lit.	Cmp.	Med.	Fin.	OA
D1	0.935	0.901	0.895	0.861	0.933	0.914
+20%D2	0.940	0.909	0.908	0.880	0.940	0.923
+40%D2	0.945	0.918	0.924	0.901	0.947	0.932
+60%D2	0.950	0.928	0.937	0.923	0.955	0.942
+80%D2	0.955	0.938	0.952	0.945	0.962	0.952
+D2	**0.961**	**0.949**	**0.967**	**0.969**	**0.969**	**0.962**

TABLE 5 – F-score versus different OOV coverage rates for the word-based trigram model

Nonetheless, not every model possesses the robustness for varying dictionary coverage rate. For example, the corresponding result of the word-based generative trigram model[10], given at TABLE 5, shows that it is quite fragile in comparison with our model. In this model, all words kept in the dictionary are used to construct the word lattice in the decoding process. Those OOV words will be treated as unseen events and given a very low score. However, it can be seen that although the results with full dictionary are satisfactory, the performance drops dramatically while the OOV coverage rate decreases. This indicates that this model is quite sensitive to those OOV words, due to its incapability of identifying OOV words beyond the dictionary. This model is thus not useful for real applications, as it is impossible to know the corresponding dictionary coverage rate in the testing set in advance. Therefore, checking the robustness of dictionary-based models for different dictionary coverage rates is important in selecting an appropriate model.

4.1.3 Effect of varying weights

The F-scores of the enhanced generative model versus various α values (the weight of $P([u,t]_i \mid [u,t]_{i-2}^{i-1})$ in Equation (4) are evaluated on the development set, and are shown in FIGURE 3. It can be seen that all the curves are flat near their peaks, which indicates that this enhanced model is not sensitive to which α value is picked. Besides, although the performance decreases when the OOV coverage rate drops, the α locations of peaks for various curves are almost the same (all around $\alpha = 0.4$). This indicates that the best α value is not sensitive to the OOV coverage rate.

[10] This well-known model adopts the form : $WSeq = \arg\max \prod_{i=1}^{m} P(w_i \mid w_{i-2}^{i-1})$ (Wang et al., 2012).

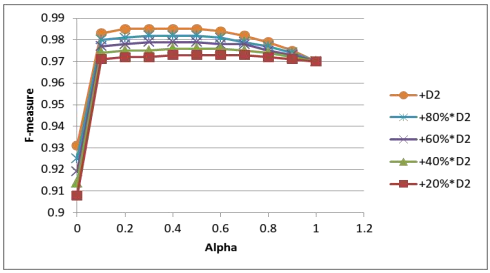


FIGURE 3 – F-score of the enhanced generative model versus various weights α on the development set

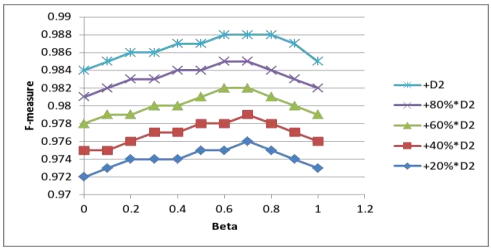


FIGURE 4 –F-score of the enhanced integrated model versus various weights β on the development set

4.2 Enhanced integrated model

Since the original generative model and the discriminative model are found to complement each other (Wang et al., 2011), it is expected that the enhanced generative model and the enhanced discriminative model (Low et al., 2005), which is re-implemented, should also complement each other. This inference is supported by the statistics that these two approaches share only 31.9% of their errors in the literature domain (similarly in other domains). Therefore, integrating these two models is expected to achieve a better result too. For the enhanced integrated model, we fix the weight α of $P([u,t]_i \mid [u,t]_{i-2}^{i-1})$ to be 0.4 (according to section 4.1.3). Afterwards, we adjust the weight of the enhanced generative model and the enhanced discriminative approach on the same development set. FIGURE 4 gives F-scores with different dictionaries versus various β values. The β locations of peaks for various curves are also almost the same (all around β =0.7) for various dictionary coverage rates. This indicates that the β weight is not sensitive to the OOV coverage rate. This figure also shows that the peak of the integrated model is robust for different dictionary coverage rates.

Last, to fairly compare different models in a more realistic condition, Table 6 shows the results of the enhanced discriminative model (denoted by ED) and the integrated model (denoted by EI) with an external dictionary (which roughly corresponds to 65% D2 coverage rate in TABLE 4, and is specified in the next section). Note that our ED result (0.968) is a little bit different from that

reported in (Low et al., 2005), which gives 0.965 F-score with a smaller dictionary that is a part of ours (see the next section). It can be seen that our enhanced integrated model achieves the best results on all five corpora.

	News	**Lit.**	**Cmp.**	**Med.**	**Fin.**
SBest	0.969	**0.955**	0.950	0.938	0.960
ED	0.968	0.951	0.946	0.938	0.961
EG	0.967	0.946	0.950	**0.944**	0.962
EI	**0.973**	**0.955**	**0.951**	**0.944**	**0.963**

TABLE 6 - F-scores on the testing sets. SBest: best results from SIGHAN 2005 (News) and CIPS-SIGHAN 2010 (other domains). Boldface indicates the best result.

4.3 Comparison with other state-of-the-art systems

To provide publically accessible dictionaries for open comparison, we combine a general dictionary downloaded from the Internet[11] and another technical dictionary extracted from Wiki[12] as our external dictionary. The first general dictionary is also adopted by (Low et al., 2005). For simplicity, we adopted the same dictionary (the union of above two dictionaries) for all five different domains. This external dictionary includes 458,165 words in total which roughly corresponds to 65% D2 coverage rate in TABLE 4. Since the external dictionary is expected to be collected by the user in real applications, dictionary words should be consistent with his/her own segmentation criterion. Therefore, to give true evaluation for reflecting the real situation, words in the dictionary are first transformed into their corresponding ones according to the same criteria adopted in various given corpora. For example, "免疫系统" (immune system) is converted into "免疫" (immune) and "系统" (system) according to the gold criterion adopted in CIPS-SIGHAN-2010.

The results of the enhanced generative model (denoted by EG) with the external dictionary and the SIGHAN best results in each domain are also given in TABLE 6. They are summarized as follows: Low et al. (2005) added the dictionary information to the discriminative approach and adopted additional corpora. They achieved the best result (0.969 F-score) on PKU News corpus in the open test of SIGHAN-2005. On the other hand, Huang et al. (2010) adopted HMM and some rules to post-process the output of the CRF discriminative approach. They achieved the best in the Literature domain in CIPS-SIGAHN-2010. Last, (Gao and Vogel, 2010) combined several classifiers with a large margin classifier and won the best on other three remaining domains. It shows our enhanced generative model achieves the best results on three out of four cross-domains, and our enhanced integrated model outperforms all the systems reported in the literature.

To further check if the difference between various models listed in Table 6 is really statistically significant, we adopt the bootstrapping technique (Koehn, 2004; Zhang et al., 2004) to conduct the significant tests. We follow the work of (Wang et al., 2010) and take the re-sampling size to be 2,000. 95% confidence interval is adopted in our tests. TABLE 7 shows that our enhanced generative model is superior to the enhanced discriminative model in overall comparison on those four cross-domains. Furthermore, the enhanced integrated model is either superior or comparable to all other models.

[11] http://ccl.pku.edu.cn/alcourse/nlp/2010/word_freq_list.rar
[12] http://dumps.wikimedia.org/zhwiki/20111017/zhwiki-20111017-all-titles-in-ns0.gz

Systems		News	Lit.	Cmp.	Med.	Fin.
A	B					
EG	SBest	<	<	~	>	~
ED	SBest	~	<	~	~	~
EG	ED	<	<	>	>	~
EI	SBest	>	~	~	>	~
	ED	>	>	>	>	~
	EG	>	>	~	~	>

TABLE 7 - Statistical significance test of F-Score among various systems. SBest: best results of the SIGHAN 2005 (News) and the CIPS-SIGHAN 2010 (others). ">" means that A is significantly better than B; "<" means A is worse than B; "~" means that they are not different.

5 Error analysis and discussion

Following the work of (Sun, 2010), we get the upper bound of our enhanced generative model (EG) and enhanced integrated model (EI) by regarding each factor as an independent model, which is shown in TABLE 8. Compared with the results in TABLE 6, we can find that there is still a large room to further improve our proposed models.

Model	News	Lit.	Cmp.	Med.	Fin.
EG	0.984	0.973	0.971	0.968	0.978
EI	0.986	0.976	0.973	0.970	0.980

TABLE 8 - F-score upper bound for EG and EI models

Furthermore, we collect and analyse the remaining errors generated by the enhanced integrated model on the Medicine corpus, which contains a large number of technical terms and is most far away from the upper bound. It is found that 66.6% (out of 1,385) of error words are related to OOV (not seen in the training-set). Among those 922 OOV errors, 758 (82.2%) of them are not covered by the dictionary and 401 (52.9%) out of 758 are technical terms. Therefore, it again confirms how important a dictionary is. Also, 88 (21.9%) of those uncovered terms are with prefix/suffix. For example, "造影术" (radiography) is an OOV word with suffix "术" (technique), while the word "造影" (radiograph) is contained in the dictionary. However, it is wrongly split into "造影" and "术", since the longest word in the dictionary is preferred. This problem will be our future work.

6 Related work

The word-based generative model (Gao et al., 2003; Zhang et al., 2003) is a classical approach for CWS. However, this approach needs an additional module to recognize OOV words. Therefore, the character-based discriminative model (Xue, 2003; Low et al., 2005; Zhang et al., 2006; Jiang et al., 2008; Zhao et al., 2010) has become the main stream due to its capability in handling OOV words.

However, the character-based discriminative model cannot give satisfactory performance for IV words. Wang et al. (2010) thus proposed a generative model to fix this problem. Afterwards, they

further proposed an integrated model to integrate generative and discriminative approaches, as these two approaches complement each other.

On the other hand, dictionary information has been utilized in the discriminative approach in the previous works of (Low et al., 2005; Zhao et al., 2010). However, they focus on improving the in-domain word segmentation accuracy, while we investigate how the domain invariant feature (based on dictionary information) helps for cross-domain tasks. Besides, the effect of varying OOV words coverage rates is studied in this paper for the first time.

In addition to dictionary feature, Zhao and Kit (2007; 2008), Sun and Xu (2011) too, also adopted the accessor variety feature to gain better generalization ability. Since this feature can be extracted from unlabelled corpora, it is suitable to be adopted for domain adaptation. Again, all their works focus on in-domain performance. Other works that focus on in-domain performance also include (Zhang and Clark, 2007), (Fu et al., 2008), (Jiang et al., 2008), (Lin, 2009), (Xiong et al., 2009), and (Zhang and Clark, 2011).

Last, (Ben-David et al., 2007) pointed out that a good feature representation for domain adaptation should minimize the difference between its distributions in source and target domains. The proposed abstract feature is also inspired by their conclusion.

Our approach differs from those previous works in several ways. First, we do not simply add the dictionary matching information as an additional feature under the Maximum Entropy framework. In contrast, we derive a new generative model with dictionary information starting from the problem formulation, and solve the problem in a principled way. Second, the robustness of the proposed model for varying dictionary coverage rate is first studied and checked in this paper. As explained in Section 4.1.2, this issue is important for selecting a model for real applications.

7 Conclusion

Current character-based approaches are not robust for cross domain Chinese word segmentation, because those surface features adopted in the model frequently possess different tag distributions for the same character in various domains. This paper thus proposes a new abstract aggregate candidate-feature, which indicates if the assigned tag follows the corresponding position-tag of the longest dictionary matching word. With this novel domain invariant feature, we then derive an enhanced generative model for cross-domain CWS to solve the problem in a principled way. Experiments show that the proposed approach is robust for various OOV coverage rates and outperforms the best system in three out of five corpora.

The proposed model is further integrated with an enhanced discriminative approach because they complement each other. With the help of a publically accessible external dictionary, experiments on the SIGHAN-2005 and CIPS-SIGHAN-2010 show that our integrated approach outperforms all the systems in open test and achieves the best F-score in each corpus across five different specified domains.

Acknowledgments

The research work has been funded by the Natural Science Foundation of China under Grant No. 6097 5053 and the Hi-Tech Research and Development Program (“863” Program) of China under Grant No. 2012AA011102 and 2011AA01A207.

References

Ben-David, S., Blitzer, J., Crammer, K. and Pereira, F. (2007). Analysis of representations for domain adaptation. Advances in neural information processing systems, volume 19, pages 137.

Chen, S. and Goodman, J. (1996). An empirical study of smoothing techniques for language modeling. In Proceedings of the 34th annual meeting on Association for Computational Linguistics, pages 310-318.

Emerson, T. (2005). The second international Chinese word segmentation bakeoff. In Proceedings of the fourth SIGHAN workshop, pages 123-133.

Fu, G. Kit, C. and Webster, J. (2008). Chinese word segmentation as morpheme-based lexical chunking. Information Sciences, volume 178, pages 2282-2296.

Gao, J., Li, M. and Huang, C. (2003). Improved Source-Channel Models for Chinese Word Segmentation. In Proceedings of the 41th Annual Meeting of Association of Computational Linguistics (ACL), pages 272-279.

Gao, Q. and Vogel, S. (2010). A Multi-layer Chinese Word Segmentation System Optimized for Out-of-domain Tasks. In Proceedings of CIPS-SIGHAN Joint Conference on Chinese Language Processing (CLP2010), pages 210-215.

Huang, D., Tong, D. and Luo, Y. (2010). HMM Revises Low Marginal Probability by CRF for Chinese Word Segmentation. In Proceedings of CIPS-SIGHAN Joint Conference on Chinese Language Processing (CLP2010), pages 216-220.

Jiang, H. and Dong, Z. (2010). An Double Hidden HMM and an CRF for Segmentation Tasks with Pinyin's Finals. In Proceedings of CIPS-SIGHAN Joint Conference on Chinese Language Processing (CLP2010), pages 277-281.

Jiang, W., Huang, L., Liu, Q. and Lu, Y. (2008). A Cascaded Linear Model for Joint Chinese Word Segmentation and Part-of-Speech Tagging. In Proceedings of ACL, pages 897-904, Columbus, Ohio, USA.

Koehn, P. (2004). Statistical significance tests for machine translation evaluation. In Proceedings of EMNLP, pages 388-395, Barcelona, Spain.

Lin, D. (2009). Combining Language Modeling and Discriminative Classification for Word Segmentation. In Proceedings of CICLing 2009, pages 170-182.

Low, J., Ng, H. and Guo W. (2005). A Maximum Entropy Approach to Chinese Word Segmentation. In Proceedings of the Fourth SIGHAN Workshop on Chinese Language Processing, pages. 161-164, Jeju Island, Korea.

Ng, H. and Low, J. (2004). Chinese part-of-speech tagging: one-at-a-time or all-at-once? word-based or character-based. In Proceedings of EMNLP, pages 277-284, Barcelona, Spain.

Stolcke, A. (2002). SRILM-an extensible language modeling toolkit. In Proceedings of the International Conference on Spoken Language Processing, pages 311-318.

Sun, W. (2010). Word-based and character-based word segmentation models: Comparison and combination. In Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010), pages 1211-1219, Beijing, China.

Sun, W. and Xu, J. (2011). Enhancing Chinese Word Segmentation Using Unlabeled Data. In Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing, pages 970-979, Edinburgh, Scotland, UK.

Sun, X., Wang, H. and Li, W. (2012). Fast Online Training with Frequency-Adaptive Learning Rates for Chinese Word Segmentation and New Word Detection. In Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics, pages 253-262, Jeju Island, Korea.

Wang, K., Zong, C. and Su, K. (2009). Which is More Suitable for Chinese Word Segmentation, the Generative Model or the Discriminative One? In Proceedings of the 23rd Pacific Asia Conference on Language, Information and Computation (PACLIC23), pages 827-834, Hong Kong, China.

Wang, K., Zong, C. and Su, K. (2010). A Character-Based Joint Model for Chinese Word Segmentation. In Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010), pages 1173-1181, Beijing, China.

Wang, K., Zong, C. and Su, K. (2012). Integrating Generative and Discriminative Character-Based Models for Chinese Word Segmentation. ACM Transactions on Asian Language Information Processing, Vol.11, No.2, June 2012, pages 7:1-7:41

Xiong, Y., Zhu, J., Huang, H. and Xu, H. (2009). Minimum tag error for discriminative training of conditional random fields. Information Sciences, volume 179, pages 169-179.

Xue, N. (2003). Chinese Word Segmentation as Character Tagging. Computational Linguistics and Chinese Language Processing, 8 (1). pages 29-48.

Zhang, H., Yu, H., Xiong, D. and Liu, Q. (2003). HHMM-based Chinese lexical analyzer ICTCLAS. In Proceedings of the Second SIGHAN Workshop on Chinese Language Processing, pages 184-187.

Zhang, R., Kikui, G. and Sumita, E. (2006). Subword-based Tagging for Confidence-dependent Chinese Word Segmentation. In Proceedings of the COLING/ACL, pages 961-968, Sydney, Australia.

Zhang, Y. and Clark, S. (2007). Chinese Segmentation with a Word-Based Perceptron Algorithm. In Proceedings of ACL, pages 840-847, Prague, Czech Republic.

Zhang, Y. and Clark, S. (2011). Syntactic processing using the generalized perceptron and beam search. Computational Linguistics, volume 37, pages 105-151.

Zhang, Y., Vogel, S. and Waibel, A. (2004). Interpreting BLEU/NIST scores: How much improvement do we need to have a better system. In Proceedings of the Fourth International Conference on Language Resource and Evaluation (LREC), pages 2051–2054.

Zhao, H. and Kit, C. (2007). Incorporating Global Information into Supervised Learning for Chinese Word Segmentation. In Proceedings of the 10th Conference of the Pacific Association for Computational Linguistics, pages 66-74.

Zhao, H. and Kit, C. (2008). Unsupervised Segmentation Helps Supervised Learning of Character Tagging for Word Segmentation and Named Entity Recognition. In Sixth SIGHAN Workshop on Chinese Language Processing.

Zhao, H. and Liu, Q. (2010). The CIPS-SIGHAN CLP 2010 Chinese Word Segmentation Bakeoff. In Proceedings of CIPS-SIGHAN Joint Conference on Chinese Language Processing (CLP2010), pages 199-209, Beijing, China.

Zhao, H., Huang, C., Li, M. and Lu, B. (2010). A Unified Character-Based Tagging Framework for Chinese Word Segmentation. ACM Transactions on Asian Language Information Processing (TALIP), 9 (2). pages 1-32.

Code-switch Language Model with Inversion Constraints for Mixed Language Speech Recognition

Ying Li, Pascale Fung
Human Language Technology Center
Department of Electronic & Computer Engineering, HKUST
eewing@ust.hk, pascale@ee.ust.hk

ABSTRACT
We propose a first ever code-switch language model for mixed language speech recognition that incorporates syntactic constraints by a code-switch boundary prediction model, a code-switch translation model, and a reconstruction model. A WFST-based decoder then recognizes speech by combining an acoustic model, a pronunciation model and the code-switch language model in an integrated approach. Our proposed approach avoids making early decisions on code-switch boundaries and is therefore more robust than previous approaches. Our proposed system using the code-switch language model outperforms a baseline of interpolated language models by a statistically significant 0.91% on a mixed language lecture speech corpus, and 1.25% on a mixed language lunch conversation corpus. Our method also outperforms a language model that permits code-switch at all word boundaries by a statistically significant 1.35% on the lecture speech corpus and 1.69% on the lunch conversation corpus.

KEYWORDS: Code-switch, mixed language, language modeling.

Proceedings of COLING 2012: Technical Papers, pages 1671–1680,
COLING 2012, Mumbai, December 2012.

1 Introduction

Multilingual people often code-switch (CS) — mixing two languages in the same sentence (intra-sentential code-switch) or between sentences (inter-sentential code-switch). For inter-sentential switches, many researchers use two language models to decode separate sentences (Fugen et al., 2003). However, a sentence that contains two languages poses a more formidable challenge to speech recognition systems, as the same sentence would contain words or phrases belonging to two or more grammatical systems or subsystems (Gumperz, 1982). It is challenging to predict where in the sentence the speaker switches to another language and back, if at all. Some researchers (Lee et al., 2009) use the transcription of the code-switch speech to train a domain-dependent language model. This approach is limited by the small amount of code-switch data available for training.

Code-switch should be distinguished from loanword, which is a word borrowed from one language and incorporated into another language to become part of the lexicon. Code-switch speech is where the speaker actually tries to speak another language in its own grammar. In code-switch, the matrix language is the 'principal' language, where the 'embedded' language is the second language(C. and C., 1993; Coulmas, 1998).

There are two main approaches to recognizing code-switch mixed language speech. One is to detect the boundaries at which the speaker code-switches, then identify the language in the speech segments between the boundaries, and decode the speech segments using the acoustic and language models in the corresponding language(Chan et al., 2005; Shia et al., 2004; Lyu and Lyu, 2008; Vu et al., 2012). For text only code-switch language, Solorio and Liu (2008) used Naive Bayes and Value Feature Interval to classify the hypothesis code-switch points by F-measure and the naturalness rating.

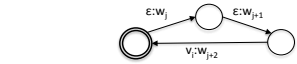
However, this approach requires multiple passes of boundary detection, language identification and speech recognition. The boundaries and language identity of each speech segment are irreversibly determined by the previews pass. Moreover, the speech segment of the embedded language tends to be very short. This poses challenges to the state-of-the-art language identification approaches.

A more holistic way to decode code-switch speech is by using a set of universal acoustic models for both matrix and embedded languages and a language model that permits code-switch (while predicting with probability where CS might occur) and which does not require an early decision of the code-switch points.

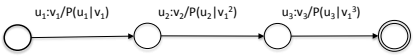
There are many methods proposed to build universal acoustic models for both the matrix and embedded languages that range from mapping the pronunciation dictionary to phonetic set combination and acoustic model merging (Imseng et al., 2011; Li et al., 2011; Bhuvanagiri and Kopparapu, 2010; Zhang et al., 2008). In this paper, we focus on code-switch language modeling.

A common approach is to build a code-switch language model from the language models of the matrix and embedded languages, trained separately from monolingual texts and combined together with linear interpolation (Bhuvanagiri and Kopparapu, 2010; Li et al., 2011; Imseng et al., 2011). This approach does not assume any syntactic constraint.

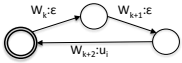
One can also use hand written grammar to constrain the code-switch point, such as in Zhang et al., 2008 or a bilingual dictionary to map the statistical n-grams in one language to the other, such as in Cao et al., 2010. Yeh et al., 2010 used a class-based n-gram language model based on



A portion of the code-switch boundary prediction model



A portion of the code-switch translation model



A portion of the code-switch reconstruction model

Figure 1: Weighted finite state transducers of the code-switch language model

perplexity and part-of-speech tag features. Tsai et al., 2010 proposed to use part-of-speech tags and to use Hakka-Chinese bilingual word mapping in the language model. The code-switch points are deterministic.

Linguists who study mixed language speech have discovered a typical constraint when speakers switch from one language to another in a sentence(Woolford, 1983; Mahootian, 1993; MacSwan, 1999). They found that code-switch can only occur in positions where "*the order of any two sentence elements, one before and one after the switch, is not excluded in either language*"(Poplack and Sankoff, 1980). This constraint, known as the “equivalence constraint" in linguistics, corresponds to an inversion constraint in statistical machine translation.

In this paper, we propose for the first time to incorporate this syntactic inversion constraint to a statistical code-switch language model. Our CS language model is composed of a CS boundary prediction model, a CS translation model, and a reconstruction model. The prediction model learns from word aligned parallel sentences to give the permissible CS points. The translation model is obtained by logit regression and incorporates syntactic inversion constraints. A *maximum a posterior* framework employs weighted finite state transducers in the process of final decoding, integrating a bilingual acoustic model, a code-switch language model, and a monolingual language model in the matrix language.

The structure of the paper is as follows: Section 2 presents the code-switch language model. The framework of decoding via the weighted finite state transducers is described in Section 3. Section 4 describes the results of the experiments. Section 5 draws the conclusion.

2 Code-switch language modeling

Given a speech utterance, S, with N_S frames, the automatic speech translation system converts S into a word sequence, W_1^M, where M is the total number of words. There are four components in the system, the language model, $P(W_1^M)$; the acoustic models, $P(S_1^{N_S}|W_1^M)$; the pronunciation dictionary; and the decoder.

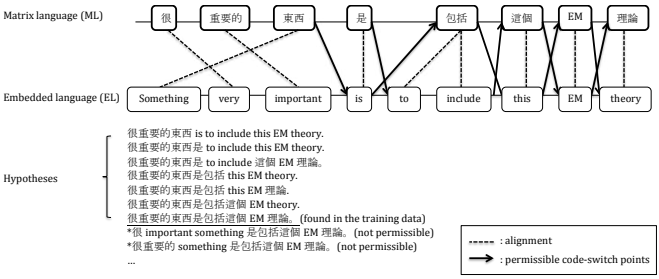


Figure 2: An example of permissible code-switch points

It is difficult to obtain mixed language text data which can be used for training the language model. Instead of directly calculating the probabilities, we use a monolingual language model in the matrix languagetogether with the CS language model.

$$P(W_1^M) = \sum_{w_1^m} P(w_1^m)P(W_1^M|w_1^m) \cong \max_{w_1^m} P(w_1^m)P(W_1^M|w_1^m) \quad (1)$$

where W_1^M is in mixed language, and w_1^m is in the matrix language. The code-switch language model can be modeled as

$$P(W_1^M|w_1^m) \cong \max_{v_1^n, u_1^n, W_1^M} \{P(v_1^n, n|w_1^m) \cdot P(u_1^n|v_1^n, w_1^m) \cdot P(W_1^M|u_1^n, v_1^n, w_1^m)\} \quad (2)$$

where $P(v_1^n, n|w_1^m)$ is the code-switch boundary prediction model, $P(u_1^n|v_1^n, w_1^m)$ is the code-switch translation model, and $P(W_1^M|u_1^n, v_1^n, w_1^m)$ is the reconstruction model. We assume that w_1^m is a word sequence in the matrix language; the words are segmented into phrases, v_1^n; and u_1^n is a phrase sequence in mixed language.

2.1 Code-switch boundary prediction model

According to the equivalence constraint suggested by linguists(Poplack and Sankoff, 1980), the code cannot occur at the points where the order of the words are inverted between the matrix language and the embedded language. An example of a Mandarin-English mixed language sentence is shown in Figure 2. Code-switch is not allowed between the first three words with syntactic inversion.

Word-aligned parallel sentences in the matrix and the embedded languages are used to constrain at which point code-switch is allowed. We propose to generate bilingual training data as follows:

1) Translate words of the mixed language sentences in the embedded language into the matrix language using a statistical machine translation system;

2) Translate the monolingual sentences from 1) into the embedded language by a statistical machine translation system with inversion transduction grammar constraint (Wu, 1997; Wu and Fung, 2005) to obtain monolingual sentences in the embedded language;

3) Align the pairs of monolingual sentences in the matrix and embedded languages from 1) and 2). In theory, we can generate as many bilingual sentence pairs as possible.

The code-switch boundary prediction model trains the probabilities of a sequence of words segmented into a sequence of phrases from the aligned parallel sentences. A phrase is a word or a concatenation of words in which there exists one or more inversions of an aligned parallel sentence pair in the matrix language and the embedded language.

$$P(v_1^n, n|w_1^m) = \frac{1}{Z_n} \prod_{i=1}^{n} P(v_i) \tag{3}$$

where $P(v_i)$ can be approximated by the relative frequency of the i-th phrase.
$Z_n = \sum_{v_1^n} \prod_{k=1}^{m} P(v_i)$ such that $\sum_{v_1^n} P(v_1^n, n|w_1^m) = 1$.

2.2 Code-switch translation model

Given the permissible code-switch points by the above model, the code-switch translation model is the actual probability of code-switch at these points. We assume that the code-switch translation probability, $P(u_1^i|v_1^i)$, depends on the previous phrase, v_{i-1}.

The code-switch translation probability distribution is specified by probabilities $\pi(\mathbf{x})$ of code-switch and $(1-\pi(\mathbf{x}))$ of not code-switch. $\mathbf{x}$ is an n-tuple containing the conditional probability $P(e|w)$ of code-switch from a word, w, in the matrix language to a word, e, in the embedded language, and reordering probability $\prod_{j=1}^{k} P(r_j|j,k,l)$ of a phrase in the matrix language of length k and a phrase in the embedded language of length m, where r_j denotes that the j-th ML word is aligned to the r_j-th EL word, phrase translation probability $Pr(u|v)$ from a phrase, v, in the matrix language to a phrase, u, in the embedded language and phrase penalty $Pen(v)$. These probabilities are trained from word-aligned bilingual sentences.

The code-switch translation probability is neither linear nor exponential; it changes dramatically near the CS threshold. Thus we propose to use a logit regression model to describe the code-switch translation probability

$$\text{logit}[\pi(\mathbf{x})] = \log(\frac{\pi(\mathbf{x})}{1-\pi(\mathbf{x})}) = \alpha + \sum \beta_j x_j \tag{4}$$

where β_j refers to the effect of the j-th item in the n-tuple $\mathbf{x}$ on the logit of the code-switch translation probabilities, controlling the other items of the n-tuple $\mathbf{x}$. The code-switch translation probability

$$\pi(\mathbf{x}) = \frac{\exp(\alpha + \sum \beta_j x_j)}{1 + \exp(\alpha + \sum \beta_j x_j)} \tag{5}$$

$$P(u_i|v_1^i) = \begin{cases} 1 - \pi(\mathbf{x_{i-1}^i}) & u_i = v_i \\ \pi(\mathbf{x_{i-1}^i}) & otherwise \end{cases} \tag{6}$$

where $\mathbf{x_{i-1}^i}$is the n-tuple of the word alignment probabilities, reordering probability and the phrase penalty.

2.3 Code-switch reconstruction model

The code-switch reconstruction model assigns probabilities to a sequence of mixed language words, W_1^M, given the constraint that the words agree with u_1^n, v_1^n, n, w_1^m

$$P(W_1^M|u_1^n, v_1^n, n, w_1^m) = \prod_{i=1}^{n} P(W_{S_i}^{E_i}|u_i) \tag{7}$$

$$P(W_{S_i}^{E_i}|u_i) = \begin{cases} \frac{1}{Z_i} \prod_{j=S_i}^{E_i} q(W_j) & W_{S_i}^{E_i} = u_i \\ 0 & otherwise \end{cases} \tag{8}$$

$p(W_j)$ is the frequency of occurrences of word W_j obtained from the bilingual sentences. $W_{S_i}^{E_i} = u_i$ indicates that the word sequence $W_{S_i}^{E_i}$ is exactly the same as the phrase u_i, S_i is the start of phrase u_i, and E_i is the end of phrase u_i. $Z_i = \sum_{u_1^n} \prod_{j=S_i}^{E_i} p(W_j)$ is set so that the probabilities sum to unity over possible values of u_i.

3 Decoding via weighted finite state transducers

The decoding of mixed language speech can be considered as searching the weighted finite state network. Suppose S denotes a speech utterance with N_S feature vectors, the recognition result $\hat{W}_1^M$ can be found as

$$\hat{W}_1^M = \arg\max_{W_1^M} P(W_1^M|S_1^{N_S}) \tag{9}$$

By Bayesian rule, the $\hat{W}_1^M$ is in

$$\{\hat{v}_1^n, \hat{u}_1^n, \hat{W}_1^M\} = \arg\max_{v_1^n, u_1^n, W_1^M} P(S_1^{N_S}|W_1^M) P(w_1^m) P(v_1^n, n|w_1^m) P(u_1^n|v_1^n, w_1^m) P(W_1^M|u_1^n, v_1^n, w_1^m) \tag{10}$$

while the possible code-switches are specified by $\hat{v}_1^n$ and $\hat{u}_1^n$.

The acoustic models $P(S_1^{N_S}|W_1^M)$, the pronunciation dictionary, the matrix language model $P(w_1^m)$, the code-switch boundary prediction model $P(v_1^n, n|w_1^m)$, the code-switch translation model $P(u_1^n|v_1^n, w_1^m)$ and the reconstruction model $P(W_1^M|u_1^n, v_1^n, w_1^m)$ are implemented as weighted finite state transducers and composed into a search network.

A lattice of the hypothesis speech recognition result is generated. When each token in the lattice transits from the end of one word to the start of the next word, a word insertion penalty is added. The insertion penalties of Chinese words and English words are separately trained from the development data. Lattice rescoring is proposed to adjust the word penalty of recognition results.

4 Experimental setup

This section briefly describes the data resources and feature analysis that were used for all the experiments used to evaluate the presented approach in the paper.

The acoustic features used in our experiments consist of 39 components (13MFCC,13ΔMFCC,13$\Delta\Delta$MFCC using subtraction of the cepstral mean), which are

analyzed at a 10msec frame rate with a 25msec window size. The acoustic models used throughout our paper are state-clustered crossword tri-phone HMMs with 16 Gaussian mixture output densities per state. The phone set consists of 21 standard Mandarin initials, 37 toneless Mandarin finals, 6 zero initials and 6 English extended phones. The pronunciation dictionary is obtained by modified dictionaries in the matrix and embedded languages using the phone set. The acoustic models are adapted to the speakers using maximum likelihood linear regression. The WFST decoder is used for decoding.

4.1 Corpora

We evaluate our proposed method on two mixed language speech corpora of different speaking styles, namely a lecture speech corpus and a lunch conversation corpus. The audio data is sampled at 16kHz.

About 20 hours of lecture speech of a digital speech processing course recorded at National Taiwan University are separated into three sets. Eighteen hours of the speech is used for adaptation of the acoustic models, and 0.9 hours of the speech is used as a development set. The testing set contains one hour of 1037 utterances. The lecture is given in Mandarin by a single speaker with 16% embedded English words.

The lunch conversation speech was recorded at the Hong Kong University of Science and Technology from a single speaker. The speech is highly spontaneous and the topics are wide ranging. The total length of the conversations is 163 minutes and 127 minutes are used to adapt acoustic models. The development set contains 26 minutes of the speech. Ten minutes of the speech is used for testing. There are 14762 Chinese words and 4280 English words. The percentage of the embedded English words is 22%.

250,000 sentences from digital speech processing conference papers, power point slides and web data are used for language model training and parallel sentence generation for the lecture speech recognition task(LM data 1). 250,000 sentences of the Gale conversational speech transcription are used for language model training and parallel sentence generation for the lunch conversion speech recognition(LM data 2).

4.2 Langauge models

The baseline language model (Model_IP) for the lecture speech recognition is an interpolation of the language model trained from LM data 1 and the language model trained on the transcriptions of the mixed language lecture speech. Another baseline model (Model_IP) of the lunch conversations is trained from LM data 2 and interpolated with the language model trained from the transcriptions of the mixed language conversations.

Our proposed language model (Model_CS) is constructed from combining the monolingual language model, the code-switch boundary prediction model, the code-switch translation model and the reconstruction model.

5 Experimental Results

Table 1 shows the word or character error rates of experiments on the mixed language lecture speech. The baseline interpolated language model reduces the overall word error rate by 0.49%. However, it degrades the recognition results of English phrases. Our proposed method outperforms the interpolated language model by 0.45% (Mandarin CER), 1.86% (English WER)

and 0.89% (overall).

Table 1: *Word/character error rate (%) of the lecture speech*

	Mandarin (ML)	English (EL)	Overall
Allow code-switch any where	35.16	43.54	36.55
Interpolated LM	34.35	44.36	36.09
Proposed method	**33.90**	**41.68**	**35.2**

The results of experiments on the mixed language lunch conversation speech are shown in Table 2. The baseline interpolated language model gives a 0.8% character error rate reduction on the Mandarin phrases and a 0.44% overall word error rate reduction, but degrades the performance on the English phrases. On the other hand, our proposed CS language model reduces the character error rate of Mandarin phrases by 0.41%, the word error rate of English phrases by 1.98% and overall word error rate by 1.25%. All the character error rate and word error rate reductions are statistically significant at 99%.

Table 2: *Word/character error rate (%) of the lunch conversations*

	Mandarin (ML)	English (EL)	Overall
Allow code-switch any where	47.20	49.14	47.63
Interpolated LM	46.40	49.98	47.19
Proposed method	**45.99**	**48.01**	**45.94**

Conclusions

In this paper, we propose a first ever statistical language model of code-switch speech that incorporates syntactic inversion constraints that have been found in that kind of speech. Our language model is composed of a code-switch prediction model, a translation model and a reconstruction model. A WFST-based decoder integrates this code-switch language model with an acoustic model and a monolingual language model in the matrix language for the final decoding. We tested our system on two tasks in mixed language lecture speech recognition, with 16% English words in Chinese sentences; and in mixed language lunch conversation, with 22% English words in Chinese sentences. Our system reduces word error rate in a baseline of the interpolated language model by 0.91% in the first task, and by 1.25% in the second task. Our model also outperforms another baseline, that of allowing code-switch at all points by 1.35% in the first task, and by 1.69% in the second task. All results are statistically significant. In addition, our method reduces error rates for both the matrix language and the embedded language.

Acknowledgments

This work was partially supported by grant number RGF 612211 of the Hong Kong Research Grant Council.

References

Bhuvanagiri, K. and Kopparapu, S. (2010). An approach to mixed language automatic speech recognition. In *Oriental COCOSDA, Kathmandu, Nepal*.

C., M.-S. and C., M. (1993). *Duelling languages: Grammatical structure in codeswitching*. Clarendon Press Oxford.

Cao, H., Ching, P., Lee, T., and Yeung, Y. (2010). Semantics-based language modeling for cantonese-english code-mixing speech recognition. In *Chinese Spoken Language Processing (ISCSLP), 2010 7th International Symposium on*, pages 246–250. IEEE.

Chan, J., Ching, P., Lee, T., and Meng, H. (2005). Detection of Language Boundary in Code-switching utterances by Bi-phone Probabilities. In *Chinese Spoken Language Processing, 2004 International Symposium on*, pages 293–296. IEEE.

Coulmas, F. (1998). *The handbook of sociolinguistics*, volume 4. Wiley-Blackwell.

Fugen, C., Stuker, S., Soltau, H., Metze, F., and Schultz, T. (2003). Efficient handling of multilingual language models. In *Automatic Speech Recognition and Understanding, 2003. ASRU'03. 2003 IEEE Workshop on*, pages 441–446. IEEE.

Gumperz, J. (1982). *Discourse strategies*, volume 1. Cambridge Univ Pr.

Imseng, D., Bourlard, H., Magimai-Doss, M., and Dines, J. (2011). Language dependent universal phoneme posterior estimation for mixed language speech recognition. In *Acoustics, Speech and Signal Processing (ICASSP), 2011 IEEE International Conference on*, pages 5012–5015. IEEE.

Lee, H., Tang, Y., Tang, H., and Lee, L. (2009). Spoken term detection from bilingual spontaneous speech using code-switched lattice-based structures for words and subword units. In *Automatic Speech Recognition & Understanding, 2009. ASRU 2009. IEEE Workshop on*, pages 410–415. IEEE.

Li, Y., Fung, P., Xu, P., and Liu, Y. (2011). Asymmetric acoustic modeling of mixed language speech. In *Acoustics, Speech and Signal Processing (ICASSP), 2011 IEEE International Conference on*, pages 5004–5007. IEEE.

Lyu, D. and Lyu, R. (2008). Language identification on code-switching utterances using multiple cues. In *Ninth Annual Conference of the International Speech Communication Association*.

MacSwan, J. (1999). *A minimalist approach to intrasentential code switching*. Routledge.

Mahootian, S. (1993). *A null theory of codeswitching*. PhD thesis, Northwestern University.

Poplack, S. and Sankoff, D. (1980). A formal grammar for code-switching. *Papers in Linguistics: International Journal of Human Communication*, 14:3–45.

Shia, C., Chiu, Y., Hsieh, J., and Wu, C. (2004). Language boundary detection and identification of mixed-language speech based on map estimation. In *Acoustics, Speech, and Signal Processing, 2004. Proceedings.(ICASSP'04). IEEE International Conference on*, volume 1, pages I–381. IEEE.

Solorio, T. and Liu, Y. (2008). Learning to predict code-switching points. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 973–981. Association for Computational Linguistics.

Tsai, T., Chiang, C., Yu, H., Lo, L., Wang, Y., and Chen, S. (2010). A study on hakka and mixed hakka-mandarin speech recognition. In *Chinese Spoken Language Processing (ISCSLP), 2010 7th International Symposium on*, pages 199–204. IEEE.

Vu, N., Lyu, D., Weiner, J., Telaar, D., Schlippe, T., Blaicher, F., Chng, E., Schultz, T., and Li, H. (2012). A first speech recognition system for mandarin-english code-switch conversational speech.

Woolford, E. (1983). Bilingual code-switching and syntactic theory. *Linguistic Inquiry*, 14(3):520–536.

Wu, D. (1997). Stochastic inversion transduction grammars and bilingual parsing of parallel corpora. *Computational Linguistics*, 23(3):377–403.

Wu, D. and Fung, P. (2005). Inversion transduction grammar constraints for mining parallel sentences from quasi-comparable corpora. *Natural Language Processing–IJCNLP 2005*, pages 257–268.

Yeh, C., Huang, C., Sun, L., and Lee, L. (2010). An integrated framework for transcribing mandarin-english code-mixed lectures with improved acoustic and language modeling. In *Chinese Spoken Language Processing (ISCSLP), 2010 7th International Symposium on*, pages 214–219. IEEE.

Zhang, Q., Pan, J., and Yan, Y. (2008). Mandarin-English bilingual speech recognition for real world music retrieval. In *Acoustics, Speech and Signal Processing, 2008. ICASSP 2008. IEEE International Conference on*, pages 4253–4256. IEEE.

A Separately Passive-Aggressive Training Algorithm for Joint POS Tagging and Dependency Parsing

Zhenghua Li[1], Min Zhang[2], Wanxiang Che[1], Ting Liu[1]*
(1) Research Center for Social Computing and Information Retrieval,
Harbin Institute of Technology, China
(2) Institute for Infocomm Research, Singapore
{lzh,car,tliu}@ir.hit.edu.cn, mzhang@i2r.a-star.edu.sg

ABSTRACT
Recent study shows that parsing accuracy can be largely improved by the joint optimization of part-of-speech (POS) tagging and dependency parsing. However, the POS tagging task does not benefit much from the joint framework. We argue that the fundamental reason behind is because the POS features are overwhelmed by the syntactic features during the joint optimization, and the joint models only prefer such POS tags that are favourable solely from the parsing viewpoint. To solve this issue, we propose a separately passive-aggressive learning algorithm (SPA), which is designed to separately update the POS features weights and the syntactic feature weights under the joint optimization framework. The proposed SPA is able to take advantage of previous joint optimization strategies to significantly improve the parsing accuracy, but also overcome their shortages to significantly boost the tagging accuracy by effectively solving the syntax-insensitive POS ambiguity issues. Experiments on the Chinese Penn Treebank 5.1 (CTB5) and the English Penn Treebank (PTB) demonstrate the effectiveness of our proposed methodology and empirically verify our observations as discussed above. We achieve the best tagging and parsing accuracies on both datasets, 94.60% in tagging accuracy and 81.67% in parsing accuracy on CTB5, and 97.62% and 93.52% on PTB.

KEYWORDS: Part-of-speech Tagging, Dependency Parsing, Joint Models, Separately Passive-aggressive Algorithm.

* Corresponding author

Proceedings of COLING 2012: Technical Papers, pages 1681–1698,
COLING 2012, Mumbai, December 2012.

1 Introduction

Given an input sentence of n words, denoted by $\mathbf{x} = w_1...w_n$, part-of-speech (POS) tagging aims to find an optimal tag sequence $\mathbf{t} = t_1...t_n$, where $t_i \in \mathcal{T}$ $(1 \leq i \leq n)$ and $\mathcal{T}$ is a predefined tag set. POS tags are designed to represent word classes so that words of the same POS tag play a similar role in syntactic structures. The size of $\mathcal{T}$ is usually much less than the vocabulary size. Typically, POS tagging is treated as a sequence labeling problem, and has been previously addressed by machine learning algorithms, such as maximum-entropy (Ratnaparkhi, 1996), conditional random fields (CRF) (Lafferty et al., 2001) and perceptron (Collins, 2002). Figure 1 gives an example sentence from Penn Chinese Treebank 5.1 (CTB5). The lowest three rows present the n-best POS tags for each word, produced by a state-of-the-art CRF model. Looking at the 1-best POS tags, we can see that the CRF model makes four errors, i.e. de/DEC→DEG, ouwen/NR→NN, xiaoli/VV→NN, and liwupudui/NR→NN. In fact, (DEC,DEG) and (NN,VV) ambiguities, which usually require long-distance syntactic knowledge to resolve, are very difficult for the sequential labeling models.

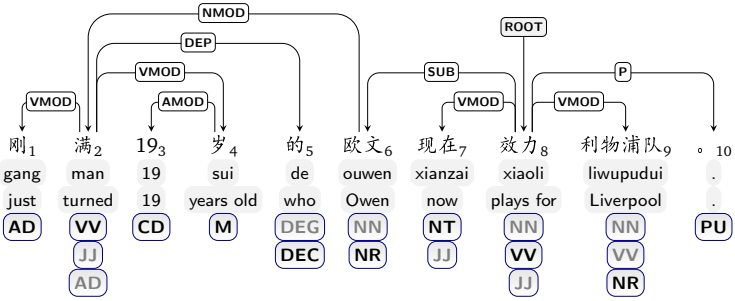


Figure 1: An example from CTB5. The Pinyin transcriptions and English translations are presented in the two rows below the Chinese sentence. We prune out the implausible POS tags according to the marginal probabilities (see Section 4.1) and list the top three candidate POS tags in the lowest three rows (incorrect POS tags in grey color and correct ones in black color).

Dependency Parsing maps a natural language sentence into a structural dependency tree conforming to a predefined dependency grammar, as depicted in Figure 1. A dependency tree is denoted by $\mathbf{d} = \{(h, m, l) : 1 \leq h \leq n, 1 \leq m \leq n, l \in \mathcal{L}\}$, where (h, m, l) means a dependency from the *head word* (also called *father*) w_h to the *modifier* (also called *child* or *dependent*) w_m with a dependency label l, and $\mathcal{L}$ is the label set. Dependency labels are used to indicate the syntactic or semantic relation between the two words. For instance, the dependency (8, 6, SUB) in Figure 1 means ouwen is the *subject* of xiaoli.

Data-driven dependency parsing models make heavy use of POS tags to compose supporting features, since it leads to severe data sparseness problem if only lexical features are used. However, POS tagging errors significantly degrade the parsing accuracy by about 6% (see Table 4 where, for example, due to the error of xiaoli/VV→NN, our pipelined parsing model fails to recognize xiaoli as the predicate of the sentence and returns a fully unreasonable structure).

Recently, there has been increasing interest in joint modeling of Chinese POS tagging and

dependency parsing (Li et al., 2011; Hatori et al., 2011; Bohnet and Nivre, 2012), motivated by the intuition that the two individual tasks should help each other. Their work demonstrates that the joint models can substantially boost the parsing accuracy. In contrast, the tagging subtask does not benefit much from the joint framework. (Li et al., 2011) show that the graph-based joint models lead to large decrease in the tagging accuracy, whereas (Hatori et al., 2011) and (Bohnet and Nivre, 2012) find small gains in the tagging accuracy with transition-based joint models (see Table 4). This is contradictory to the intuition that better syntactic structure should help POS disambiguation. The detailed error analysis in (Li et al., 2011) show that their joint models are helpful in resolving syntax-sensitive POS ambiguities like {VV,NN} and {DEC,DEG}, but become very weak in disambiguating {NN,NR} and {NN,JJ} which usually play similar roles in syntactic trees.

We believe that one possible reason is that the graph-based joint models of (Li et al., 2011) is dominated by the syntactic features. Looking deeper into their joint models, we find that on average, the score corresponding to the POS features only is $1/50$ of the score of the syntactic features in a returned joint result. In other words, the POS features have little impact on determining the best joint result. Therefore, the joint models prefer such POS tags that are more helpful and discriminative solely from the parsing viewpoint.

To address this issue, this paper proposes a variant of the passive-aggressive (PA) online training algorithm (Crammer et al., 2003), which we name as *separately passive-aggressive algorithm (SPA)*. SPA separately updates the POS feature weights and the syntactic feature weights and naturally raises the weights of the POS features under the joint optimization framework. As a result, SPA can make better use of the discriminative power of the POS features in resolving the syntax-insensitive POS ambiguities, leading to a large tagging accuracy improvement. On the other hand, the improved tagging accuracy can further help parsing. Specifically, we make the following contributions.

- We propose a separately passive-aggressive training algorithm for joint POS tagging and dependency parsing. Empirically, we compare SPA with averaged perceptron (AP) and PA from the perspective of the model score, showing that SPA is more suitable for the joint models. Experimental results demonstrate that SPA outperforms AP and PA in both the tagging and parsing accuracies. More importantly, SPA significantly improve both the tagging and parsing accuracies over the pipelined baselines.
- We present the first feature-rich graph-based joint model for POS tagging and labeled dependency parsing. We conduct experiments on two versions of CTB5 and achieve the best tagging and parsing accuracies on both datasets. Especially, our joint model trained with SPA achieves a tagging accuracy of 94.7%, largely outperforming the 93.9% of the baseline CRF model.
- We also conduct experiments on PTB to find out the effect of joint modeling on English. Thanks to relatively richer morphologies, English POS tagging achieves much higher accuracy than Chinese (97% vs. 94%). Therefore, English dependency parsing suffers less error propagation problem than Chinese. Nevertheless, we still find significant gains from the joint model. The tagging accuracy and the parsing accuracy increase by about 0.5% and 0.4% respectively. Our joint model achieves the best tagging and parsing accuracies on this dataset as well.
- We present the first work that conducts extensive studies on the effect of labeled dependency parsing. We find that dependency labels consistently improve the unlabeled parsing accuracy. Especially, the unlabeled attachment score (UAS) can be boosted by

0.6-0.7% on the two Chinese datasets.

2 Related work

This work is most closely related to (Li et al., 2011) who present the first work on joint models for Chinese POS tagging and unlabeled dependency parsing. Similar to us, their joint models are based on graph-based dependency parsing. They find that the joint models largely outperform the pipeline models in the parsing accuracy but lead to substantial tagging accuracy drop. Compared with their work, we propose a better training algorithm for the joint models that can improve both tagging and parsing accuracies. In addition, our joint model adopts richer features and handles labeled dependency parsing.

(Hatori et al., 2011) propose the first transition-based joint model for Chinese POS tagging and unlabeled dependency parsing and gain large improvement in the parsing accuracy. However, their joint models only slightly improve the tagging accuracy over a sequential tagging model. (Bohnet and Nivre, 2012) propose a transition-based joint model which can handle labeled non-projective dependency parsing. They conduct experiments on a variety of languages including Chinese, English, Czech, and German. Similarly, their joint model largely improves the parsing accuracy but only slightly increases the tagging accuracy. Differently, we are the first work on joint POS tagging and dependency parsing that achieves large improvement in the tagging accuracy.

(Smith and Eisner, 2008) apply loopy belief propagation (LBP) to dependency parsing and points out that LBP can naturally represent POS tags as latent variables so that the POS tags can be inferred jointly with the parse. (Lee et al., 2011) extend the LBP based approach of (Smith and Eisner, 2008) and study joint morphological disambiguation and dependency parsing for morphologically-rich languages including Latin, Czech, Ancient Greek, and Hungarian. For these languages, morphological analysis requires the disambiguation of POS tags, gender, case, etc. They show that the joint model can well capture the interaction between morphology and syntax and achieve gains on both subtasks. (Rush et al., 2010) propose dual decomposition (DD) for integrating different NLP subtasks at the test phase. They experiment with two cases, one integrating a phrase-structure parser and a dependency parser, and the other integrating a phrase-structure parser and a POS tagger. Both cases show that DD can help the individual subtasks. (Auli and Lopez, 2011) conduct an extensive comparison of LBP and DD for joint CCG supertagging and parsing. They show that LBP and DD achieves similar parsing accuracy improvement but has largely different convergence characteristics. Moreover, their work focuses on integrating two separately-trained sub-models, and they find that training the integrated model on LBP leads to large improvement drops compared with separately-trained models.

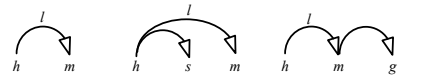
3 Pipeline POS tagging and dependency parsing

The pipeline method treats POS tagging and dependency parsing as two cascaded problems. First, an optimal POS tag sequence $\hat{\mathbf{t}}$ is determined.

$$\hat{\mathbf{t}} = \arg\max_{\mathbf{t}} \mathrm{Score}_{\mathrm{pos}}(\mathbf{x}, \mathbf{t}) \tag{1}$$

Then, an optimal dependency tree $\hat{\mathbf{d}}$ is determined based on $\mathbf{x}$ and $\hat{\mathbf{t}}$.

$$\hat{\mathbf{d}} = \arg\max_{\mathbf{d}} \mathrm{Score}_{\mathrm{syn}}(\mathbf{x}, \hat{\mathbf{t}}, \mathbf{d}) \tag{2}$$



(a) *single dependency* (b) *adjacent sibling* (c) *outermost grandchild*

Figure 2: Three types of scoring subtrees used in our parsing and joint models.

Feature category	Atomic features incorporated
Dependency features $\mathbf{f}_{\text{dep}}(\mathbf{x},\mathbf{t},h,m,l)$	$l, w_h, w_m, t_h, t_m, t_{h\pm1}, t_{m\pm1}, t_b, dir(h,m), dist(h,m)$
Sibling features $\mathbf{f}_{\text{sib}}(\mathbf{x},\mathbf{t},h,m,l,s)$	$l, w_h, w_s, w_m, t_h, t_m, t_s, t_{h\pm1}, t_{m\pm1}, t_{s\pm1}, dir(h,m), dist(h,m)$
Grandchild features $\mathbf{f}_{\text{grd}}(\mathbf{x},\mathbf{t},h,m,l,g)$	$l, w_h, w_m, w_g, t_h, t_m, t_g, t_{h\pm1}, t_{m\pm1}, t_{g\pm1}, dir(h,m), dir(m,g)$

Table 1: Brief illustration of the syntactic features. b is an index between h and m. $dir(i,j)$ and $dist(i,j)$ denote the direction and distance of the dependency (i,j). Please refer to Table 4 of (Bohnet, 2010) for the complete feature list.

CRF-based POS tagging. We adopt the first-order CRF to build our baseline POS tagger. As a conditional log-linear probabilistic model, CRF defines the probability of a tag sequence as

$$\begin{aligned} P(\mathbf{t}|\mathbf{x}) &= \exp(\texttt{Score}_{\text{pos}}(\mathbf{x},\mathbf{t}))/\sum_{\mathbf{t}'}\exp(\texttt{Score}_{\text{pos}}(\mathbf{x},\mathbf{t}')) \\ \texttt{Score}_{\text{pos}}(\mathbf{x},\mathbf{t}) &= \mathbf{w}_{\text{pos}}\cdot\mathbf{f}_{\text{pos}}(\mathbf{x},\mathbf{t}) = \sum_{1\le i\le n}\mathbf{w}_{\text{pu}}\cdot\mathbf{f}_{\text{pu}}(\mathbf{x},t_i) + \mathbf{w}_{\text{pb}}\cdot\mathbf{f}_{\text{pb}}(\mathbf{x},t_{i-1},t_i) \end{aligned} \tag{3}$$

where $\mathbf{f}_{\text{pos/pu/pb}}(.)$ refers to the feature vectors and $\mathbf{w}_{\text{pos/pu/pb}}$ is the corresponding weight vectors. We call $\mathbf{f}_{\text{pu}}(\mathbf{x},t_i)$ the *POS unigram features*, and $\mathbf{f}_{\text{pb}}(\mathbf{x},t_{i-1},t_i)$ the *POS bigram features*. For Chinese, we adopt the features proposed by (Zhang and Clark, 2008a). They use Chinese characters contained in a word to compose rich features, which turns out to be helpful for low-frequency words. For English, we adopt the features of (Ratnaparkhi, 1996) which exploit suffixes and prefixes to improve tagging performance over rare words.

Second-order graph-based dependency parsing. The graph-based approach views dependency parsing as finding a highest scoring tree in a directed graph (McDonald et al., 2005; Carreras, 2007; Koo and Collins, 2010). We adopt the second-order model of (Carreras, 2007) since previous studies show that it leads to best parsing accuracy on a variety of languages (Koo and Collins, 2010; Bohnet, 2010). The score of a dependency tree is factored into scores of the three kinds of subtrees in Figure 2.

$$\begin{aligned} \texttt{Score}_{\text{syn}}(\mathbf{x},\mathbf{t},\mathbf{d}) &= \mathbf{w}_{\text{syn}}\cdot\mathbf{f}_{\text{syn}}(\mathbf{x},\mathbf{t},\mathbf{d}) \\ &= \sum_{\{(h,m,l)\}\subseteq\mathbf{d}}\mathbf{w}_{\text{dep}}\cdot\mathbf{f}_{\text{dep}}(\mathbf{x},\mathbf{t},h,m,l) \\ &+ \sum_{\{(h,m,l),(h,s)\}\subseteq\mathbf{d}}\mathbf{w}_{\text{sib}}\cdot\mathbf{f}_{\text{sib}}(\mathbf{x},\mathbf{t},h,m,l,s) \\ &+ \sum_{\{(h,m,l),(m,g)\}\subseteq\mathbf{d}}\mathbf{w}_{\text{grd}}\cdot\mathbf{f}_{\text{grd}}(\mathbf{x},\mathbf{t},h,m,l,g) \end{aligned} \tag{4}$$

For syntactic features, we adopt those of (Bohnet, 2010) which include three categories corresponding to the three types of scoring subtrees. We summarize the atomic features used in

each feature category in Table 1. For unlabeled parsing and joint models, the label l is omitted. Compared with the syntactic features used in (Li et al., 2011), this feature set explores more context POS tags including $t_{s\pm1}, t_{g\pm1}$. In addition, for Chinese, we use the last character of each word as its lemma, and duplicate each word-related feature by replacing words with lemmas (Che et al., 2012). Experiments on CTB5 show that these lemma-related features can improve our baseline parsing models by 0.3-0.4% in UAS. For English, we use coarse-grained POS tags to duplicate all the feature that depend on POS tags (Koo and Collins, 2010), resulting in a 0.4% UAS gain on PTB.

4 Joint POS tagging and dependency parsing

In the joint framework, we aim to simultaneously solve the two problems.

$$(\hat{\mathbf{t}}, \hat{\mathbf{d}}) = \arg\max_{\mathbf{t},\mathbf{d}} \text{Score}_{\text{joint}}(\mathbf{x}, \mathbf{t}, \mathbf{d}) \tag{5}$$

The score of a tagged dependency tree is the combination of the POS score and the syntactic score that are previously defined in the pipeline models.

$$\begin{aligned} \text{Score}_{\text{joint}}(\mathbf{x}, \mathbf{t}, \mathbf{d}) &= \text{Score}_{\text{pos}}(\mathbf{x}, \mathbf{t}) + \text{Score}_{\text{syn}}(\mathbf{x}, \mathbf{t}, \mathbf{d}) \\ &= \mathbf{w}_{\text{pos}} \cdot \mathbf{f}_{\text{pos}}(\mathbf{x}, \mathbf{t}) + \mathbf{w}_{\text{syn}} \cdot \mathbf{f}_{\text{syn}}(\mathbf{x}, \mathbf{t}, \mathbf{d}) \\ &= \mathbf{w}_{\text{pos}\oplus\text{syn}} \cdot \mathbf{f}_{\text{pos}\oplus\text{syn}}(\mathbf{x}, \mathbf{t}, \mathbf{d}) = \mathbf{w}_{\text{joint}} \cdot \mathbf{f}_{\text{joint}}(\mathbf{x}, \mathbf{t}, \mathbf{d}) \end{aligned} \tag{6}$$

where $\oplus$ denotes vector concatenation. Note that our joint model incorporates the same POS and syntactic features with the pipeline models. Under the joint model, the weights of POS and syntactic features, denoted by $\mathbf{w}_{\text{pos}\oplus\text{syn}}$ or $\mathbf{w}_{\text{joint}}$, are simultaneously learned. Therefore, they can interact with each other to determine an optimal joint result.

4.1 Decoding

Similar to (Li et al., 2011), we extend the parsing algorithm of (Carreras, 2007) using the idea of (Eisner, 2000) and propose a dynamic programming (DP) based decoding algorithm for our joint model. Figure 3 illustrates the basic DP structures and operations. The key idea is to augment the basic DP structures in the parsing algorithm (namely *spans*) with a few POS tags. A span means a partially built structure spanning a sub-sentence. For example, the left-side span in Figure 3(a), which is called an *incomplete span* and is denoted by $I_{(h,m,l)(t_h,t_m)}$, represents a partial tree spanning $w_h...w_m$ with w_h being tagged as t_h and w_m as t_m. The left-side span in Figure 3(b) is a complete span and is denoted by $C^{(e)(t_e)}_{(h,m)(t_h,t_m)}$.

The decoding algorithm works in a bottom-up fashion and combines two smaller spans into a larger one at each step. During combination, the newly-introduced features are incorporated and the score of the span is computed accordingly. For example, the operation in Figure 3(a) introduces five feature sets, i.e. $\mathbf{f}_{\text{dep}}(\mathbf{x}, t_h, t_m, h, m, l)$, $\mathbf{f}_{\text{sib}}(\mathbf{x}, t_h, t_m, t_s, h, m, l, s)$, $\mathbf{f}_{\text{grd}}(\mathbf{x}, t_h, t_m, t_g, h, m, l, g)$, $\mathbf{f}_{\text{pu}}(\mathbf{x}, t_m)$, and $\mathbf{f}_{\text{pb}}(\mathbf{x}, t_r, t_{r+1})$. And the operation in Figure 3(b) introduces one feature set $\mathbf{f}_{\text{grd}}(\mathbf{x}, t_h, t_m, t_g, h, m, l, g)$.[1] Note that in the above syntactic feature functions, several context POS tags are not encoded in the DP structures and therefore are not provided in the parameter lists, including $t_{h\pm1}, t_{m\pm1}, t_b, t_{s\pm1}, t_{g\pm1}$. For those, we use the most

[1] (Li et al., 2011) adopt a complex strategy to incorporate the POS features in their joint decoding algorithm. The way illustrated here is much easier and its correctness can be easily proved.

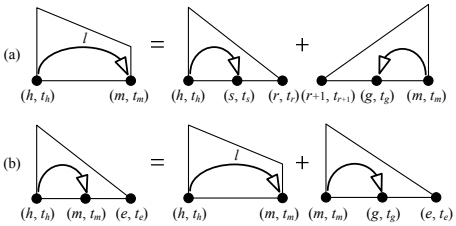


Figure 3: An illustration of the dynamic programming based decoding algorithm for our joint model. We omit the creation of right-headed spans for brevity.

likely POS tags provided by the baseline CRF tagging model following (Li et al., 2011). They find that this approximation substantially improves the efficiency of their joint models without accuracy loss. The time complexity of the algorithm is $O(|\mathscr{L}|n^4q^5)$, and the space complexity is $O(|\mathscr{L}|n^2q^2 + n^3q^3)$ where q is the tag number of each word ($\leq |\mathscr{T}|$).

POS tag pruning. Since the time complexity is high in terms of q, we follow (Li et al., 2011) and prune out the lower-probability POS tags for each word based on their marginal probabilities provided by our baseline CRF model. After pruning, each word has 1.4 candidate POS tags on average and the oracle tagging accuracy is 99.27% on CTB5. On PTB, each word has 1.2 candidate POS tags and the oracle is 99.71%.

Dependency pruning. The parsing time grows quickly with n. Therefore, we train a CRF-based first-order dependency parser to eliminate the unlikely dependencies following (Charniak and Johnson, 2005; Petrov and Klein, 2007; Koo and Collins, 2010). After pruning, 31.3% of the dependencies are left and the oracle dependency accuracy (UAS) is 99.77% on CTB5. On PTB, 28.9% of the dependencies are retained and the oracle UAS is 99.91%.

5 A separately passive-aggressive training algorithm

Online training has proven successful in several structured classification problems such as POS tagging (Collins, 2002) and parsing (McDonald et al., 2005; Zhang and Clark, 2011). Algorithm 1 shows the generic framework of online training when applied to our joint task. Online training iteratively traverses the entire training dataset and use one instance to update the feature weights at each time. First, the best result for the instance is found based on the current feature weights (line 6). Then, the feature weights are updated by comparing the best result and the gold-standard reference (line 7).

According to the update criterion, three different online training algorithms are widely used in parsing community, i.e. *averaged perceptron (AP)* (Collins, 2002), *passive-aggressive algorithm (PA)* (Crammer et al., 2003), and *margin infused relaxed algorithm (MIRA)* (Crammer and Singer, 2001). Previous work mostly adopts AP to train their joint models (Li et al., 2011; Hatori et al., 2011; Bohnet and Nivre, 2012). We compare AP and PA for our joint models and find similar accuracy in both tagging and parsing. Then we propose a variant of PA named as separately passive-aggressive algorithm (SPA) to improve the tagging accuracy. For the sake of conciseness, we do not make comparison with MIRA, since our preliminary results show that MIRA achieves similar performance to AP and PA.

Algorithm 1 Generic online training for joint POS tagging and dependency parsing

1. **Input:** Training Data $\mathbb{D} = \{(\mathbf{x}^{(j)}, \mathbf{t}^{(j)}, \mathbf{d}^{(j)})\}_{j=1}^{N}$
2. **Output:** $\mathbf{w}_{\text{joint}}$ $(\equiv \mathbf{w}_{\text{pos}\oplus\text{syn}})$
3. **Initialization:** $\mathbf{w}_{\text{joint}}^{(0)} = \mathbf{0}; \mathbf{v} = \mathbf{0}; k = 0$
4. **for** $i = 1$ **to** I **do** // *iterations*
5. **for** $j = 1$ **to** N **do** // *traverse the samples*
6. $(\hat{\mathbf{t}}, \hat{\mathbf{d}}) = \arg\max_{\mathbf{t},\mathbf{d}} \mathbf{w}_{\text{joint}}^{(k)} \cdot \mathbf{f}_{\text{joint}}(\mathbf{x}^{(j)}, \mathbf{t}, \mathbf{d})$ // *decode based on current feature weights.*
7. $\mathbf{w}_{\text{joint}}^{(k+1)} = \text{update } \mathbf{w}_{\text{joint}}^{(k)} \text{ with } (\mathbf{x}^{(j)}, \mathbf{t}^{(j)}, \hat{\mathbf{t}}, \mathbf{d}^{(j)}, \hat{\mathbf{d}})$ // *update weights according to some criterion.*
8. $\mathbf{v} = \mathbf{v} + \mathbf{w}_{\text{joint}}^{(k+1)}$
9. $k = k + 1$
10. **end for**
11. **end for**
12. $\mathbf{w}_{\text{joint}} = \mathbf{v}/(I \times N)$ // *average the weights*

All algorithms adopt the update direction of the distance between the reference feature vector $\mathbf{f}_{\text{joint}}(\mathbf{x}^{(j)}, \mathbf{t}^{(j)}, \mathbf{d}^{(j)})$ and the feature vector of the best result $\mathbf{f}_{\text{joint}}(\mathbf{x}^{(j)}, \hat{\mathbf{t}}, \hat{\mathbf{d}})$. However, different strategies are adopted to determine the update step. AP uses a constant update step of 1.

$$\texttt{AP}\left\{\mathbf{w}_{\text{joint}}^{(k+1)} = \mathbf{w}_{\text{joint}}^{(k)} + \mathbf{f}_{\text{joint}}(\mathbf{x}^{(j)}, \mathbf{t}^{(j)}, \mathbf{d}^{(j)}) - \mathbf{f}_{\text{joint}}(\mathbf{x}^{(j)}, \hat{\mathbf{t}}, \hat{\mathbf{d}})\right. \tag{7}$$

PA computes the update step τ_{joint} by considering the loss of the best result, the score distance, and the feature vector distance.

$$\texttt{PA}\left\{\begin{array}{l}\tau_{\text{joint}} = \dfrac{\text{Score}_{\text{joint}}(\mathbf{x}^{(j)}, \hat{\mathbf{t}}, \hat{\mathbf{d}}) - \text{Score}_{\text{joint}}(\mathbf{x}^{(j)}, \mathbf{t}^{(j)}, \mathbf{d}^{(j)}) + \rho_{\text{pos}}(\mathbf{t}^{(j)}, \hat{\mathbf{t}}) + \rho_{\text{syn}}(\mathbf{d}^{(j)}, \hat{\mathbf{d}})}{\|\mathbf{f}_{\text{joint}}(\mathbf{x}^{(j)}, \mathbf{t}^{(j)}, \mathbf{d}^{(j)}) - \mathbf{f}_{\text{joint}}(\mathbf{x}^{(j)}, \hat{\mathbf{t}}, \hat{\mathbf{d}})\|^2} \\ \mathbf{w}_{\text{joint}}^{(k+1)} = \mathbf{w}_{\text{joint}}^{(k)} + \tau_{\text{joint}}(\mathbf{f}_{\text{joint}}(\mathbf{x}^{(j)}, \mathbf{t}^{(j)}, \mathbf{d}^{(j)}) - \mathbf{f}_{\text{joint}}(\mathbf{x}^{(j)}, \hat{\mathbf{t}}, \hat{\mathbf{d}}))\end{array}\right. \tag{8}$$

where $\rho_{\text{pos}}(\mathbf{t}^{(j)}, \hat{\mathbf{t}})$ is the incorrect POS tag number in $\hat{\mathbf{t}}$ according to $\mathbf{t}^{(j)}$, and $\rho_{\text{syn}}(\mathbf{d}^{(j)}, \hat{\mathbf{d}})$ is the dependency error number in $\hat{\mathbf{d}}$ according to $\mathbf{d}^{(j)}$. Following (Johansson and Nugues, 2008), $\rho_{\text{syn}}(\mathbf{d}^{(j)}, \hat{\mathbf{d}})$ increases by 1 for an incorrect dependency and by 0.5 for a correct dependency with a wrong label. Theoretically, Eq. 8 computes the smallest update that makes the correct hypothesis outscores the returned highest-scoring hypothesis by the overall error.

We can see that AP and PA use the same update step for the POS features $\mathbf{f}_{\text{pos}}(.)$ and syntactic features $\mathbf{f}_{\text{syn}}(.)$. Therefore, the weights of the POS features and the syntactic features are of the same scale after training is completed. We argue that this is problematic since the number of the syntactic features are much larger than the number of the POS features. We find that in $\mathbf{f}_{\text{joint}}(\mathbf{x}^{(j)}, \hat{\mathbf{t}}, \hat{\mathbf{d}})$, $\mathbf{f}_{\text{syn}}(\mathbf{x}^{(j)}, \hat{\mathbf{t}}, \hat{\mathbf{d}})$ contains more than 3,000 non-zero features, whereas $\mathbf{f}_{\text{pos}}(\mathbf{x}^{(j)}, \hat{\mathbf{t}})$ only has less than 200 non-zero features on average. As a result, the model is dominated by the syntactic features and the POS features would play a very limited role in determining the best joint result. Actually, we find that the POS features contribute a very small part to the score of the best joint results when trained with AP or PA (see Figure 5).

After several empirical trials, we find that we can raise the score contributed by the POS features by increasing the update step of the POS features. Furthermore, we find that we can elegantly achieve this goal by slightly modifying the update formulas of PA. We name the proposed training algorithm as the *separately passive-aggressive algorithm (SPA)*. SPA separately

computes two update steps for the POS features and the syntactic features.

$$\text{SPA}\begin{cases} \tau_{\text{pos}} = \dfrac{\text{Score}_{\text{pos}}(\mathbf{x}^{(j)}, \hat{\mathbf{t}}) - \text{Score}_{\text{pos}}(\mathbf{x}^{(j)}, \mathbf{t}^{(j)}) + \rho_{\text{pos}}(\mathbf{t}^{(j)}, \hat{\mathbf{t}})}{\|\mathbf{f}_{\text{pos}}(\mathbf{x}^{(j)}, \mathbf{t}^{(j)}) - \mathbf{f}_{\text{pos}}(\mathbf{x}^{(j)}, \hat{\mathbf{t}})\|^2} \\ \tau_{\text{syn}} = \dfrac{\text{Score}_{\text{syn}}(\mathbf{x}^{(j)}, \hat{\mathbf{t}}, \hat{\mathbf{d}}) - \text{Score}_{\text{syn}}(\mathbf{x}^{(j)}, \mathbf{t}^{(j)}, \mathbf{d}^{(j)}) + \rho_{\text{syn}}(\mathbf{d}^{(j)}, \hat{\mathbf{d}})}{\|\mathbf{f}_{\text{syn}}(\mathbf{x}^{(j)}, \mathbf{t}^{(j)}, \mathbf{d}^{(j)}) - \mathbf{f}_{\text{syn}}(\mathbf{x}^{(j)}, \hat{\mathbf{t}}, \hat{\mathbf{d}})\|^2} \\ \mathbf{w}_{\text{pos}}^{(k+1)} = \mathbf{w}_{\text{pos}}^{(k)} + \tau_{\text{pos}}(\mathbf{f}_{\text{pos}}(\mathbf{x}^{(j)}, \mathbf{t}^{(j)}) - \mathbf{f}_{\text{pos}}(\mathbf{x}^{(j)}, \hat{\mathbf{t}})) \\ \mathbf{w}_{\text{syn}}^{(k+1)} = \mathbf{w}_{\text{syn}}^{(k)} + \tau_{\text{syn}}(\mathbf{f}_{\text{syn}}(\mathbf{x}^{(j)}, \mathbf{t}^{(j)}, \mathbf{d}^{(j)}) - \mathbf{f}_{\text{syn}}(\mathbf{x}^{(j)}, \hat{\mathbf{t}}, \hat{\mathbf{d}})) \end{cases} \quad (9)$$

Analogous to PA, Eq. 9 separately finds the smallest update to the POS feature weights that makes the POS score of the correct hypothesis higher than the POS score of the returned highest-scoring hypothesis by the POS error, and the smallest update to the syntactic feature weights that makes the syntactic score of the correct hypothesis higher than the syntactic score of the returned highest-scoring one by the syntactic error. Note that Eq. 9 sometimes leads to negative τ_{pos} or τ_{syn}, because the correct hypothesis already has a POS or syntactic subscore larger than the 1-best one but the deficiency is entirely confined to the other subscore. However, such cases are rare. We just set the update step to zero when it is negative.

Since $\mathbf{f}_{\text{pos}}(.)$ contains much less non-zero features than $\mathbf{f}_{\text{syn}}(.)$, $\|\mathbf{f}_{\text{pos}}(\mathbf{x}^{(j)}, \mathbf{t}^{(j)}) - \mathbf{f}_{\text{pos}}(\mathbf{x}^{(j)}, \hat{\mathbf{t}})\|^2$ is much smaller than $\|\mathbf{f}_{\text{syn}}(\mathbf{x}^{(j)}, \mathbf{t}^{(j)}, \mathbf{d}^{(j)}) - \mathbf{f}_{\text{syn}}(\mathbf{x}^{(j)}, \hat{\mathbf{t}}, \hat{\mathbf{d}})\|^2$. Therefore, τ_{pos} is much larger than τ_{syn}. Our experiments show that τ_{pos} is about 10 times larger than τ_{syn} on average. This means that the POS features are updated in much larger step than the syntactic features. As a result, the POS features play a more important role in the joint models. We find that the POS score becomes much closer to the syntactic score in the best joint results (see Figure 5).

As discussed in Section 1, (Li et al., 2011) find that their joint models trained with AP are good at resolving syntax-sensitive POS ambiguities like {VV,NN}, whereas their baseline sequential POS tagging model does well in disambiguating the syntax-insensitive ones like {NN,NR}. We believe that it is because the discriminative power of the POS features in resolving such syntax-insensitive POS ambiguities are suppressed in the joint models when trained with AP or PA. Compared with AP and PA, SPA raises the weight of the POS features and can better utilize the disambiguation power of both the POS and syntactic features, leading to large tagging accuracy boost. On the other hand, better tagging results can further help parsing.

6 Experiments

Data. We conduct experiments on CTB5 (Xue et al., 2005). Following the standard practice, we adopt the data split of (Duan et al., 2007; Zhang and Clark, 2008b; Huang and Sagae, 2010) and adopt Penn2Malt[2] for constituent-to-dependency conversion with the head-finding rules of (Zhang and Clark, 2008b).

We also evaluate our models on another version of CTB5 used in (Bohnet and Nivre, 2012) to compare with their joint model. We thank Bernd Bohnet for sharing their dataset. We refer to their dataset as CTB5-Bohnet. We carefully compare CTB5 with CTB5-Bohnet and find that except for the mismatch of about 30 sentence, the datasets differ in both dependency structures and dependency labels. After discussions with Bernd Bohnet, we find out that they adopt Yue Zhang's constituent-to-dependency conversion tool[3] whereas we use Penn2Malt for

[2] http://w3.msi.vxu.se/~nivre/research/Penn2Malt.html
[3] http://sourceforge.net/projects/zpar/files/0.3/

Corpus	Train	Dev	Test
CTB5	16,091	803	1,910
CTB5-Bohnet	16,069	803	1,905
PTB	39,832	1,346	2,416

Table 2: Data used in this work (in sentence number).

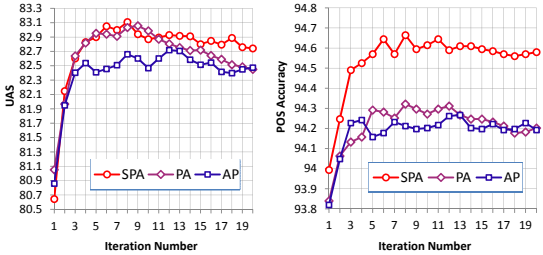


Figure 4: Training curves of UAS and POS tagging accuracy of the unlabeled joint model using SPA/PA/AP on the CTB5 development dataset.

CTB5, although the head-finding rules are the same.

For English, we adopt PTB (sec 02-21 for training, sec 24 for development, sec 23 for test) and convert the original bracketed structures into dependencies using Penn2Malt with its default head-finding rules. Table 2 summarizes the data sets used in the present work.

Evaluation metrics. We use the standard *POS tagging accuracy (POS)* to evaluate POS tagging. For dependency parsing, we use *unlabeled attachment score (UAS)* and *labeled attachment score (LAS)* (all excluding punctuation) (Hajič et al., 2009).

6.1 Experiments on CTB5

6.1.1 Comparison of the three training algorithms

The curves in Figure 4 show the performance of the unlabeled joint model on the development dataset using SPA/PA/AP after each iteration during training. We can see that the PA outperforms AP in both UAS and tagging accuracy. SPA achieves a slightly higher peak UAS than PA, and substantially outperforms both PA and AP in tagging accuracy. In addition, Figure 4 empirically indicates that SPA can converge as fast as AP and PA. We leave the theoretic proof of the convergence of SPA in the future work.

To better understand the reason behind the improvement in tagging accuracy, we try to analyze the two-part model scores, i.e. the POS score $\mathsf{Score}_{\mathrm{pos}}(.)$ and the syntactic score $\mathsf{Score}_{\mathrm{syn}}(.)$. After each iteration, we parse the development dataset using the current weights, and then compute and average each part of the model score. Figure 5 shows the results. For PA and AP, the scale of the POS score is about 1/50 ($10^{1.7} = 50$) of the syntactic score, which means the POS features play an insignificant role in determining the joint result. Obviously, SPA can raise the weight of the POS features, as the POS score is about 1/8 ($10^{0.9} = 8$) of the syntactic score.

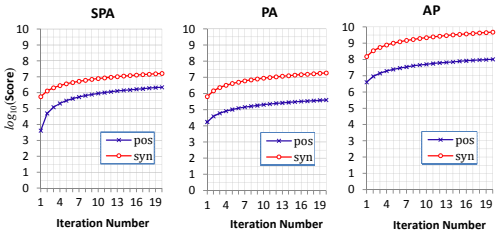


Figure 5: Model score analysis on the CTB5 development dataset. "pos" and "syn" refer to $\text{Score}_{\text{pos}}(.)$ and $\text{Score}_{\text{syn}}(.)$ (see Eq. 6).

	UAS	POS
SPA	81.21	94.51
PA	80.98	94.18
AP	80.94	94.17
pipeline	80.22	93.88

Table 3: Performance of three training algorithms on the CTB5 test dataset.

As a result, the POS features can make more contributions, and the discriminative power of the syntactic features in resolving the syntax-sensitive ambiguities and the power of the POS features in the syntax-insensitive POS ambiguities can be better balanced. We believe this is the reason behind the improvement in tagging accuracy.

Table 3 reports the results of our unlabeled joint model trained with SPA/PA/AP on the CTB5 test dataset. We adopt the parameters which lead to the best parsing accuracy on the development dataset as shown in Figure 4. We also present the performance of our pipelined unlabeled parsing model trained with PA. Different to (Li et al., 2011), our joint models trained with AP or PA both outperform our baseline POS tagging model by about 0.3%. We believe that this may be due to the richer syntactic features incorporated in our joint models. Moreover, SPA can outperform both PA and AP in tagging accuracy by more than 0.3%, which is an error reduction of 5%. Meanwhile, SPA also improves UAS by about 0.2% since better tagging results help parsing. This again demonstrates the effectiveness of the proposed SPA.

6.1.2 Main results

Table 4 shows the final results on the CTB5 test set. Three sets of results are presented. For "gold POS" and "pipeline", we train the parsing models with PA. For "joint" models, we adopt SPA. For each case, we train our models with two different settings, one labeled and one unlabeled, to study the effect of dependency labels. We can see that we achieve best results on all three cases. Specifically, a few interesting conclusions can be drawn.

- Labeling the dependencies during parsing improves UAS by 0.6%/0.5% in the pipeline and joint cases and by 0.2% when using gold-standard POS tags. Dependency labels also slightly help POS tagging in the joint case.
- Compared with the pipeline models, the joint models with SPA can largely improve POS

	Models	LAS	UAS	POS
joint	Ours (labeled)	**79.01**	**81.67**	**94.60**
	Ours (unlabeled)	—	81.21	94.51
	(Li et al., 2011)	—	80.74	93.08
	(Hatori et al., 2011)	—	81.33	93.94
pipeline	Ours (labeled)	**77.80**	**80.82**	**93.88**
	Ours (unlabeled)	—	80.22	
	(Li et al., 2011)	—	79.29	93.51
	(Hatori et al., 2011)	—	78.04	93.82
gold POS	Ours (labeled)	**85.36**	**86.76**	
	Ours (unlabeled)	—	86.55	
	(Li et al., 2011)	—	86.18	100.0
	(Hatori et al., 2011)	—	85.96	
	(Zhang and Nivre, 2011)	84.4	86.0	
	(Huang and Sagae, 2010)	—	85.20	

Table 4: Final results on the CTB5 test dataset.

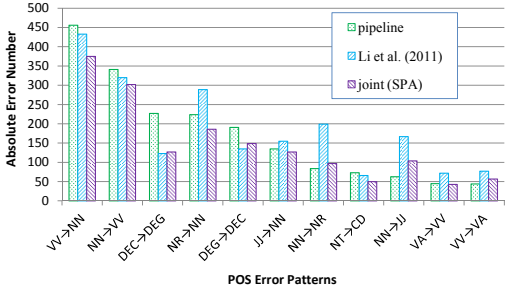


Figure 6: Statistics for different POS error patterns on the CTB5 test dataset.

tagging accuracy by 0.6-0.7%, which is an error reduction of 10%.

- Our joint models largely outperforms their pipeline counterparts by 0.9-1.0% in UAS and 1.2% in LAS.

6.1.3 Analysis

Figure 6 compares different models on a number of high-frequency POS error patterns. An error pattern X→Y means that the focus word, whose true tag is X, is assigned a tag Y. We thank the authors of (Li et al., 2011) for sharing their results. The joint model of (Li et al., 2011) reduces errors for syntax-sensitive ambiguities such as {(DEC,DEG} and {NN,VV}, but largely increases errors for syntax-insensitive ambiguities like {NN,NR}, {NN,JJ}, and {VV,VA}, which can explain its low tagging accuracy. Compared with (Li et al., 2011), our joint model trained with SPA does much better in resolving the syntax-insensitive ambiguities and achieves similar performance to the baseline CRF-based tagging model on those patterns. In summary, we can conclude that our joint model trained with SPA performs similarly to the baseline CRF-

Models		LAS	UAS	POS
joint	Ours (labeled)	**80.18**	**83.14**	**94.71**
	Ours (unlabeled)	—	82.37	94.65
	(Bohnet and Nivre, 2012)	77.91	81.42	93.24
pipeline	Ours (labeled)	**79.01**	**82.07**	**93.89**
	Ours (unlabeled)	—	81.38	
	(Bohnet and Nivre, 2012)	76.79	80.33	92.81

Table 5: Final results on the CTB5-Bohnet test dataset.

Models		LAS	UAS	POS
joint	Ours (labeled)	**92.44**	**93.52**	**97.62**
	Ours (unlabeled)	—	93.12	**97.62**
	(Bohnet and Nivre, 2012)	**92.44**	93.38	97.33
	(Bohnet and Nivre, 2012) †	*92.68*	*93.67*	*97.42*
pipeline	Ours (labeled)	**92.00**	93.14	97.16
	Ours (unlabeled)	—	92.85	
	(Bohnet and Nivre, 2012)	91.71	92.79	**97.28**
	(Zhang and Nivre, 2011)	—	92.9	—
	(Martins et al., 2010)	—	**93.26**	—
	(Koo and Collins, 2010)	—	93.04	—
	(Huang and Sagae, 2010)	—	92.1	—
	(Koo et al., 2008) †	—	*93.16*	—
	(Carreras et al., 2008) †	—	*93.5*	—
	(Suzuki et al., 2009) †	—	*93.79*	—

Table 6: Final results on the PTB test dataset. Results marked by † use additional resources and are therefore not directly comparable to the others.

based tagging model on the syntax-insensitive ambiguities and meanwhile similarly to the joint model of (Li et al., 2011) on the syntax-sensitive ambiguities. This demonstrates that SPA can better balance the discriminative power of both the POS and syntactic features.

6.2 Experiments on CTB5-Bohnet

We conduct experiments on CTB5-Bohnet to make comparison with the recent results of (Bohnet and Nivre, 2012). Table 5 presents the results. We can see that our pipeline and joint models largely outperform those of (Bohnet and Nivre, 2012). Moreover, the parsing accuracies on this dataset are much higher than those on CTB5 due to the different conversion strategy. We will study the reasons in the future work. Again, we find that labeled parsing can largely improve UAS.

6.3 Experiments on PTB

To find out the effect of the joint models on English, we conduct experiments on PTB. Table 6 shows the results. Several state-of-the-art results are also presented. The pipeline models on English suffer from less error propagation problem than that on Chinese, as the POS tag-

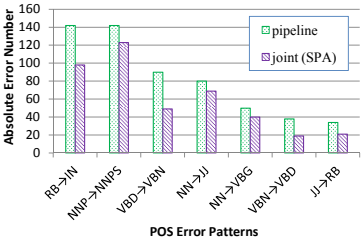


Figure 7: Statistics for different POS error patterns on the PTB test dataset.

ging accuracy on English is much higher (97% vs. 94%). However, we still find that our joint models can obtain a 0.3-0.4% gain in UAS, a 0.4% gain in LAS, and a 0.5% gain in tagging accuracy. Similar to the findings on the Chinese datasets, we demonstrate that labeled parsing can boost UAS by 0.3-0.4% on English. Our labeled joint model outperforms the one of (Bohnet and Nivre, 2012) by 0.1% in UAS and 0.3% in tagging accuracy. Actually, our labeled joint model achieves the best parsing and tagging accuracy on PTB.

We compare the POS tagging results of the unlabeled joint model and the baseline tagging model. Figure 7 shows a few high-frequency error patterns. We can see that the joint model can help resolve a variety of POS ambiguities. For example, the error number of RB→IN (adverbs wrongly tagged as prepositions or subordinating conjunctions) are reduced from 142 to 98, which is a 31% error reduction. Also, the ambiguous pair {VBD, VBN} (verbs of past tense, verbs of past participle) are much better disambiguated by the joint model.

Conclusions

This paper presents a separately passive-aggressive training algorithm (SPA) for joint POS tagging and labeled dependency parsing models. We show that SPA can more properly learn the feature weights than the averaged perceptron (AP) and the passive aggressive algorithm (PA) and can better balance the discriminative power of the POS feature in resolving the syntax-insensitive POS ambiguities and the power of the syntactic features in resolving the syntax-sensitive ambiguities, leading to large tagging accuracy improvement. On the other hand, better POS tagging can further help parsing.

For future work, we are interested in studying SPA from the theoretic perspective and try to provide more insights and justifications of its effectiveness in training the joint models. Besides, although this paper focuses on graph-based joint models, we believe that SPA can also be applied to transition-based joint models (Hatori et al., 2011; Bohnet and Nivre, 2012).

Acknowledgements

We thank Meishan Zhang, for suggesting the easier way to incorporate the POS features during joint decoding, and the anonymous reviewers, for their valuable comments which lead to better understanding of the proposed SPA. This work was supported by National Natural Science Foundation of China (NSFC) via grant 61133012, the National “863” Major Projects via grant 2011AA01A207, and the National “863” Leading Technology Research Project via grant 2012AA011102.

References

Auli, M. and Lopez, A. (2011). A comparison of loopy belief propagation and dual decomposition for integrated ccg supertagging and parsing. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 470–480, Portland, Oregon, USA. Association for Computational Linguistics.

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

Bohnet, B. and Nivre, J. (2012). A transition-based system for joint part-of-speech tagging and labeled non-projective dependency parsing. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 1455–1465, Jeju Island, Korea. Association for Computational Linguistics.

Carreras, X. (2007). Experiments with a higher-order projective dependency parser. In *Proceedings of EMNLP/CoNLL*, pages 141–150.

Carreras, X., Collins, M., and Koo, T. (2008). Tag, dynamic programming, and the perceptron for efficient, feature-rich parsing. In *CoNLL 2008: Proceedings of the Twelfth Conference on Computational Natural Language Learning*, pages 9–16, Manchester, England. Coling 2008 Organizing Committee.

Charniak, E. and Johnson, M. (2005). Coarse-to-fine n-best parsing and maxent discriminative reranking. In *Proceedings of ACL-05*, pages 173–180.

Che, W., Spitkovsky, V., and Liu, T. (2012). A comparison of chinese parsers for stanford dependencies. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 11–16, Jeju Island, Korea. Association for Computational Linguistics.

Collins, M. (2002). Discriminative training methods for hidden markov models: Theory and experiments with perceptron algorithms. In *Proceedings of EMNLP 2002*.

Crammer, K., Dekel, O., Keshet, J., Shalev-Shwartz, S., and Singer, Y. (2003). Online passive aggressive algorithms. In *Procedings of NIPS 2003*.

Crammer, K. and Singer, Y. (2001). Ultraconservative online algorithms for multiclass problems. *Journal of Machine Learning Research*, 3.

Duan, X., Zhao, J., , and Xu, B. (2007). Probabilistic models for action-based Chinese dependency parsing. In *Proceedings of ECML/ECPPKDD*.

Eisner, J. (2000). Bilexical grammars and their cubic-time parsing algorithms. In *Advances in Probabilistic and Other Parsing Technologies*, pages 29–62.

Hajič, J., Ciaramita, M., Johansson, R., Kawahara, D., Martí, M. A., Màrquez, L., Meyers, A., Nivre, J., Padó, S., Štěpánek, J., Straňák, P., Surdeanu, M., Xue, N., and Zhang, Y. (2009). The CoNLL-2009 shared task: Syntactic and semantic dependencies in multiple languages. In *Proceedings of CoNLL 2009*.

Hatori, J., Matsuzaki, T., Miyao, Y., and Tsujii, J. (2011). Incremental joint pos tagging and dependency parsing in chinese. In *Proceedings of 5th International Joint Conference on Natural Language Processing*, pages 1216–1224, Chiang Mai, Thailand. Asian Federation of Natural Language Processing.

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

Johansson, R. and Nugues, P. (2008). Dependency-based semantic role labeling of PropBank. In *EMNLP-2008*.

Koo, T., Carreras, X., and Collins, M. (2008). Simple semi-supervised dependency parsing. In *Proceedings of ACL-08: HLT*, pages 595–603, Columbus, Ohio. Association for Computational Linguistics.

Koo, T. and Collins, M. (2010). Efficient third-order dependency parsers. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 1–11, Uppsala, Sweden. Association for Computational Linguistics.

Lafferty, J., McCallum, A., and Pereira, F. (2001). Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of ICML 2001*, pages 282–289.

Lee, J., Naradowsky, J., and Smith, D. A. (2011). A discriminative model for joint morphological disambiguation and dependency parsing. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 885–894, Portland, Oregon, USA. Association for Computational Linguistics.

Li, Z., Zhang, M., Che, W., Liu, T., Chen, W., and Li, H. (2011). Joint models for chinese pos tagging and dependency parsing. In *EMNLP 2011*, pages 1180–1191.

Martins, A., Smith, N., Xing, E., Aguiar, P., and Figueiredo, M. (2010). Turbo parsers: Dependency parsing by approximate variational inference. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 34–44, Cambridge, MA. Association for Computational Linguistics.

McDonald, R., Crammer, K., and Pereira, F. (2005). Online large-margin training of dependency parsers. In *Proceedings of ACL 2005*, pages 91–98.

Petrov, S. and Klein, D. (2007). Improved inference for unlexicalized parsing. In *Proceedings of NAACL 2007*.

Ratnaparkhi, A. (1996). A maximum entropy model for part-of-speech tagging. In *Proceedings of EMNLP 1996*.

Rush, A. M., Sontag, D., Collins, M., and Jaakkola, T. (2010). On dual decomposition and linear programming relaxations for natural language processing. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 1–11.

Smith, D. A. and Eisner, J. (2008). Dependency parsing by belief propagation. In *Proceedings of EMNLP 2008*, pages 145–156.

Suzuki, J., Isozaki, H., Carreras, X., and Collins, M. (2009). An empirical study of semi-supervised structured conditional models for dependency parsing. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 551–560, Singapore. Association for Computational Linguistics.

Xue, N., Xia, F., Chiou, F.-D., and Palmer, M. (2005). The Penn Chinese Treebank: Phrase structure annotation of a large corpus. In *Natural Language Engineering*, volume 11, pages 207–238.

Zhang, Y. and Clark, S. (2008a). Joint word segmentation and POS tagging using a single perceptron. In *Proceedings of ACL-08: HLT*, pages 888–896.

Zhang, Y. and Clark, S. (2008b). A tale of two parsers: Investigating and combining graph-based and transition-based dependency parsing. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 562–571, Honolulu, Hawaii. Association for Computational Linguistics.

Zhang, Y. and Clark, S. (2011). Syntactic processing using the generalized perceptron and beam search. *Computational Linguistics*, 37(1):105ÍC151.

Zhang, Y. and Nivre, J. (2011). Transition-based dependency parsing with rich non-local features. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 188–193, Portland, Oregon, USA. Association for Computational Linguistics.

Graph-based Multi-tweet Summarization Using Social Signals

LIU XiaoHua[1,2] *LI YiTong*[3] *WEI FuRu*[2] *ZHOU Ming*[2]
(1) School of Computer Science and Technology, Harbin Institute of Technology, Harbin, 150001, China
(2) Microsoft Research Asia, Beijing, 100190, China
(3) School of Electronic and Information Engineering, Beihang University, Beijing, 100191, China
{xiaoliu,v-yitli,fuwei,mingzhou}@microsoft.com

Abstract

We study the multi-tweet summarization task, which aims to find representative tweets from a given set of tweets. Multi-tweet summarization allows people to quickly grasp the essential meaning of a large number of tweets. It can also be used as a pre-processing component for information extraction tasks on tweets. The challenge of this task lies in computing a tweet's salience score with little information in a single tweet. We propose a graph-based multi-tweet summarization system that incorporates social network features, which make up for the information shortage in a tweet. Another distinguished feature of our system is that tweet readability and user diversity are considered. We evaluate our system on a manually annotated dataset, and show that our system outperforms the state-of-the-art baseline. We further evaluate our method in a real scenario of summarization of Twitter search results and demonstrate its effectiveness.

Title and Abstract in another language (Chinese)

基于图模型和社会关系特征的推特消息摘要方法

本文考察多推特消息摘要任务。其目的是帮助用户快速了解大量推特消息的基本意思，或在信息抽取前对推特消息做预处理。 该任务的主要挑战是：单条推特消息往往不能提供足够的信息来计算它的显著度。本文提出基于图模型的方法，考虑社会关系网相关的特征、可读性及用户的多样性来克服单条推特消息的不足。 在人工标注的数据集以及推特搜索上，该方法均展示了其有效性。

Keywords: Graph, Summarization, Social signals, Tweets.

Keywords in Chinese: 图模型，摘要，社会关系网特征，推特消息 .

Proceedings of COLING 2012: Technical Papers, pages 1699–1714,
COLING 2012, Mumbai, December 2012.

1 Introduction

Twitter[1] is a microblogging and social networking service with a huge number of users and is continuously growing at a tremendous rate. Tweets, short messages shared through Twitter with less than 140 characters, have become an important repository of real-time information. However, it is often inefficient for people to consume a large number of tweets, owing to the noise prone nature of tweets[2]. We propose the task of multi-tweet summarization to overcome this obstacle: eliminating redundancy and noise while keeping the essential information for a given set of tweets. As an illustrative example, consider the following tweets, returned from Twitter search related to the query "obama":

- "@morrowchris: Christmas card from the Obama's :) http://pic.twitter.com/F3VU52io" thts special.
- Christmas card from the Obama's :) http://pic.twitter.com/7xqBQdBV
- RT @liberalease: RT if you like President Obama better than ALL OF GOP COMBINED. / That was easy :)
- Obama did it! The war is OFFICIALLY OVER! #WelcomeHomeTroops! :)
- obama really brought the troops home :)

The first and the second tweet are almost the same; the third tweet is a private conversation, thus not meaningful for the general audience; and the last two talk about similar things. After summarization, the expected outputs are as follows, which keep the main information with noises removed.

- Christmas card from the Obama's :) http://pic.twitter.com/7xqBQdBV
- obama really brought the troops home :)

We advocate that multi-tweet summarization can play a critical role for many tweet related studies, which have attracted increasing research interests in recent days, such as named entity recognition (NER) and sentiment analysis (SA) for tweets. One common issue of these tasks is the scalability challenge, which means they are required to process a huge number of tweets each day. Summarization system is then an ideal pre-processing component in the sense that it helps to reduce the number of tweets for processing without much lose of information.

Multi-document summarization has been well studied, and a couple of systems have been developed. We test LexRank (Erkan and Radev, 2004), a state-of-the-art summarization system and find a sharp drop of ROUGE-2, from 0.3894 on news to 0.2871 on tweets. This can be largely attributed to the short and noise prone nature of tweets, which causes a single tweet to be insufficient to provide reliable information to compute its salience score. We develop a graph-based summarization system that aggregates social signals, i.e., re-tweeted times and follower numbers to handle this challenge. More specifically, the translation probability from one tweet to the other depends on both the similarity between the two tweets and the social network features associated with the second tweet. This largely differentiates our system from existing studies, such as the work of Sharifi et al. (2010), which uses only tweet-level content features (e.g., keywords) to select representative sentences.

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

[2] Noise in tweets means ill-formed words or sentences in tweets.

Besides utilizing social signals, our system has two additional features. Firstly, the readability feature is introduced to the graph model to reduce the chance of tweets hard to read to appear in the summarization. Several factors are considered while computing a tweet's readability, including: 1) The number of out-of-vocabulary (OOV) words; 2) the number of words; and 3) the number of abnormal symbols, e.g., "!,,),(,*". Secondly, while selecting representative tweets using an alternative of the Maximal Marginal Relevance (MMR) (Goldstein et al., 1999) algorithm, our system penalizes tweets which are selected from a same twitter account, to achieve diversity among users.

We collect 100 sets of tweets, each of which is related to a trending topic. For each set of tweets, we manually select representative tweets as the summarization, forming the gold-standard dataset. We show that our system compares favorably to the LexRank (Erkan and Radev, 2004) baseline in terms of ROUGE-1 and ROUGE-2. We also show the positive effects of considering social signals, readability and user diversity, respectively. To understand how well our method performs in a real scenario, we apply our system to summarize Twitter search results, and consistently observe an improvement of user's satisfaction of Twitter search in a serials of user studies. It is worth mentioning that our proposed summarization system can be easily adapted to other social contents that are short and noisy but with rich social evidences, e.g., Facebook updates or short messages shared through Facebook.

Contributions of this work are summarized as follows.

1. We propose a graph-based multi-tweets summarization system that leverages social network features, readability and user diversity for selecting representative tweets.
2. We evaluate our system on a human annotated dataset and show our system outperforms the baseline. We conduct extensive user studies and show our system considerably improves user's satisfaction of Twitter search.

The rest of this paper is organized as follows. Section 2 introduces related work. Section 3 defines the task. Section 4 describes the baseline. Section 5 details our method and Section 6 evaluates our method. Section 7 demonstrates the application of the proposed method to the Twitter search. Finally, Section 8 concludes with a discussion of future work.

2 Related Work

Two categories of research are highly related to our work: multi-document summarization and recent studies of tweets.

2.1 Multi-document Summarization

Abstraction and selection are two strategies employed for multi-document summarization. The former involves information fusion (Barzilay et al., 1999; Xie et al., 2008), sentence compression (Knight and Marcu, 2002), and reformulation (Barzilay et al., 2001; Saggion, 2011); while the latter requires computing salience scores of some units (e.g., sentences, paragraphs) and extracting those with highest scores with redundancy removed. NewsBlaster[3] and our method are examples of abstraction and selection based methods, respectively. We choose the selection strategy because it is relatively simpler, e.g., not requiring

[3]http://newsblaster.cs.columbia.edu

language generation to produce a grammatical and coherent summary, and better suites the scenario of tweet summarization. Note that our method considers each tweet as the unit for summarization, which often cannot provide reliable information to compute the salience. This is one main difference between our system and the existing studies.

Existing selection-based methods can be divided into four categories: cluster based (Hardy et al., 2002), centroid based (Radev, 2004), graph based (Erkan and Radev, 2004; Mani and Bloedorn, 1999), and machine learning based (Neto et al., 2002). Cluster-based methods first separate a document set into several groups, each representing a sub-topic. Then representative sentences are selected from each group, and finally those sentences are put together as the summarization of the whole document set. Centroid-based methods compute the center of a document set, and then use the similarity between the sentence and the center as the sentence salience score. Graph-based methods construct a graph, where a vertex denotes a sentence and the weight of an edge represents the similarity between the two sentences connected by the edge. Then, similar to PageRank (Page et al., 1998), a Markov Random Walk is performed on the graph to compute the salience score of every sentence. Machine learning based methods model the summarization process as a classification problem: Whether or not a sentence should be selected as summary sentences. A proper classifier, e.g., a Naive Bayes classifier, is learnt statistically from the training data. There are methods between those categories. For example, Wan and Yang (2008) consider cluster level information, i.e., the importance of the cluster and the relevance of sentence to the cluster, for computing sentence salience score. Motivated by LexRank (Erkan and Radev, 2004), we adopt graph based methods. Differently, our system incorporates rich social network features and considers readability to compute salience score of every tweet.

Most existing studies focus on formal texts such as news. However, exceptions exist. For instance, Qazvinian and Radev (2010, 2008) study the problem of summarizing a scientific paper. They propose a clustering approach where communities in the citation summary's lexical network are formed and sentences are extracted from separate clusters. Sharifi et al. (2010) use the Phrase Reinforcement algorithm to generate one-line summary for a collection of tweets related to a topic. Though our method is also designed for tweets, there are several significant differences. Firstly, our method does not assume that the input tweets are about a topic. Secondly, our method selects representative tweets by exploiting social network features, readability and keywords. In contrast, Sharifi et al. (2010) find the most commonly used phrases that encompass the topic phrase.

The maximal marginal relevance (MMR) measure (Goldstein et al., 1999) is widely used in summarization to simultaneously rewards relevant sentences and penalizes redundant ones by considering a linear combination of two similarity measures. We adopt an alternative implementation of MMR, which greedily selects the next most salient tweet whose similarity to any previously selected tweet is less than a threshold and that the number of tweets from the same account is also below a threshold.

2.2 Research on Tweets

Recently we have witnessed growing research interests in tweets. For example, Kwak et al. (2010) first study the topological characteristics of Twitter and its power as a new medium for information sharing; various studies are carried out on how Twitter is used by leg-

islators, particularly by members of the United States Congress (Golbeck et al., 2010), by city police departments in large U.S. cities (Heverin and Zach, 2010), and by scholars (Priem and Costello, 2010); Jansen et al. (2009) report research results investigating microblogging as a form of electronic word-of-mouth for sharing consumer opinions concerning brands; Heverin and Zach (2010) give insights into why particular events resonate with the population. All the above studies indicate the critical role of tweets as a dynamic information source.

There is another line of studies aiming to help people to efficiently access tweets. For instance, Finin et al. (2010) annotate named entities in tweets by exploiting Amazon's Mechanical Turk service[4] and CrowdFlower[5]; Liu et al. (2011) propose to combine a K-Nearest Neighbors (KNN) classifier with a linear Conditional Random Fields (CRF) model under a semi-supervised learning framework to recognize named entities in tweets; Liu et al. (2010) conduct a pilot study of Semantic Role Labeling on tweets; Sankaranarayanan et al. (2009) extract breaking news from tweets to build a news processing system, called TwitterStand; Duan et al. (2010) give an empirical study on learning to rank of tweets; Weng et al. (2010) propose TwitterRank, an extension of the PageRank algorithm to identify influential twitter accounts; O'Connor et al. (2010) present TweetMotif which groups tweets by frequent significant terms; Inouye and Kalita (2011) compare several tweet summarization algorithms that use text features like TFIDF to compute the similarity between any two tweet; Sharifi et al. (2010) exploit the Phrase Reinforcement Algorithm to find the most commonly used phrases that encompass the given topic phrase, based on which salient sentences are selected; and most recently, Efron (2011) offers a comprehensive introduction to the problems encountered by researchers and developers of Information Retrieval (IR) systems in microblog settings. Our work develops this line of research, with its focus on summarizing multiple tweets using a novel summarization system which considers social network related information, such as re-tweeted times and follower numbers, and partially addresses some of the research challenges discussed by Efron (2011).

3 Task Description

A tweet is a short text message containing no more than 140 characters. Here is an example: "mycraftingworld: #Win Microsoft Office 2010 Home and Student *2Winners* #Contest from @office and @momtobedby8 #Giveaway http://bit.ly/bCsLOr ends 11/14", where "mycraftingworld" is the name of the user who published this tweet. Words beginning with the "#" character, like ""#Win", "#Contest" and "#Giveaway", are hash tags, usually indicating the topics of the tweet; words starting with "@", like "@office" and "@momtobedby8", represent user names, and "http://bit.ly/bCsLOr" is a shortened link.

Given a collection of tweets, our task is to output a subset of no more than M representative tweets that best preserve important information in the original set. The number of input tweets varies from hundreds, e.g., for tweets related to a given query or user, to tens of millions, e.g., for tweets in a given time range. M is a systematic parameter whose value is set according to Formula 1, where α is set to 0.05 and n is the input size. In this study, we limit our attention to English tweets only, though our method is almost language

[4]https://www.mturk.com/mturk/
[5]http://crowdflower.com/

independent[6].

$$M = \lceil \alpha \cdot n \rceil \quad (1)$$

An example of tweet summarization is given below. The input collection is:

- Finally got Windows 8 on my laptop as a primary OS. Sort of my way of welcoming the new holidays :)
- I Just Got Windows 8 . Whoooo ! :)
- ...
- Windows 8 Picture Passwords: Smudging Your Finger for Security interesting alternative...it's not a replacement :)
- Windows 8 will have picture password sign in http://is.gd/JoXYHx

Suppose $M = 2$, the generated summarization is:

- I Just Got Windows 8 . Whoooo ! :)
- Windows 8 will have picture password sign in http://is.gd/JoXYHx

From this example, it can be seen that selection based multi-tweet summarization allows users to quickly grasp the essential information conveyed in the input tweets, which are prone to noise and rich in redundancy. This is exactly the main motivations of this study.

4 The Baseline

We choose an adapted LexRank as the baseline, considering that LexRank outperforms both centroid-based methods and other systems participating in Document Understanding Conferences (DUC) in most of the cases, and proves quite insensitive to the noise in the data. Note that the one-line summarization system (Sharifi et al., 2010), which requires a given topic and focuses on the selection of key phrases most related to the topic, works on a setting different from ours.

In general, LexRank is a graph-based method for computing relative importance of textual units. Erkan and Radev (2004) use it to compute the sentence importance based on the concept of eigenvector centrality in a graph representation of sentences. They use a connectivity matrix based on intra-sentence cosine similarity as the adjacency matrix of the graph representation of sentences. We adapt LexRank to compute the importance of tweets.

Algorithm 1 shows the framework of the baseline. The input tweets are denoted by ts and the outputted summarization are denoted by ret.

We first call the function $repr$ to represent each tweet t into bag-of-words vector $\vec{t}$, with the usual TFIDF weighting schema as defined in Formula 2, where tf is the occurrence times of the term in the tweet, N is the total number of the tweets for summarization, df is the number of tweets that contain that term. To extract words from tweets, some preprocessing is conducted. Firstly, stop words are removed. Stop words are mainly from a set of frequently-used words[7]. We also extract the top 200 most frequent words from about

[6]For example, to summarize Chinese tweets, the main effort involves an additional pre-processing: to run Chinese work breaker to get words.

[7]http://www.textfixer.com/resources/common-english-words.txt

Algorithm 1 Framework of the baseline.

Require: A collection of tweets: ts.

1: Initialize the set of tweet vectors ts_v:$ts_v = \emptyset$.
2: **for all** tweet $t \in ts$ **do**
3: Get a feature vector $\vec{t}$:$\vec{t} = repr(t)$.
4: Add $\vec{t}$ to ts_v:$ts_v = ts_v \cup \{\vec{t}\}$.
5: **end for**
6: Construct the graph:$gr = graph(ts_v)$.
7: Compute salience scores:$sc = scores(gr)$.
8: Select tweets :$ret = select(sc, ts)$.
9: **return** ret.

10 million tweets, from which another 54 words are manually selected as stop words. Secondly, tweet metadata (e.g., links, account names and hash tags) is extracted using regular expressions and then normalized. Links and account names are replaced by LINK and ACCOUNT, respectively, while hash tags are treated as common keywords. Finally, a simple dictionary-lookup based normalization procedure is conducted, using a pre-compiled list including incorrect/correct word pairs, e.g., "loooove"/"love", to correct common ill-formed words.

$$\text{TFIDF} = tf \times ln(\frac{N}{df}) \tag{2}$$

Then we call the function $graph$ to construct a graph, denoted by $G =< V, E >$, where V stands for the set of vertexes and E represents the set of edges. Firstly, a vertex is introduced for each tweet. Secondly, for each tweet pair, if their similarity is non-zero, an unidirectional edge connecting the corresponding vertices is added. The edge weight is defined in Formula 3, where i denotes the i^{th} vertex, corresponding to tweet $\vec{t}_i$. We enforce $\text{sim}(\vec{t}, \vec{t}) = 0$ to avoid self-transition. Following Formula 3, the transition probability from the i^{th} vertex to the j^{th} vertex can be defined by Formula 5. It is worth noting that $p(i, j)$ is usually not equal to $p(j, i)$ because of the different normalization factor in the denominator.

$$w(i, j) = \text{sim}(\vec{t}_i, \vec{t}_j) \tag{3}$$

$$\text{sim}(\vec{t}_i, \vec{t}_j) = \frac{\vec{t}_i \cdot \vec{t}_j}{|\vec{t}_i| \cdot |\vec{t}_j|} \tag{4}$$

$$p(i, j) = \begin{cases} \dfrac{w(i, j)}{\sum_{j'} w(i, j')}, if \sum\limits_{j'} w(i, j') \neq 0 \\ 0, otherwise \end{cases} \tag{5}$$

Next we call the function $scores$ to compute the salience score for each tweet according to Formula 6, where s_i is the salience score of the i^{th} vertex, λ is the damping factor, and V is

the number of tweets for summarization.

$$s_i = \lambda \cdot \sum_{j \neq i} s_j \cdot p(j,i) + (1-\lambda) \cdot 1/|V| \tag{6}$$

We can consider the above process as a Markov chain, which takes tweets as states and the transition matrix T is defined in Formula 7, where E is the identity matrix. Since T is irreducible, it is guaranteed that the salience scores can be obtained by the principal eigenvector of the transition matrix T.

$$T = \lambda \cdot P + (1-\lambda)/|V| \cdot E \tag{7}$$

For implementation, the initial salience scores of all tweets are set to 1 and the iteration algorithm in Formula 6 is used to compute the new scores. Usually the convergence of the iteration algorithm is achieved when the difference between the scores computed at two successive iterations for any tweet goes below a given threshold δ.

Finally, we run the function $select$, an alternative implementation of the MMR algorithm, to select at most M tweets as the summarization. Whether a tweet is selected as a representative depends on two factors: its salience score and its similarity to the already selected tweets. Specifically, a tweet is chosen if it is the candidate with the greatest salience score and its similarity to any selected tweet is below a threshold ϵ [8]. No matter whether the most salient candidate is chosen or not, it will be removed from the candidate set. This selection process repeats until M tweets are chosen or the candidate set is empty. Details are illustrated in Algorithm 2.

Algorithm 2 Representative tweet selection.

Require: maximum number of tweets allowed: M.
Require: Scored tweets: $\{(\vec{t}, score)\}$.

Initialize the set of selected tweets sel:$sel = \emptyset$.
Initialize the set of candidates cd: $cd = \{(\vec{t}, score)\}$.
while $|sel| < M$ and $|cd| \neq 0$ **do**
    Select the most salient t^*:$t^* = \arg\max_{t \in cd} t.score$.
    Remove t^* from cd:$cd = cd - \{t^*\}$.
    if $\forall s \in sel, sim(s.\vec{t}, t^*.\vec{t}) < \epsilon$ **then**
        Select t^*:$sel = sel \cup \{t^*\}$.
    end if
end while
return sel.

5 Our System

The baseline has several limitations. Firstly, only terms are used to compute a tweet's salience score. Because a tweet is short and often noisy, the computed importance score is often not reliable. For example, consider the following two tweets:

[8] We treat δ and ϵ as systematic parameters, whose value are experimentally determined on the development dataset.

- is Obama planning to spend the 17 days in Hawaii? #vacation#
- Obama Christmas shopping for his family, already in Hawaii for vaca

They have similar meanings but low cosine similarity, largely caused by the “vaca" in the second tweet , which is actually an abnormal abbreviation of “vacation" in the first tweet. Secondly, tweets are selected only according to their salience scores, despite how hard they can be understood. It has been observed that in the summarization outputted by the baseline, a significant part of tweets are hard to read, which are short, or noisy, e.g., having many OOV words and grammatically incorrect. As an illustrative example, consider the following two tweets. The first one is short and not meaningful while the second one is informally written with low readability.

- Rodney Hood, folks, Rodney Hood....
- Rodney Hood cold . yeah he going pro in another year or so

Thirdly, because user diversity is ignored, too many tweets from the same Twitter account occur in the summarization.

Our system try to overcome these limitations by: 1) Incorporating social network features, i.e., re-tweeted times and follower numbers to make the salience score more reliable; 2) introducing the readability feature to make the outputted summarization more readable; and 3) considering user diversity, i.e., dropping a tweet if the number of selected tweets from the same Twitter account goes above a threshold.

Accordingly, we make three updates on the framework illustrated in Algorithm 1. First the $graph$ function is modified so that the weight between two tweets is computed according to Formula 8, where $a(j)$ is defined in Formula 9. $a(j)$ incorporates two kinds of evidences: 1) Social network features, which says a tweet is more important if it has been re-tweeted more times (retw(j)), or it is published by an account with more followers (foll(j)); and 2) Readability (readability(j)), which says more readable tweets are more favorable. With the updated graph, the same scoring function $scores$ is called to assign a salience score for each tweet.

$$w(i,j) = \frac{\text{sim}(\vec{t}_i, \vec{t}_j) \cdot a(j)}{\sum_{j'} \text{sim}(\vec{t}_i, \vec{t}_{j'}) \cdot a(j')} \tag{8}$$

$$a(j) = \text{retw}(j) \cdot \text{foll}(j) \cdot \text{readability}(j) \tag{9}$$

Secondly, we modify Algorithm 2 to add user diversity into the summarization. We calculate the number of different Twitter accounts of the input tweets, denoted by K, then define a threshold N according to Formula 10, where β is a systematic parameter with a positive real value. To decide if tweet t should be put into the outputted summarization, we check the number of the selected tweets from t's account. If that number is greater than N, we drop the tweet, otherwise we select it.

$$N = \left\lceil \beta \cdot \frac{M}{K} \right\rceil \tag{10}$$

We use Twitter APIs, i.e., http://twitter.com/#!/[account], to compute the number of followers for a given Twitter account. For example, filling “[account]" with “lxh5147" can

check out how many Twitter accounts are following “lxh5147". We estimate how many times a tweet is re-tweeted by analyzing the content of a large collection of reference tweets, which are crawled using Twitter APIs within the same day. For any two tweets, if the first one starts with “RT" followed by the content of the second one, we say the second tweet is re-tweeted by the first one.

Readability is the ease in which text can be read and understood. One widely adopted readability is measured according to the Flesch Formula 11, Where: ASL is the average sentence length (number of words divided by number of sentences) and ASW is the average word length in syllables (number of syllables divided by number of words).

$$206.835 - (1.015 \times ASL) - (84.6 \times ASW) \tag{11}$$

We enhance this formula by considering two additional factors: 1) The average number of OOV words, i.e., the number of OOV words divided by the total number of words, denoted by AOW; and 2) the average number of abnormal symbols, i.e., the number of abnormal symbols divided by the total number of words, denoted by AAS. We compile a dictionary of 1 million words[9], and a list of 125 symbols to identify OOV words and abnormal symbols, respectively.

The updated measurement is then defined in Formula 12, in which the coefficients (i.e., 10.5 and 9.8) are determined using linear regression. We further normalize the readability using Formula 13, where $readability^{(i)}$ is the readability of the i^{th} tweet computed using Formula 12. In Formula 13, we assume a normal distribution of tweet readability, whose mean and variance are defined in Formula 14 and 15, respectively, where n is the number of the input tweets.

$$206.835 - (1.015 \times ASL) - (84.6 \times ASW) - (10.5 \times AOW) - (9.8 \times AAS) \tag{12}$$

$$\text{readability}(i) = Pr(readability < readability^{(i)}) \tag{13}$$

$$\hat{\mu} = \frac{1}{n}\sum_{i=1}^{n} readability^{(i)} \tag{14}$$

$$\hat{\sigma}^2 = \frac{1}{n-1}\sum_{i=1}^{n}(readability^{(i)} - \hat{\mu})^2 \tag{15}$$

6 Evaluation

In this section, we evaluate our system on a manually annotated dataset. We also study the contribution of social signals, readability and user diversity, respectively.

[9]To get a copy of the dictionary, please contact the first author.

6.1 Data Preparation

Unlike such multi-document summarization tasks in Document Understanding Conference (DUC), no gold-standard dataset for tweet summarization is available. We manually annotate a dataset (DS) for automatic evaluation of summarization, as described below.

100 English trending topics from March 1^{st} to March 30^{th} 2010 are randomly selected. Some examples are "lady gaga", "Obama", "Denver", "james Cameron", and "ipad". For each trending topic, at most 1,000 English tweets are crawled using Twitter APIs. For each tweet, the information about its re-tweeted times and the number of followers of its account are computed. To facilitate future experiments, for each crawled tweet, stop words are removed and its metadata and keywords are extracted. The readability of each tweet is automatically calculated using Formula 13. Three annotators[10] are involved. For each topic, they independently select M tweets (computed using Formula 1) from the related tweets, thus forming the gold-standard dataset DS. 10 topics are randomly chosen for development and the remainder for testing, denoted by DS_T. The system parameters, i.e., λ, δ, ϵ and β are experimentally set to the optimal values (0.85, 0.0001, 0.2 and 10), which yield the best performance on the development dataset.

For any topic c and any pair of annotated results from two annotators, denoted by A_c and B_c, we compute the inter-agreement with Formula 16. The average inter-agreement (over all topics and all possible pairs) is 0.78.

$$\text{inter}-\text{agreement}(A_c, B_c) = \frac{|A_c \bigcap B_c|}{|A_c \bigcup B_c|} \tag{16}$$

6.2 Evaluation Metrics

We adopt the widely used ROUGE-N as performance metrics, which is an n-gram recall based statistic that can be computed as follows:

$$\text{ROUGE}-\text{N}(s) = \frac{\sum_{r \in R} \overrightarrow{\Phi_n(r)} \cdot \overrightarrow{\Phi_n(s)}}{\sum_{r \in R} \overrightarrow{\Phi_n(r)} \cdot \overrightarrow{\Phi_n(r)}} \tag{17}$$

Where: $R = \{r_1, r_2, \cdots, r_m\}$ is a set of reference summaries; s is a summary generated automatically by some system; $\overrightarrow{\Phi_n(d)}$ is a binary vector representing the n-grams contained in a document d; the i^{th} component $\Phi_n^i(d)$ is 1 if the i^{th} n-gram is contained in d and 0 otherwise.

6.3 Results

Table 1 reports the ROUGH-1 and ROUGE-2 of the baseline (BS) and our system (SS), with $\alpha = 0.05$. We observe a significant improvement of ROUGE-1 and ROUGE-2, showing the overall advantages of our system. We vary M, and find our method consistently outperforms the baseline, as showed in Table 2. As a case study, we list the outputs of our system and the baseline, respectively, in Table 3.

[10]They are native English speakers.

Social signals, readability and user diversity are added to the baseline, respectively, to study their contributions. The corresponded systems are denoted by BS_S, BS_R and BS_U, respectively. Table 4 shows the results, from which it can be seen that social signals are most helpful, followed by readability and then user diversity. We also combine any two of the three factors, add them to the baseline, and test the updated system. Results are listed in Table 5. The subscript letters S, R and U stand for social network features, readability and user diversity, respectively. We see the combination of S and R outperforms other settings.

System	ROUGE-1	ROUGE-2
BS	0.3591	0.2871
SS	0.4562	0.3692

Table 1: Performance of Different Systems. $\alpha = 0.05$.

System	5	10	15	25	30	35	40	45
BS	0.1134	0.1726	0.2108	0.2541	0.2983	0.3120	0.3312	0.3425
SS	0.1632	0.2153	0.2691	0.3125	0.3468	0.3670	0.3981	0.4315

Table 2: Comparison of ROUGE-1 with Varied M.

System	Outputted Summarization
BS	Apple said to be launching two new iPad models in 2012
	RETWEEET if youu Own A ipod , Ipad , Or iPhone :)
	Last chance to get your mits on an iPad 2 thanks to @SKECHERS_UK
SS	Apple reportedly to debut two new iPad models next year, more than double battery life
	10 year olds have a Blackberry, an iPad, a laptop, and a Facebook. When I was 10, I felt cool with my new markers.
	Two new iPad versions to unveil in January says sources

Table 3: Summarization of Our Method and The Baseline(For Topic “iPad").

7 Application to Twitter Search

Twitter search is an increasingly popular service of providing access to tweets. It returns a list of matched tweets for a given query, as illustrated in Figure 1. We observe two problems here. Firstly, there are often similar tweets in the search results. For example, the first and the second tweet in Figure 1 are almost the same. This kind of redundancy is annoying since in general users are more interested in “new" information when reading tweets. Secondly, a large number of tweets in search results, e.g., those about private conversations or those which are fragmented and hard to understand (e.g., the fifth tweet in Figure 1), are not meaningful for the general audience.

We apply our multi-tweet summarization system to the results of Twitter search, and build an end-to-end application, as illustrated in Figure 2. By default only representative tweets are displayed, and each of them has a “show similar results” link. Clicking the link will show at most 10 tweets most similar to the corresponded tweet.

We conduct user studies to evaluate whether our system is helpful to Twitter search. We first randomly sample 50 queries (DS_U) from the trending topics from March 1^{st} to March

System	ROUGE-1	ROUGE-2
BS	0.3591	0.2871
BS_S	0.4218	0.3251
BS_R	0.3945	0.3148
BS_U	0.3748	0.3016

Table 4: Contribution of Social Signals, Readability and User Diversity.

System	ROUGE-1	ROUGE-2
BS	0.3591	0.2871
BS_{SR}	0.4512	0.3587
BS_{SU}	0.4381	0.3471
BS_{RU}	0.4217	0.3294

Table 5: Contribution of Social Signals, Readability and User Diversity.

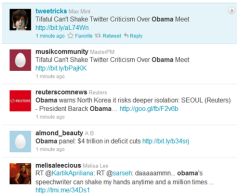


Figure 1: An example of Twitter search about “Obama"

Figure 2: An example of summarization of Twitter search results about “Obama".

30^{th} 2010. Then for each query in DS_U, results from Twitter search and our system are displayed side by side, and 3 users[11] are asked to choose which is better. For each user, we record how many queries our system is voted to be better. The inter-rater agreement[12] between users is also computed. Table 6 shows the results, suggesting that users tend to be more satisfied with our system. The inter-rater agreement is 0.74, indicating that users are more likely to agree with each other. Note that the values in the “Votes for ours" column are computed using Formula 18, where Q_u denotes the queries for which the user $u \in \{A, B, C\}$ believes our system gives better results than Twitter search, and Q denotes all queries in

[11]They are college students who did not receive any special training as preparation.

[12]Cohen's kappa coefficient is used as the inter-rater agreement.

DS_U.

$$\frac{|Q_u|}{|Q|} \times 100\% \tag{18}$$

User	Votes for ours (%)
A	72
B	57
C	58

Table 6: Comparison between our system and Twitter search. A, B and C denote three annotators, respectively. The inter-rater agreement is 0.74.

8 Conclusion and Future work

We study the task of multi-tweet summarization, which selects a given number of representative tweets so as to keep important information while dropping noise and redundancy. One main motivation of this task is to provide a tool to help people efficiently access a large number of tweets, which are short and prone to noise. This is important considering that tweets are one increasing popular repository of fresh information. We advocate that multi-tweet summarization is an important building block for other information extraction tasks on tweets, in the sense that it allows these tasks to focus on important tweets.

One main challenge is the lack of information to compute a tweet's salience score. We propose a graph-based system which leverages social network features, readability and user diversity to address this challenge. On a manually annotated gold-standard dataset, we show our system outperforms the state-of-the-art baseline. We apply our system to Twitter search and demonstrate that it improves user's satisfaction to Twitter search.

In our experiments, we have observed that users are often more interested in tweets with events or opinions. Therefore, exploiting events and opinions in tweets represents a promising direction to explore in future.

References

Barzilay, R., Elhadad, N., and McKeown, K. R. (2001). Sentence ordering in multidocument summarization. In *HLT*, pages 1–7.

Barzilay, R., McKeown, K. R., and Elhadad, M. (1999). Information fusion in the context of multi-document summarization. In *ACL*, pages 550–557.

Duan, Y., Jiang, L., Qin, T., Zhou, M., and Shum, H.-Y. (2010). An empirical study on learning to rank of tweets. In *Coling*, pages 295–303.

Efron, M. (2011). Information search and retrieval in microblogs. *Journal of the American Society for Information Science and Technology*, 62(6):996–1008.

Erkan, G. and Radev, D. R. (2004). LexRank: Graph-based Lexical Centrality as Salience in Text Summarization. *Journal of Artificial Intelligence Research*, 22.

Finin, T., Murnane, W., Karandikar, A., Keller, N., Martineau, J., and Dredze, M. (2010). Annotating named entities in twitter data with crowdsourcing. In *CSLDAMT*, pages 80–88.

Golbeck, J., Grimes, J. M., and Rogers, A. (2010). Twitter use by the u.s. congress. *Journal of the American Society for Information Science and Technology*, 61(8):1612–1621.

Goldstein, J., Kantrowitz, M., Mittal, V., and Carbonell, J. (1999). Summarizing text documents: Sentence selection and evaluation metrics. In *In Research and Development in Information Retrieval*, pages 121–128.

Hardy, H., Shimizu, N., Strzalkowski, T., Ting, L., Zhang, X., and Wise, G. B. (2002). Cross-document summarization by concept classification. In *SIGIR*, pages 121–128.

Heverin, T. and Zach, L. (2010). Twitter for city police department information sharing. *Proceedings of the American Society for Information Science and Technology*, 47(1):1–7.

Inouye, D. and Kalita, J. K. (2011). Comparing twitter summarization algorithms for multiple post summaries. In *SocialCom/PASSAT*, pages 298–306.

Jansen, B. J., Zhang, M., Sobel, K., and Chowdury, A. (2009). Twitter power: Tweets as electronic word of mouth. *Journal of the American Society for Information Science and Technology*, 60(11):2169–2188.

Knight, K. and Marcu, D. (2002). Summarization beyond sentence extraction: a probabilistic approach to sentence compression. *Artif. Intell.*, pages 91–107.

Kwak, H., Lee, C., Park, H., and Moon, S. (2010). What is twitter, a social network or a news media? In *WWW*, pages 591–600.

Liu, X., Li, K., Han, B., Zhou, M., Jiang, L., Xiong, Z., and Huang, C. (2010). Semantic role labeling for news tweets. In *Coling*, pages 698–706.

Liu, X., Zhang, S., Wei, F., and Zhou, M. (2011). Recognizing named entities in tweets. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1*, HLT '11, pages 359–367, Stroudsburg, PA, USA. Association for Computational Linguistics.

Mani, I. and Bloedorn, E. (1999). Summarizing similarities and differences among related documents. *Inf. Retr.*, pages 35–67.

Neto, J., Freitas, A., and Kaestner, C. (2002). Automatic text summarization using a machine learning approach. In Bittencourt, G. and Ramalho, G., editors, *Advances in Artificial Intelligence*, volume 2507 of *Lecture Notes in Computer Science*, pages 205–215. Springer Berlin / Heidelberg.

O'Connor, B., Krieger, M., and Ahn, D. (2010). Tweetmotif: Exploratory search and topic summarization for twitter. In *ICWSM*.

Page, L., Brin, S., Motwani, R., and Winograd, T. (1998). The PageRank Citation Ranking: Bringing Order to the Web. Technical report, Stanford Digital Library Technologies Project.

Priem, J. and Costello, K. L. (2010). How and why scholars cite on twitter. *Proceedings of the American Society for Information Science and Technology*, 47(1):1–4.

Qazvinian, V. and Radev, D. R. (2008). Scientific paper summarization using citation summary networks. In *Proceedings of the 22nd International Conference on Computational Linguistics - Volume 1*, COLING '08, pages 689–696, Stroudsburg, PA, USA. Association for Computational Linguistics.

Qazvinian, V. and Radev, D. R. (2010). Identifying non-explicit citing sentences for citation-based summarization. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, ACL '10, pages 555–564, Stroudsburg, PA, USA. Association for Computational Linguistics.

Radev, D. (2004). Centroid-based summarization of multiple documents. *Information Processing & Management*, pages 919–938.

Saggion, H. (2011). Learning predicate insertion rules for document abstracting. In *CICLing (2)*, pages 301–312.

Sankaranarayanan, J., Samet, H., Teitler, B. E., Lieberman, M. D., and Sperling, J. (2009). Twitterstand: news in tweets. In *GIS*, pages 42–51.

Sharifi, B., Hutton, M.-A., and Kalita, J. (2010). Summarizing microblogs automatically. In *HLT: NAACL*, HLT '10, pages 685–688.

Wan, X. and Yang, J. (2008). Multi-document summarization using cluster-based link analysis. In *SIGIR*, pages 299–306.

Weng, J., Lim, E. P., Jiang, J., and He, Q. (2010). TwitterRank: finding topic-sensitive influential twitterers. In *WSDM*, pages 261–270, New York, NY, USA.

Xie, Z., Eugenio, B. D., and Nelson, P. C. (2008). From extracting to abstracting: Generating quasi-abstractive summaries. In *LREC*.

Topical Word Trigger Model for Keyphrase Extraction

Zhiyuan Liu Chen Liang Maosong Sun
Department of Computer Science and Technology
State Key Lab on Intelligent Technology and Systems
National Lab for Information Science and Technology
Tsinghua University, Beijing 100084, China
{lzy.thu, chenliang.harry}@gmail.com, sms@tsinghua.edu.cn

ABSTRACT
Keyphrase extraction aims to find representative phrases for a document. Keyphrases are expected to cover main themes of a document. Meanwhile, keyphrases do not necessarily occur frequently in the document, which is known as the *vocabulary gap* between the words in a document and its keyphrases. In this paper, we propose Topical Word Trigger Model (TWTM) for keyphrase extraction. TWTM assumes the content and keyphrases of a document are talking about the same themes but written in different languages. Under the assumption, keyphrase extraction is modeled as a translation process from document content to keyphrases. Moreover, in order to better cover document themes, TWTM sets trigger probabilities to be topic-specific, and hence the trigger process can be influenced by the document themes. On one hand, TWTM uses latent topics to model document themes and takes the coverage of document themes into consideration; on the other hand, TWTM uses topic-specific word trigger to bridge the vocabulary gap between the words in document and keyphrases. Experiment results on real world dataset reveal that TWTM outperforms existing state-of-the-art methods under various evaluation metrics.

TITLE AND ABSTRACT IN CHINESE

采用基于主题的词触发模型进行关键词抽取

关键词抽取旨在发现文档中有代表性的词或者短语。抽取的关键词应当覆盖文档的主要话题。同时，关键词并不一定在文档中频繁出现，这就是所谓的文档与关键词间的“词汇鸿沟”问题。本文提出一种基于主题的词触发模型（TWTM）进行关键词抽取。该模型假设文档内容和关键词是在用不同的语言描述相同的话题。在这个假设下，关键词抽取就可以被建模为从文档到关键词的翻译过程。为了更好地覆盖文档话题，该模型的触发概率都是主题相关的，从而使触发过程受到文档主题的影响。一方面，该模型利用隐含主题对文档话题进行建模，从而将文档主题的覆盖度考虑在内；另一方面，该模型采用主题相关的词触发建立起了文档与关键词的桥梁。在真实数据上的实验结果表明，该模型优于已有关键词抽取方法。

KEYWORDS: keyphrase extraction, latent topic model, word trigger model.

KEYWORDS IN CHINESE: 关键词抽取, 隐含主题模型, 词触发模型.

Proceedings of COLING 2012: Technical Papers, pages 1715–1730,
COLING 2012, Mumbai, December 2012.

1 Introduction

For information retrieval and management, people usually annotate a collection of keyphrases to a document as its brief summary. Keyphrases can be found in most digital libraries and information retrieval systems (Turney, 2000; Nguyen and Kan, 2007). Web information, most of which is in the form of text, is growing at a rapid rate. For the large volume of documents, it will be inefficient for human editors to manually index keyphrases. Therefore, automatically extracting keyphrases for documents is proposed as a challenging task in natural language processing. The task is also referred to as *keyphrase extraction*.

When we enter Web 2.0 era, social tagging is invented to help users manage and share information. The social tags can be regarded as a type of keyphrases. Various methods have been proposed for automatic social tag suggestion, which can be regarded as a special type of keyphrase extraction. In social tag suggestion, given a document, the system will select keyphrases from a controlled tag list instead of document itself. It indicates that keyphrases do not necessarily occur in the given document. It is obvious that it provides a more flexible and convenient scheme compared to traditional keyphrase extraction, and thus becomes the main application of keyphrase extraction. In this paper, we will focus on the new setting of keyphrase extraction, which is named as *keyphrase extraction from a controlled vocabulary*. In the following paper, unless specifically noted, we use keyphrase extraction as referred to the new setting.

As a summary of document, keyphrases are expected to represent and cover the main themes of the given document. Suppose there is an article talking about the "Apple" company and its smartphone "iPhone". The extracted keyphrases are expected to cover the both themes, i.e., "Apple" and "iPhone". This indicates that a set of keyphrases that focuses on only one theme will not be adequate. Meanwhile, representative keyphrases do not necessarily occur frequently in the document. Take the article for example again, it may mention "iPhone" (smartphone of Apple), "iPad" (tablet computer of Apple) and "Steve Jobs" (founder of Apple) for many times, but refer to "Apple" not so frequently. Nevertheless, it is intuitive that "Apple" should be a representative keyphrase of the document. We refer the phenomenon as a *vocabulary gap* between words in document and keyphrases. In summary, given a document, keyphrase extraction should: (1) find a set of representative keyphrases that can better cover the main themes of the document. (2) The selection of keyphrases should primarily rely on their semantic relatedness with the document rather than being constrained by their occurrence frequencies in the document. This requires keyphrase extraction can bridge the vocabulary gap between document content and keyphrases.

Many unsupervised methods have also been extensively explored for keyphrase extraction. The most simple unsupervised method is ranking the candidate keyphrases according to TFIDF (Salton and Buckley, 1988) and then selecting top-ranked ones as keyphrases. There are also graph-based methods (Mihalcea and Tarau, 2004; Wan and Xiao, 2008b,a; Liu et al., 2010), clustering-based methods (Grineva et al., 2009; Liu et al., 2009) and latent topic models (Heinrich, 2005; Blei and Lafferty, 2009) proposed for keyphrase extraction. Most of these methods take frequencies of candidate keyphrases as the crucial decision criteria, and thus tend to select those high-frequency ones as keyphrases. Given sufficient annotation data for training, we can adopt the classification-based approach for keyphrase extraction. For example, some methods (Frank et al., 1999; Witten et al., 1999; Turney, 2000) regard keyphrase extraction as a binary classification problem (is-keyphrase

or non-keyphrase). Keyphrase extraction can also be considered as a multi-label classification problem (Tsoumakas and Katakis, 2007), in which each keyphrase is regarded as a category label. Various methods such as Naive Bayes (Garg and Weber, 2008) and *k*NN (Li et al., 2009) have been explored. Some researchers proposed using latent topics to build semantic relations between words and tags. The representative methods include TagLDA (Krestel et al., 2009; Si and Sun, 2009) and Content Relevance Model (CRM) (Iwata et al., 2009). However, these methods usually suffer from the over-generalization problem.

Recently, a new approach based on word alignment models (WAM) in statistical machine translation (SMT) has been proposed for keyphrase extraction (Ravi et al., 2010; Liu et al., 2011b,a, 2012). WAM-based methods assume the content and keyphrases of a document are describing the same themes but written in different languages. Under this assumption, WAM-based methods regard keyphrase extraction as a translation process from document content to keyphrases. This process is modeled as a trigger from important words in document content to keyphrases according to trigger probabilities between words and keyphrases. WAM-based methods will learn trigger probabilities from sufficient document-keyphrase pairs. With the trigger probabilities, given a novel document, WAM-based methods are able to extract relevant keyphrases that do not necessarily occur so frequently in the document.

Although achieving significant improvement in bridging the vocabulary gap between document and keyphrases, WAM-based methods, however, cannot well guarantee the coverage of document themes. The crucial reason is analyzed as follows. WAM-based methods, formalizing trigger probabilities at *word level*, consider each single word in document and project from document content to keyphrases. However, the coverage of document themes should be appreciated at *topic level*, which is beyond the power of WAM-based methods.

A promising approach for representing document themes is latent topic models (Blei et al., 2003). In topic models, both words/keyphrases and documents are represented as probabilistic distributions over latent topics. Topic models are widely adopted as a guaranteed approach to represent document themes. Topic models themselves can also be used for keyphrase extraction, referred to as topic-based methods, by simply ranking keyphrases according to their semantic relevance with the document themes in terms of latent topics. Topic-based methods can be regarded as a trigger process at topic level, contrast to the trigger process at word level in WAM-based methods. Since common keyphrases receive larger probabilities given a topic, topic-based methods tend to select those keyphrases that are too general to tightly capture the main themes of the document. For example, it may select "IT" as keyphrase for the above mentioned document, which is so general that cannot reflect the document themes well.

Is there a way to leverage the power of both word-level projection and topic-level coverage in keyphrase extraction? Can the two techniques, i.e., word alignment models and latent topic models, be integrated together to complement each other for keyphrase extraction? To address the problems, we propose Topical Word Trigger Model (TWTM) for keyphrase extraction. TWTM inherits the advantage of WAM-based methods, and also incorporates latent topic models so as to promise the coverage of document themes.

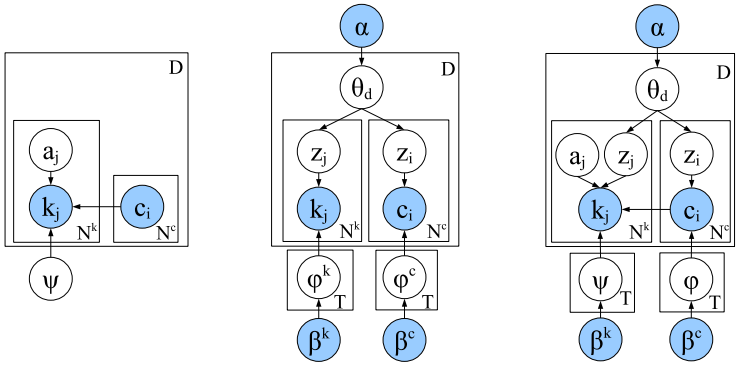
To compare and analyze the characteristics of different approaches for keyphrase extraction, we also introduce the method based on word alignment models, i.e., Word Trigger Model (WTM), and the method based on polylingual topic models, i.e., Topic Trigger Model (TTM).

As these names suggest, WTM identifies keyphrases by triggering at word level; while TTM triggering at topic level. TWTM, integrating their advantages, performs both word-level and topic-level triggers to extract keyphrases.

To demonstrate the effectiveness of our method, we carry out experiments on a real-world dataset crawled from a website with keyphrases having been annotated collaboratively by users. Experiment results show that TWTM can identify appropriate keyphrases with better coverage of document themes compared to existing WAM-based methods.

2 Our Method

In this section we first introduce two simple trigger methods, WTM and TTM, in which WTM performs triggering at word level while TTM at topic level. Afterwards, we introduce our method TWTM.



(a) WTM (b) TTM (c) TWTM

Figure 1: Trigger Models for keyphrase extraction.

2.1 Notations and Definitions

Before introducing baselines and our method, we give some notations. We denote a document as $d \in D$, where D is the document set. For each document d, we denote its content as a sequence of words $\mathbf{c} = \{c_i\}_{i=1}^{N^c}$ and its keyphrases as a set $\mathbf{k} = \{k_i\}_{i=1}^{N^k}$. The vocabulary of words in documents is denoted as W, and the vocabulary of keyphrases as V. Each word c_i in documents is an instance of a word type w in W, i.e., $c_i = w \in W$; each keyphrase k_i of documents is an instance of a keyphrase type v in V, i.e., $k_i = v \in V$.

We define keyphrase extraction as follows. Given a document d with its content $\mathbf{c}$, keyphrase extraction aims to seek a set of keyphrases $\mathbf{k}$ that maximizes $\Pr(\mathbf{k}|\mathbf{c})$. By simply assuming the keyphrases are independent conditional over d, we have $\Pr(\mathbf{k}|\mathbf{c}) = \prod_{k\in\mathbf{k}} \Pr(k|\mathbf{c})$. The optimal set of keyphrases $\mathbf{k}^*$ can be represented as follows:

$$\mathbf{k}^* = \arg\max_{\mathbf{k}} \Pr(\mathbf{k}|\mathbf{c}) = \arg\max_{\mathbf{k}} \prod_{k\in\mathbf{k}} \Pr(k|\mathbf{c}). \quad (1)$$

Suppose the number of keyphrases is pre-defined as N^k, we can simply find $\mathbf{k}^*$ by ranking

each candidate keyphrase $v \in V$ according to its score $\Pr(v|\mathbf{c})$ in descending order and selecting top-N^k keyphrases.

2.2 Word Trigger Model

Word Trigger Model (WTM) is inspired by IBM Model-1 (Brown et al., 1993), the most widely used word alignment models in SMT. WTM assumes the content and the keyphrases of a document are describing the same themes while written in two different languages: document content in one language while keyphrases in the other. From this perspective, keyphrase extraction can be regarded as a translation process from a given document content to keyphrases.

In more detail, the translation process is modeled as a trigger process as follows. First, WTM finds several important words in the document content as *trigger words*. Then, activated by these trigger words, WTM maps the document content into keyphrases. A *trigger* in the translation process can be regarded as a mapping function from the words in document to keyphrases.

WTM formalizes a trigger as a hidden variable $\mathbf{a}$. By assuming each keyphrase of a document is triggered by only one word in the content, a $\mathbf{a}$ maps each keyphrase at position j (i.e., k_j) as triggered by a word at position i in document content c (i.e., c_i), denoted as $a_j = i$. Given a document, its content $\mathbf{c}$ and keyphrases $\mathbf{k}$ can be connected by a trigger variable $\mathbf{a}$. The probability of triggering $\mathbf{k}$ from $\mathbf{c}$ can be formalized as

$$\Pr(\mathbf{k}|\mathbf{c}) = \sum_{\mathbf{a}} \Pr(\mathbf{k}, \mathbf{a}|\mathbf{c}) = \sum_{\mathbf{a}} \Pr(\mathbf{k}|\mathbf{a}, \mathbf{c}) \Pr(\mathbf{a}|\mathbf{c}). \tag{2}$$

Following the same assumptions of IBM Model-1 (Brown et al., 1993), for each document $d \in D$, WTM assumes the content $\mathbf{c}$ of d already exists, and the keyphrases $\mathbf{k}$ are generated from $\mathbf{c}$ as follows:

1. For each document d with content $\mathbf{c}$ and N^k keyphrases:
   (a) For j from 1 to N^k:
      i. Sample each word trigger link a_j from $1, \ldots, N^c$ according to a uniform distribution.
      ii. Sample each $k_j = v$ according to trigger probability $\Pr(k_j = v|c_{a_j} = w, \psi)$.

Here we denote the trigger probability from a word $w \in W$ to a keyphrase $v \in V$ as $\psi_{vw} = \Pr(v|w)$, and the trigger probabilities from W to V form a matrix ψ. The corresponding graphical representation is shown in Figure 1a. The boxes are "plates" representing replicates. In this graphical model, the variables $\mathbf{k}$ and $\mathbf{c}$ are shaded indicating that they are observed; while the unshaded variables, including $\mathbf{a}$ and ψ, are latent (i.e., unobserved). Under the same assumptions of IBM Model-1, we write

$$\Pr(\mathbf{k}|\mathbf{c}) \propto \frac{1}{(N^c)^{(N^k)}} \prod_{j=1}^{N^k} \sum_{i=1}^{N^c} \Pr(k_j = v|c_i = w) = \frac{1}{(N^c)^{(N^k)}} \prod_{j=1}^{N^k} \sum_{i=1}^{N^c} \psi_{vw}. \tag{3}$$

We see that, in WTM, trigger probabilities $\psi_{vw} = \Pr(k_j = v|c_i = w)$ are the key parameters for learning. WTM has global optimum, and is efficient and easily scalable to large training

data. We use Expectation-Maximization (EM) (Dempster et al., 1977) to estimate trigger probabilities ψ.

Using the estimated ψ, when given a novel document d with its content $\mathbf{c}$, we can rank each candidate keyphrase v as follows:

$$\Pr(v|\mathbf{c}) = \sum_{w \in \mathbf{c}} \Pr(v|w, \psi) \Pr(w|\mathbf{c}) = \sum_{w \in \mathbf{c}} \psi_{vw} \Pr(w|\mathbf{c}), \tag{4}$$

where $\Pr(w|\mathbf{c})$ indicates the weight of the word w in $\mathbf{c}$, which can be calculated using the TFIDF score of w in $\mathbf{c}$. From the ranking list in descending order, we can select the top-ranked ones as keyphrases of the given document.

2.3 Topic Trigger Model

WTM triggers keyphrases at the word level. We can also trigger at the topic level, and thus propose Topic Trigger Model (TTM) for keyphrase extraction. TTM is inspired by Polylingual Topic Models (Mimno et al., 2009), which is originally proposed to model parallel documents in multiple languages. TTM is an extension of latent Dirichlet allocation (LDA) (Blei et al., 2003).

Suppose there are T latent topics in TTM, and the number of topics $|T|$ can be pre-defined by users. TTM assumes that the content $\mathbf{c}$ and keyphrases $\mathbf{k}$ of a document d share the same distribution over $|T|$ topics (i.e., θ_d), which is drawn from a symmetric Dirichlet prior with concentration parameter α. TTM also assumes that each topic $t \in T$ corresponds to two different multinomial distributions over words, one for keyphrases (i.e., ϕ_t^k) and another for content (i.e., ϕ_t^c), each of which is drawn from a specific symmetric Dirichlet with concentration parameter, β^k or β^c. TTM can be viewed as a generative process of both document content and keyphrases as follows:

1. Sample word distribution ϕ_t^c from $Dirichlet(\beta^c)$ and sample keyphrase distribution ϕ_t^k from $Dirichlet(\beta^k)$ for each topic $t \in T$.
2. For each document $d \in D$ with N^c words and N^k keyphrases:
   (a) Sample topic distribution θ_d from $Dirichlet(\alpha)$.
   (b) For i from 1 to N^c
      i. Sample a topic $z_i = t$ from $Multinomial(\theta_d)$.
      ii. Sample a word $c_i = w$ according to multinomial distribution $\Pr(c_i = w|z_i = t, \phi^c)$.
   (c) For j from 1 to N^k
      i. Sample a topic $z_j = t$ from $Multinomial(\theta_d)$.
      ii. Sample a keyphrase $k_j = v$ according to multinomial distribution $\Pr(k_j = v|z_j = t, \phi^k)$.

The corresponding graphical model is shown in Figure 1b, where the observed variables (i.e., words $\mathbf{c}$, keyphrases $\mathbf{k}$ and hyper-parameters β^k and β^c) are shaded.

Given the observed words in a collection of documents, the task of TTM learning is to compute the posterior distribution of the latent topic assignments $\mathbf{z}$, the topic mixtures θ_d of

each document d, and the distributions over words ϕ_t^c and ϕ_t^k of each topic t. By assuming a Dirichlet prior β on ϕ, ϕ can be integrated according to the Dirichlet-multinomial conjugacy. In this paper, we use Gibbs Sampling to estimate parameters, which has been widely used as an inference method for many latent topic models. In Gibbs sampling, it is usual to integrate out the mixtures θ and topics ϕ and just sample the latent variables $\mathbf{z}$. The process is thus called *collapsed*.

Gibbs Sampling iteratively performs latent topic assignments for each word in the document set, and estimates the distributions over words of each topic (i.e., $\phi_{wt}^c = \Pr(c_i = w|z_i = t)$ and $\phi_{vt}^k = \Pr(k_j = v|z_j = t)$), and the distribution over topics of each document (i.e., $\theta_{td} = \Pr(z_i = t|d)$). Take a word token $c_i = w$ in d for example, given the current state of all but the variable z_i, the conditional probability of $z_i = t$ is

$$\Pr(z_i = t|c_i = w, \mathbf{c}^{\neg i}, \mathbf{z}^{\neg i}, \mathbf{k}) \propto \frac{N_{wt}^{c,\neg i} + \beta^c}{N_t^c - 1 + |W|\beta^c} \times \frac{N_{td}^{\neg i} + \alpha}{N_d^c + N_d^k - 1 + |T|\alpha}, \tag{5}$$

where z is the current topic assignments for all tokens in the document set; N_{wt}^c is the number of occurrences of word w that are assigned with topic t; N_t^c is the number of occurrences of all words that are assigned with topic t; N_{td} is the number of occurrences of topic t assigned in the current document d; N_d^c and N_d^k are the numbers of all tokens in the content and keyphrases of d, respectively; $\neg i$ indicates taking no account of the current position i.

According to the posterior probability $\Pr(z_i = t|c_i = w, \mathbf{c}^{\neg i}, \mathbf{z}^{\neg i}, \mathbf{k})$, we re-sample the topic assignment z_i of the c_i in d. Whenever z_i of c_i is assigned with a new topic drawn from Equation (5), N_{wt}^c and N_{td} are updated. We perform topic assignments in the same way for each word k_i in k of d. After enough sampling iterations to burn in the Markov chain, ϕ^c, ϕ^k and θ are estimated as follows:

$$\phi_{wt}^c = \frac{N_{wt}^c + \beta^c}{N_t^c + |W|\beta^c}, \quad \phi_{vt}^k = \frac{N_{vt}^k + \beta^k}{N_t^k + |V|\beta^k}, \quad \theta_{td} = \frac{N_{td} + \alpha}{N_d^c + N_d^k + |T|\alpha}. \tag{6}$$

When finishing the learning process, we obtain the distributions over words of each topic, i.e., ϕ^k and ϕ^c. Suppose we are asked to extract keyphrases from a novel document with only content $\mathbf{c}$. First, we infer topic assignments for each word in $\mathbf{c}$ with Gibbs Sampling. With the topic assignments, we summarize the distribution over topics of the content $\mathbf{c}$ as

$$\Pr(t|\mathbf{c}) = \frac{N_{td}^c + \alpha}{N_d^c + |T|\alpha}. \tag{7}$$

Triggered by the topics of $\mathbf{c}$, we rank each candidate keyphrase $v \in V$ as follows:

$$\Pr(v|\mathbf{c}) = \sum_{t \in T} \Pr(v|t, \phi^k) \Pr(t|c) = \sum_{t \in T} \phi_{vt}^k \theta_{td}, \tag{8}$$

and then select the top-ranked as keyphrases of the given document.

2.4 Topical Word Trigger Model

WTM and TTM perform trigger operations at either word or topic level. WTM addresses the problem of vocabulary gap between documents and keyphrases, and can thus suggest

keyphrases that are uncommon or even not showing up in the given document. TTM, on the other hand, takes the main themes of the given document in consideration when extracting keyphrases. In order to aggregate the advantages of the both methods, extended from WTM and TTM, we propose Topical Word Trigger Model (TWTM) for keyphrase extraction.

Similar to TTM, TWTM also assumes that topics are sampled at the word level. Each document is represented as a multinomial distribution over T latent topics. On the document content side, each topic $t \in T$ corresponds to a multinomial distribution over words, which is similar to TTM. On the keyphrase side, each topic $t \in T$ corresponds to a topic-specific translation table ψ_t. Given each document $d \in D$, the generative process of both document content and keyphrases is as follows:

1. Sample word distribution ϕ_t^c from $Dirichlet(\beta^c)$ for each topic $t \in T$.
2. Sample keyphrase distribution ψ_w^t from $Dirichlet(\beta^k)$ for each topic $t \in T$ and each word $w \in W$.
3. For each document $d \in D$ with N^c words and N^k keyphrases:
   (a) Sample topic distribution θ_d from $Dirichlet(\alpha)$.
   (b) For i from 1 to N^c
      i. Sample a topic $z_i = t$ from $Multinomial(\theta_d)$.
      ii. Sample a word $c_i = w$ according to multinomial distribution $\Pr(c_i = w | z_i = t, \phi^c)$.
   (c) For j from 1 to N^k
      i. Sample a topic $z_j = t$ from $Multinomial(\theta_d)$.
      ii. Sample each word trigger link a_j from all words in d that are generated from t according to a uniform distribution.
      iii. Sample each $k_j = v$ according to trigger probability $\Pr(k_j = v | c_{a_j} = w, \psi^t)$.

The graphical model is shown in Figure 1c. Topic-dependent translation probabilities ψ_t are the key parameters. Each ψ_t maintains the translation probability from each word $c_i = w$ in contents to each keyphrase $k_j = v$ under topic t, i.e., $\psi_{vw}^t = \Pr(k_j = v | c_i = w, t)$.

Given the observed words and keyphrases in a collection of documents, the task of TWTM learning is to compute the posterior distribution of the latent topic assignments $\mathbf{z}$, the topic mixtures θ_d of each document d, the distribution over words ϕ_t of each topic t, and the trigger probabilities ψ^t of each topic t. We can also use Gibbs Sampling to estimate parameters in TWTM. On the document content side, we can perform topic assignments $\mathbf{z}$ for each word as in TTM using Equation (5). On the keyphrase side, the problem is more complicated. Suppose we will assign each keyphrase k_j with a topic z_j and a trigger a_j. Take a keyphrase $k_j = v$ for example, given the current state of all but the variable z_j and a_j, the conditional probability of $z_j = t, a_j = i$ is calculated as follows,

$$\Pr(z_j = t, a_j = i | k_j = v, \mathbf{k}^{\neg j}, \mathbf{z}^{\neg j}, \mathbf{a}^{\neg j}, c_i = w, \mathbf{c}) \propto \frac{N_{vw}^{t,\neg j} + \beta^k}{N_w^t - 1 + |V|\beta^k} \times \frac{N_{td}^{\neg j} + \alpha}{N_d - 1 + |T|\alpha}, \tag{9}$$

where z is the current topic assignments for all translation pairs in the document set; N_{vw}^t is the number of occurrences that w is translated to v given topic t; N_w^t is the number of occurrences of word w in all translation pairs given topic t; N_{td} is the number of occurrences of topic t assigned in the current document d; N_d is the number of all translation pairs in

the current document d; $\neg j$ also indicates taking no account of the current position j. Given the conditional probability of $z_j = t, a_j = i$, we formalize the marginal probability of $z_j = t$ as follows,

$$\Pr(z_j = t | k_j = v, k^{\neg j}, z^{\neg j}, a^{\neg j}, c) \propto \sum_{i=1}^{N^c} \frac{N_{vc_i}^{t,\neg j} + \beta^k}{N_{c_i}^t - 1 + |V|\beta} \times \frac{N_{td}^{\neg j} + \alpha}{N_d - 1 + |T|\alpha}. \tag{10}$$

After re-assigning the topic $z_j = t$ for the current keyphrase according to Equation (10), we can further compute the trigger probability as follows:

$$\Pr(a_j = i | z_j = t, k_j = v, k^{\neg j}, z^{\neg j}, a^{\neg j}, c) = \frac{N_{vc_i}^{t,\neg j} + \beta}{N_{c_i}^t - 1 + |V|\beta^k}. \tag{11}$$

According to Equation (11), we re-assign trigger word c_i for the current keyphrase k_j. After enough sampling iterations to burn in the Markov chain, ϕ^c, ψ^t and θ are estimated as follows:

$$\phi_{wt}^c = \frac{N_{wt}^c + \beta^c}{N_t^c + |W|\beta^c}, \quad \psi_{vw}^t = \frac{N_{vw}^t + \beta}{N_w^t + |V|\beta}, \quad \theta_{td} = \frac{N_{td} + \alpha}{N_d^k + N_d^c + |T|\alpha}. \tag{12}$$

The potential size of topical trigger probabilities ψ is $|V| \times |W| \times |T|$. The size is comparative larger than ψ in WTM, and thus faces more serious problem of data sparsity. To remedy the problem, we use interpolation smoothing technique for ψ of TWTM. In this paper, we employ smoothing using ψ of WTM as follows:

$$\Pr_{SMOOTH}(v|w,t) = \lambda \Pr_{TWTM}(v|w,t) + (1-\lambda) \Pr_{WTM}(v|w,t), \tag{13}$$

where $\Pr_{SMOOTH}(v|w,t)$ is the smoothed topical trigger probabilities, $\Pr_{TWTM}(v|w,t)$ is the original topical trigger probabilities of TWTM, $\Pr_{WTM}(v|w,t)$ is the trigger probabilities of WTM. λ is the smoothing factor ranging from 0.0 to 1.0. When $\lambda = 0.0$, $\Pr_{SMOOTH}^t(v|w)$ will be reduced to non-topic trigger probabilities; and when $\lambda = 1.0$, there will be no smoothing in $\Pr_{SMOOTH}(v|w,t)$.

In TWTM, we perform keyphrase extraction as follows. Suppose we need perform keyphrase extraction for a document d with its content $\mathbf{c}$. We perform Gibbs Sampling to iteratively estimate the topic distribution of d (i.e., θ_d) according to document content $\mathbf{c}$. Afterwards, we select N^k keyphrases using the ranking score of each keyphrase v:

$$\Pr(v|\mathbf{c}) = \sum_{w \in \mathbf{c}} \sum_{t \in T} \Pr(v|w,t) \Pr(w|\mathbf{c}) \Pr(t|\mathbf{c}) = \sum_{w \in \mathbf{c}} \sum_{t \in T} \psi_{vw}^t \theta_{td} \Pr(w|\mathbf{c}), \tag{14}$$

where $\Pr(w|\mathbf{c})$ is the weight of the word w in document content c, which can be estimated by the TFIDF score of w in $\mathbf{c}$; $\Pr(t|\theta_d)$ is the probability of the topic t given the document d.

3 Experiments and Analysis

3.1 Dataset and Experiment Setting

To evaluate the performance of TWTM for keyphrase extraction, we carry out experiments on a real world dataset, crawled from `douban.com`, the largest product review website in China. Each product contains a description which is considered as a document content, and also contains a set of keyphrases annotated by users collaboratively which are considered

as standard keyphrases. The dataset consists of annotations for three types of products, i.e., book, movie and music. The statistics of the dataset is shown in Table 1, where $|D|$, $|W|$, $|V|$, $\hat{N}^c$ and $\hat{N}^k$ are the number of documents, the vocabulary of contents, the vocabulary of keyphrases, the average number of words and keyphrases in each document, respectively. In Table 1, we use DOUBAN to represent the whole dataset and use BOOK, MOVIE and MUSIC to show the statistics for instances for different product types.

Data	$\|D\|$	$\|W\|$	$\|V\|$	$\hat{N}^c$	$\hat{N}^k$
DOUBAN	71,525	160,276	99,457	86.30	10.53
BOOK	26,807	81,846	41,199	83.13	8.95
MOVIE	18,933	86,339	37,034	86.04	16.03
MUSIC	25,785	106,523	31,228	89.77	8.13

Table 1: Statistical information of dataset.

To evaluate the performance of our method and compare with other methods, we randomly select 1,000 instances from each of three types of products to form the test set with 3,000 instances, and use the rest of the dataset as training set.

In our experiments we select three evaluation metrics. The first metric is precision, recall and F-measure represented as $p = c_{correct}/c_{extract}$, $r = c_{correct}/c_{standard}$ and $F = 2pr/(p+r)$, where $c_{correct}$ is the total number of keyphrases that are correctly suggested by a method, $c_{extract}$ is the total number of automatic extracted keyphrases, and $c_{standard}$ is the total number of human-labeled standard keyphrases.

In fact, ranking order of extracted keyphrases also indicates the performance of different methods. A method is regarded better than another one if it ranks correct keyphrases higher. However, precision/recall/F-measure does not take the order of extracted keyphrases into account. To address the problem, we select the following two additional metrics. One metric is *binary preference measure* (Bpref) (Buckley and Voorhees, 2004). Bpref can consider the order of extracted keyphrases for evaluation. For a document, if there are R correct keyphrases within M extracted keyphrases by a method, in which r is a correct keyphrase and n is an incorrect keyphrase. It is defined as $\text{Bpref} = \frac{1}{R}\sum_{r\in R}\left(1 - \frac{|n \text{ ranked higher than } r|}{M}\right)$.

The other metric is *mean reciprocal rank* (MRR) (Voorhees, 2000) which is usually used to evaluate how the first correct keyphrase for each document is ranked. For a document d, $rank_d$ is denoted as the rank of the first correct keyphrase with all extracted keyphrases, It is defined as $\text{MRR} = \frac{1}{|D|}\sum_{d\in D}\frac{1}{rank_d}$, where D is the document set for keyphrase extraction.

3.2 Case Studies

Before quantitative evaluation, we perform case studies by looking into the topics learned by TWTM. By setting $T = 100$ of TWTM, We select two topics, i.e., Topic-59 and Topic-92 for study. In first several rows of Table 2, we list the top-10 words and top-10 keyphrases given the two topics separately (i.e., ranked by $\Pr(w|t)$ and $\Pr(v|t)$). From the top words and keyphrases, we can conclude that Topic-59 is about "art design" and Topic-92 is about "computer programming".

What will the topics influence the trigger probabilities? We pick a word "graphics" for example. In the context of the topic "art design", the word "graphics" always correlates with "design", "color" and "art"; while in the context of the topic "computer programming", the

word "graphics" generally refers to "computer graphics" and thus correlates to "programming", "software" and "programming language". At the bottom of Table 2, we show the top-6 keyphrases triggered by the word "graphics" with respect to the two topics. The value in the bracket after each keyphrase v is the probability $\psi^t_{vw} = \Pr(v|w,t)$, which is the topical specific trigger probability from w (here is the word "graphics") to v under the topic t. From the top triggered keyphrases by "graphics" under two topics, we can see they are discriminative with each other, and have intense topic characteristics.

	Topic-59		Topic-92	
Top	Words	Keyphrases	Words	Keyphrases
1	design	design	program	computer
2	creativity	creativity	develop	program
3	designer	designing	application	software engineer
4	magazine	handcraft	object	C++
5	fashion	graphic design	technology	programming
6	game	fashion	design	program design
7	work	game	system	program develop
8	color	product design	function	Linux
9	vision	industrial design	software	computer science
10	advertise	magazine	method	Alan
graphics	design (0.482)		game programming (0.201)	
	color science (0.089)		programming language (0.107)	
	font design (0.084)		Web 2.0 (0.094)	
	product design (0.077)		C (0.078)	
	landscape design (0.050)		Linux (0.077)	
	art design (0.039)		Computer Graphics (0.049)	

Table 2: Examples of topics learned with TWTM.

After investigating the topics, we look into keyphrase extraction results given a product description. Here we select a Japanese classical literature *The Tale of Genji* for example, which was written by *Murasaki Shikibu* in the early years of the 11th century [1]. The book recounts the life and love stories of a son of the Japanese emperor. In Table 3, we show the top-10 keyphrases extracted by WTM, TTM and TWTM, in which we use (—) to highlight the inappropriate keyphrases.

Method	Extracted Keyphrases
WTM	The Tale of Genji, classic, Japan, foreign literature, Murasaki Shikibu, politics (-), love, political science (-), eason (-), political philosophy (-)
TTM	novel, Japan, foreign literature, history, love, sociology (-), culture, literature, Russia (-), female (-)
TWTM	novel, foreign literature, The Tale of Genji, history, Japan, classic, literature, love, Murasaki Shikibu, politics (-)

Table 3: Examples of extracted keyphrases for the book *The Tale of Genji*.

From Table 3 we observe that: (1) WTM can suggest keyphrases that are closely related to the book, such as "The Tale of Genji" and "Murasaki Shikibu". However, due to not considering document themes, WTM will extract irrelevant keyphrases such as "politics", "political

[1] http://en.wikipedia.org/wiki/The_Tale_of_Genji.

science", "eason" and "political philosophy". (2) TTM triggers keyphrases with the favor of latent topics, which are usually too general to commendably represent the document main themes. We can see TTM fail to extract specific keyphrases such as "The Tale of Genji" and "Murasaki Shikibu". What is worse, TTM over-generalizes the document themes and extract inappropriate keyphrases "sociology", "Russia" and "female". (3) Taking advantages of both WTM and TTM, TWTM can extract specific and representative keyphrases and at the same time guarantee the coverage of document themes. We can see that TWTM achieves a smart balance between word-level projections and topic-level coverage.

3.3 Parameter Influences

There are two crucial parameters in TWTM, the number of topics T and the smoothing factor λ. In Table 4 and Table 5, we demonstrate the performance of TWTM for keyphrase extraction when parameters change.

T	Precision	Recall	F-measure	Bpref	MRR
10	0.313	0.304	0.309	0.313	0.825
30	0.337	0.325	0.331	0.337	0.837
50	0.339	0.329	0.334	0.339	0.827
70	0.351	0.339	0.345	0.351	**0.840**
100	**0.354**	**0.343**	**0.349**	**0.354**	0.838

Table 4: The influence of topic number T of TWTM for keyphrase extraction when $N^k = 10$ and smoothing factor $\lambda = 0.4$.

λ	Precision	Recall	F-measure	Bpref	MRR
0.0	0.310	0.254	0.279	0.310	0.676
0.2	0.318	0.314	0.316	0.318	0.823
0.4	0.354	0.343	0.349	0.354	**0.838**
0.6	**0.364**	**0.349**	**0.357**	**0.364**	0.812
0.8	0.350	0.334	0.342	0.351	0.764
1.0	0.323	0.306	0.314	0.324	0.731

Table 5: The influence of smooth factor λ of TWTM for keyphrase extraction when $N^k = 10$ and the number of topics $T = 100$.

From Table 4, we can see that, as the number of topics T increases from $T = 10$ to $T = 100$, the performance roughly improves. This indicates that the granularity of topics will influence the keyphrase extraction performance. When $T = 70$ and $T = 100$, the performance achieves a relatively stable good status. Hence, when comparing TWTM with other methods, we set $T = 100$ for TWTM.

As shown in Table 5, when the smoothing factor is set with $\lambda = 0.4$ or $\lambda = 0.6$, TWTM achieves the relatively best performance. When either $\lambda = 0.0$ (i.e., WTM), or $\lambda = 1.0$ (i.e., non-smoothed TWTM), the performance is much poorer compared to smoothed TWTM. This reveals that it is necessary to address the sparsity problem of TWTM by smoothing with WTM. Therefore, when comparing TWTM with other methods, we set $\lambda = 0.4$ for TWTM.

3.4 Performance Comparison

Besides WTM and TTM, we also select three representative methods as baselines for comparison: the classification-based method Naive Bayes (NB) (Garg and Weber, 2008), the topic-based method CRM (Iwata et al., 2009) and the word-projection method TAM (Si et al., 2010). We set $T = 1,024$ for CRM which achieves its best performance.

In Figure 2 we show the precision-recall curves of NB, CRM, TAM, WTM, TTM and TWTM on the dataset. Each point of a precision-recall curve represents extracting different number of keyphrases ranging from $N^k = 1$ (bottom right, with higher precision and lower recall) to $N^k = 10$ (upper left, with higher recall but lower precision), respectively. The closer the curve to the upper right, the better the overall performance of the method.

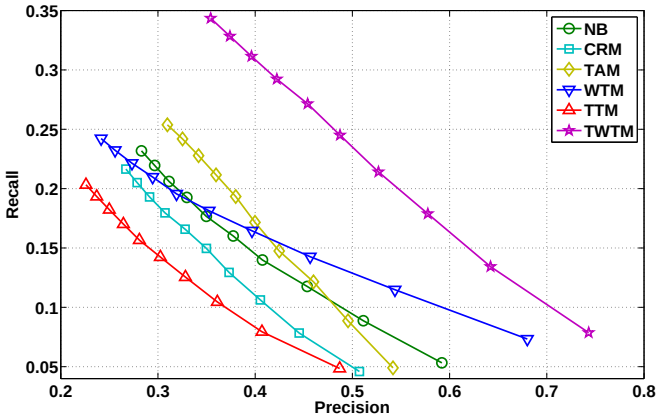


Figure 2: Precision-recall curves for keyphrase extraction.

Figure 2 clearly shows that TWTM outperforms other methods significantly. The interesting phenomena is that, when N^k is getting larger, the advantages of TWTM are more obvious compared to baselines. We know that when a system is asked to extract more keyphrases (i.e., N^k is larger), it is becoming important for extracted keyphrase to have a good coverage of document themes. Otherwise, the performance will drop sharply. This is the issue suffered by WTM. We can see that, although WTM is relatively excellent when suggesting top several keyphrases, it performs poor when suggesting more keyphrases due to the poor ability on ensuring coverage.

In Table 6 we list the comparison results of various methods when extracting $N^k = 10$ keyphrases. We can observe that TWTM outperforms the best baseline TAM by 7% of F-measure. Moreover, as mentioned above, the dataset consists of three types of products. In Table 6 we also demonstrate the results of TWTM on the test instances divided by product types, denoted as "BOOK", "MOVIE" and "MUSIC". The performance is consistently decent on the three types of instances. We also observe that F-measure scores on the three types of instances are proportional to the size of their training instances as shown in Table 1. Apparently, more training instances will enhance sufficiently learning of TWTM, which may

Method	Precision	Recall	F-measure	Bpref	MRR
NB	0.283	0.232	0.255	0.283	0.702
CRM	0.267	0.216	0.239	0.267	0.648
TAM	0.310	0.254	0.279	0.310	0.676
WTM	0.242	0.242	0.242	0.242	0.785
TTM	0.226	0.203	0.214	0.226	0.638
TWTM	**0.354**	**0.343**	**0.349**	**0.354**	**0.838**
BOOK	0.365	0.428	0.394	0.365	0.861
MOVIE	0.356	0.274	0.310	0.356	0.820
MUSIC	0.341	0.326	0.334	0.341	0.831

Table 6: Comparison results when extracting $N^k = 10$ keyphrases.

also eventually improve the performance of keyphrase extraction.

Conclusion and Future Work

This paper focuses on keyphrase extraction from a controlled vocabulary. The proposed TWTM has two features: (1) TWTM uses latent topics to represent document themes, and thus takes the coverage of document themes into consideration; (2) TWTM models topic-specific word triggers, which are more discriminative. Hence TWTM is able to bridge the vocabulary gap between document content and keyphrases more precisely. Experiment results on real world dataset demonstrate that TWTM outperforms existing state-of-the-art methods under various evaluation metrics. We also demonstrate that TWTM achieves a balance between word-level projection and topic-level coverage.

Moreover, TWTM is not restricted to supervised learning. TWTM can also be adopted in unsupervised fashion. So long as we can build appropriate translation pairs to represent semantic relations between documents and keyphrases, TWTM will be able to exert its capacity. For example, for news articles, we can use news titles and contents to build translation pairs, by regarding titles as an approximate language to keyphrases; for scientific papers, we can use abstracts and contents to build translation pairs.

We design the following research plans: (1) The number of topics in TWTM requires being pre-defined by users. We plan to incorporate Bayes Nonparametric (Blei et al., 2010) for TWTM to automatically learn the number of topics. (2) The trigger probabilities in TWTM do not take rich linguistic knowledge into consideration. We plan to incorporate more complicated alignment models from SMT into our model. (3) This paper focuses on supervised learning of TWTM. We plan to investigate unsupervised learning of TWTM for documents such as news articles and scientific papers. (4) The influence of product types for keyphrase extraction has not been thoroughly investigated in this paper. We plan to study the impact of product types and explore domain adaptation (Blitzer et al., 2006) for cross-domain keyphrase extraction using TWTM.

Acknowledgments

This work is supported by the National Natural Science Foundation of China (NSFC) under the grant No. 61170196 and 61202140. The authors would like to thank Douban Inc. for providing the anonymized data.

References

Blei, D., Griffiths, T., and Jordan, M. (2010). The nested chinese restaurant process and bayesian nonparametric inference of topic hierarchies. *Journal of the ACM*, 57(2):7.

Blei, D. and Lafferty, J. (2009). *Text mining: Classification, Clustering, and Applications*, chapter Topic models. Chapman & Hall.

Blei, D., Ng, A., and Jordan, M. (2003). Latent dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022.

Blitzer, J., McDonald, R., and Pereira, F. (2006). Domain adaptation with structural correspondence learning. In *Proceedings of EMNLP*, pages 120–128.

Brown, P., Pietra, V., Pietra, S., and Mercer, R. (1993). The mathematics of statistical machine translation: Parameter estimation. *Computational linguistics*, 19(2):263–311.

Buckley, C. and Voorhees, E. (2004). Retrieval evaluation with incomplete information. In *Proceedings of SIGIR*, pages 25–32.

Dempster, A., Laird, N., Rubin, D., et al. (1977). Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)*, 39(1):1–38.

Frank, E., Paynter, G., Witten, I., Gutwin, C., and Nevill-Manning, C. (1999). Domain-specific keyphrase extraction. In *Proceedings of IJCAI*, pages 668–673.

Garg, N. and Weber, I. (2008). Personalized, interactive tag recommendation for flickr. In *Proceedings of RecSys*, pages 67–74.

Grineva, M., Grinev, M., and Lizorkin, D. (2009). Extracting key terms from noisy and multi-theme documents. In *Proceedings of WWW*, pages 661–670.

Heinrich, G. (2005). Parameter estimation for text analysis. *Web: http://www. arbylon. net/publications/text-est*.

Iwata, T., Yamada, T., and Ueda, N. (2009). Modeling social annotation data with content relevance using a topic model. In *Proceedings of NIPS*, pages 835–843.

Krestel, R., Fankhauser, P., and Nejdl, W. (2009). Latent dirichlet allocation for tag recommendation. In *Proceedings of ACM RecSys*, pages 61–68.

Li, X., Snoek, C., and Worring, M. (2009). Learning social tag relevance by neighbor voting. *IEEE Transactions on Multimedia*, 11(7):1310–1322.

Liu, Z., Chen, X., and Sun, M. (2011a). A simple word trigger method for social tag suggestion. In *Proceedings of EMNLP*, pages 1577–1588.

Liu, Z., Chen, X., and Sun, M. (2012). Mining the interests of chinese microbloggers via keyword extraction. *Frontiers of Computer Science*, 6(1):76–87.

Liu, Z., Chen, X., Zheng, Y., and Sun, M. (2011b). Automatic keyphrase extraction by bridging vocabulary gap. In *Proceedings of CoNLL*, pages 135–144.

Liu, Z., Huang, W., Zheng, Y., and Sun, M. (2010). Automatic keyphrase extraction via topic decomposition. In *Proceedings of EMNLP*, pages 366–376.

Liu, Z., Li, P., Zheng, Y., and Sun, M. (2009). Clustering to find exemplar terms for keyphrase extraction. In *Proceedings of EMNLP*, pages 257–266.

Mihalcea, R. and Tarau, P. (2004). Textrank: Bringing order into texts. In *Proceedings of EMNLP*, pages 404–411.

Mimno, D., Wallach, H., Naradowsky, J., Smith, D., and McCallum, A. (2009). Polylingual topic models. In *Proceedings of EMNLP*, pages 880–889.

Nguyen, T. and Kan, M. (2007). Keyphrase extraction in scientific publications. In *Proceedings of the 10th International Conference on Asian Digital Libraries*, pages 317–326.

Ravi, S., Broder, A., Gabrilovich, E., Josifovski, V., Pandey, S., and Pang, B. (2010). Automatic generation of bid phrases for online advertising. In *Proceedings of WSDM*, pages 341–350.

Salton, G. and Buckley, C. (1988). Term-weighting approaches in automatic text retrieval. *Information processing and management*, 24(5):513–523.

Si, X., Liu, Z., and Sun, M. (2010). Modeling social annotations via latent reason identification. *IEEE Intelligent Systems*, 25(6):42–49.

Si, X. and Sun, M. (2009). Tag-LDA for scalable real-time tag recommendation. *Journal of Computational Information Systems*, 6(1):23–31.

Tsoumakas, G. and Katakis, I. (2007). Multi-label classification: An overview. *International Journal of Data Warehousing and Mining (IJDWM)*, 3(3):1–13.

Turney, P. (2000). Learning algorithms for keyphrase extraction. *Information Retrieval*, 2(4):303–336.

Voorhees, E. (2000). The trec-8 question answering track report. In *Proceedings of TREC*, pages 77–82.

Wan, X. and Xiao, J. (2008a). Collabrank: towards a collaborative approach to single-document keyphrase extraction. In *Proceedings of COLING*, pages 969–976.

Wan, X. and Xiao, J. (2008b). Single document keyphrase extraction using neighborhood knowledge. In *Proceedings of AAAI*, pages 855–860.

Witten, I., Paynter, G., Frank, E., Gutwin, C., and Nevill-Manning, C. (1999). Kea: Practical automatic keyphrase extraction. In *Proceedings of DL*, pages 254–255.

Easy-First Chinese POS Tagging and Dependency Parsing

Ji Ma, Tong Xiao, Jingbo Zhu, Feiliang Ren
Natural Language Processing Laboratory
Northeastern University, China

majineu@outlook.com, zhujingbo@mail.neu.edu.cn,
xiaotong@mail.neu.edu.cn, renfeiliang@ics.neu.edu.cn

ABSTRACT

The easy-first non-directional dependency parser has demonstrated its advantage over transition based dependency parsers which parse sentences from left to right. This work investigates easy-first method on Chinese POS tagging, dependency parsing and joint tagging and dependency parsing. In particular, we generalize the easy-first dependency parsing algorithm to a general framework and apply this framework to Chinese POS tagging and dependency parsing. We then propose the first joint tagging and dependency parsing algorithm under the easy-first framework. We train the joint model with both supervised objective and additional loss which only relates to one of the individual tasks (either tagging or parsing). In this way, we can bias the joint model towards the preferred task. Experimental results show that both the tagger and the parser achieve state-of-the-art accuracy and runs fast. And our joint model achieves tagging accuracy of 94.27 which is the best result reported so far.

KEYWORDS: dependency parsing, POS tagging, perceptron, easy-first

Proceedings of COLING 2012: Technical Papers, pages 1731–1746,
COLING 2012, Mumbai, December 2012.

1 Introduction

To sequential labelling problems, such as POS tagging or incremental parsing, traditional approaches 1) decompose the input sequence into several individual items each of which corresponds to a token of the input sequence; 2) predict these items in a fixed left-to-right (or right-to-left) order (Collins 2002; Ratnaparkhi 1996). The drawback of such fixed order approach is that when predicting one item, only the labels on the left side can be used while the labels on the right side is still unavailable. (Goldberg and Elhadad, 2010) proposed the easy-first dependency parsing algorithm to relax the fixed left-to-right order and to incorporate more structural features from both sides of the attachment point. Comparing with a deterministic transition based parser which parses the sentence left-to-right, their deterministic easy-first parser achieves significant better accuracy.

The key idea behind their parsing algorithm is to always process the easy attachments in the early stages. The hard ones are delayed to late stages until more structural features on both sides of the attachment point are accessible so as to make more informed decision. In this work, we further generalize the algorithm of (Goldberg and Elhadad, 2010) to a general sequential labelling framework. We apply this framework to Chinese POS tagging and dependency parsing which has never been studied under the easy-first framework before.

One characteristic of Chinese dependency parsing is that parsing performance can be dramatically affected by the quality of POS tags of the input sentence (Li et al., 2010). Recent work (Li et al., 2010; Hatori et al., 2011; Bohnet and Nivre, 2012) empirically verified that solving POS tagging and dependency parsing jointly can boost the performance of both the two tasks. To further improve tagging and parsing accuracy, we also solve the two tasks jointly. While previous joint methods are either graph-based or transition-based algorithm, in this work we propose the first joint tagging and dependency parsing algorithm under the easy-first framework. In addition, we also adopt a different training strategy to learn the model parameters. Previous approaches all train their joint model with the objective of minimizing the loss between the reference and the predicted output. Those methods make no distinction between the loss caused by POS tagging and the loss caused by dependency parsing. Though such objective is proper for minimizing the total loss, it may not be optimal in terms of pursuing the best result of a certain individual task (neither tagging nor dependency parsing). To this end, our training method also incorporates additional loss which relates to only one of the individual tasks. And the losses are iteratively optimized on the training set. In this way we can bias the joint model towards the preferred task. Similar techniques have been used in parser domain adaptation (Hall et al., 2011). However, to our knowledge, no one has done this in joint tagging and dependency parsing before.

Experimental results show that under the easy-first framework, even deterministic tagger and parser achieve quite promising performance. For Chinese POS tagging, the tagger achieves an accuracy of 93.84[1] which is among the top Chinese taggers reported so far. Moreover, the tagging speed is about 2000 sentences per-second, much faster than the state-of-the-art taggers (Li et al., 2010; Hatori et al., 2011). For Chinese dependency parsing, when the input sentence is equipped with automatically assigned POS tags, the parser achieves an unlabelled score of 77.66 and runs at the speed of more 300 sentences per second. Such accuracy is among the state-of-the-art transition-based dependency parsers even those parsers are enhanced with beam

1 On the same data set, the best tagger so far reported achieves an accuracy of 93.82. Joint methods yield higher accuracy, but are not directly comparable.

search (section 4). For joint tagging and dependency parsing, we achieve significant improvement on both the two sub-tasks. In particular, we achieve the tagging accuracy of 94.27 which is the highest score reported on the same data set so far.

2 Easy-First POS tagging, dependency parsing and joint tagging and parsing

In this section, we first describe a generalized easy-first sequential labelling framework. Then we show how to apply this framework to the task of POS tagging and dependency parsing. Finally, we propose the joint POS tagging and dependency parsing algorithm under this framework.

2.1 Generalized easy-first sequential labelling algorithm

For solving sequential labelling problems, the first step is to decompose the input sequence into a list of small items $t_1, t_2, \ldots, t_n$. The items are then labelled separately. This list of items is the input of the generalized algorithm. In addition, the algorithm also requires a set of task specific labels $\{l_1, l_2, \ldots, l_m\}$ and a labelling function. Take POS tagging for example, t_i corresponds to the i-th word of the input sentence and the label set corresponds to the set of POS tags. The labelling function associates a word with a certain POS tag.

The pseudo-code of the algorithm is shown in Algorithm 1. Initially, all the items are marked as unprocessed (line 2). Then, the algorithm solves each of the unprocessed item t by labelling t with a suitable label l. After that, t is marked as processed. This process repeats until no items left unprocessed (line 3 to line 8).

One thing to note is that the small items are not processed in a $t_1, t_2 \ldots$ to t_n order. Instead, at each step the algorithm automatically chooses an item-label pair according to a scoring function $score(t, l)$ (line 4). This function is not only responsible for selecting a correct label for a certain item but also responsible for determining the order in which each of the items are processed. Ideally, the scoring function prefers to process easy items in the early stages and delay the hard ones to late stages. When dealing with the hard ones, the label information built on both sides of the current item become accessible. In this way, more informed prediction can be made and the extent of error propagation can be limited (Goldberg and Elhadad, 2010).

Algorithm 1 is a generalization of the easy-first parsing algorithm of (Goldberg and Elhadad, 2010). This generalized version can be naturally instantiated to a wide verity of applications.

2.2 Easy-first POS tagging

In this section, we show how to instantiate algorithm 1 into a POS tagger. The problem of POS tagging is to find a sequence of POS tags for the input sentence. For this problem,

- t_i corresponds to i-th word, w_i, of the input sentence.
- L corresponds to the POS tag set, **POS**. We use x to denote a certain tag in **POS.**
- The labelling function, $\text{labelling}^{pos}(w_i, x)$, set x as the POS tag of w_i

A concrete example of tagging the sentence "中国/China 对/to 外/outside world 开放/open 稳步/steadily 前行/advance" ("China's opening up steadily advances") is shown in figure 1. The challenging part of this sentence is w_4, "开放/opening up", which can be either a VV (verb) or a NN (noun). If we process this sentence in a left-to-right order, the word "开放" would be quite difficult. In the easy-first framework, this can be easily handled with the following steps.

Algorithm 1: Generalized Easy-First Sequential Labelling	
Input	T= t_1, t_2, …t_n: a sequence of items to be labelled L={l_1, l_2, …, l_m}: a set of task specific labels labelling: task specific labelling function
1	$nProcessed \leftarrow 0$
2	set each t_i as unprocessed
3	**repeat**
4	$(\bar{t}, \bar{l}) \leftarrow \text{argmax}_{l \in L, t.unprocessed=true} score(t, l)$
5	labelling $(\bar{t}, \bar{l})$
6	set $\bar{t}$ as processed
7	$nProcessed \leftarrow nProcessed + 1$
8	**until** nProcessed = |T|

Initially (step 0), all the six words w_1, …,w_6 are marked as unprocessed. Each step, the tagger enumerates all possible (w, x) pairs and chooses the most confident one according to the scoring function. Since the word "中国/China" always occurs as a NR (proper noun) in the training set, thus at the step 1, ("中国", NR) is selected[2] and the tagger tags "中国" with NR. After this step, "中国" is marked as processed.

At step 2, for those **unprocessed** words, the tagger re-computes the scores of all (w, x) pairs based on the local context, surrounding words and tags within a window, and selects the one with the highest score to deal with. This time, ("外", NN) is selected and the tagger tags "外" as a NN. Similarly, at step 3, the tagger assigns tag P (preposition) to word "对/to". Step 4 and so on.

At the last step (step 6), the only unprocessed word is w_4, "开放". Since for Chinese, an adverb always precede the verb which it modifies and succeeds the noun which is the subject. Therefore, based on the following tags o_5:AD and o_6:VV, the tagger can easily infer that the correct tag for "开放" is NN. After "开放" is tagged, all words are processed and the tagging procedure stops.

We see algorithm 1 can be easily instantiated to POS tagging, a relatively simple task. In the next sections, we show how to apply algorithm 1 to more complicated tasks.

2.3 Easy-first dependency parsing

The easy-first dependency parsing algorithm was originally proposed by (Goldberg and Elhadad, 2010), we re-describe it here for completeness and also to illustrate how algorithm 1 can be instantiated to dependency parsing.

Given an input sentence of n words, the task of dependency parsing is: for each word, find its lexical head which it modifies. One exception is the **head word** of the sentence which does not modify any other word. All the others each modify exactly one word and no one modifies itself. For dependency parsing:

- t_i corresponds to i-th word, w_i, of the input sentence.
- L corresponds to an action set **ACT** which contains three actions {attach_Left, attach_Right, set_Root}.

Note for dependency parsing, each label corresponds to an action which is designed to manipulate a list of partial analysis p_1,…, p_n, called pending (Goldberg and Elhadad, 2010). p_i re-

[2] At each step, the selected item is boldfaced. The underlined items are those marked as processed.

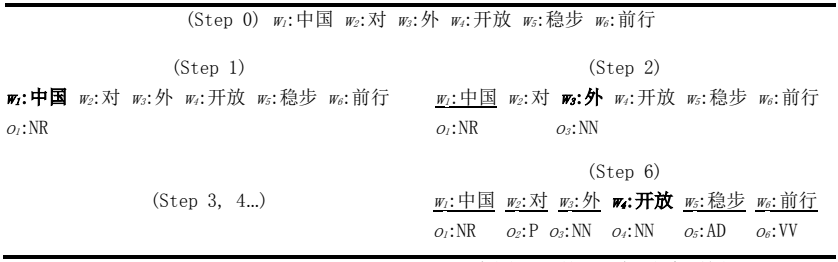


FIGURE 1 – A trace of easy-first tagger for sentence "中国 对 外 开放 稳步 前行" ("China's opening up steadily advances"). o_i denotes the POS tag of w_i

-cords the dependency tree rooted at w_i and p_i is initialized with w_i.

The main loop of easy-first dependency parsing is shown in figure 2. Each step, the parser computes the scores of all possible (p, a) pairs based on local context, surrounding partial analysis within a window, and then selects the best pair to feed to the labelling function, labelling^{dep}. Then labelling^{dep} performs the given action. In particular, labelling^{dep} (p_i, attach_Left) set p_i as the child of its left neighbour in the pending list. labelling^{dep} (p_i, attach_right) set p_i as the child of its right neighbour in the pending list. After that, the selected partial analysis p is marked as processed and removed from the pending list (line 7 to line 8).

Since at each step, one partial analysis is removed from the pending list, after $n-1$ steps, the pending list only contains one item, say p_h, which is the dependency tree of the whole sentence. In such satiation, (p_h, set_Root) is enforced to be the only choice and labelling^{dep} (p_h, set_Root) sets the root of p_h which is w_h as the head of the sentence.

An example of parsing the sentence "中国 对 外 开放 稳步 前行" is shown in figure 3. At the first step, (p_3, attach_Left) is selected and fed to the labelling^{dep} function. The labelling function set p_3 as the child of its left neighbour p_2 as shown in figure 2 (1). p_3 is then removed from the pending list. Note that, after p_3 is removed, p_2 and p_4 become neighbours. The following steps are executed in a similar way except for step 6 where only p_6 is left in the pending list. Thus, at step 6, (p_6, set_Root) is the only choice. The labelling^{dep} function sets w_6, "前行" as the head of the sentence. As all partial analysis are marked as processed, the parsing process stops. Head-modifier relationships can be directly read-off the dependency tree. That is, parent is the head of its children.

```
1   Set nProcessed ← 0
2   p_1, ... p_n ← w_1, ... w_n
3   Set each p_i as unprocessed
4   repeat
5     (p̄, ā) ← argmax_{p∈Pending, a∈ACT} score(p, a)
6     labelling^dep(p̄, ā)
7     set p̄ as processed,
8     remove p̄ from pending
9     nProcessed ← nProcessed + 1
10  until nProcessed = |T|
```

FIGURE 2 – Main loop of easy-first parsing algorithm

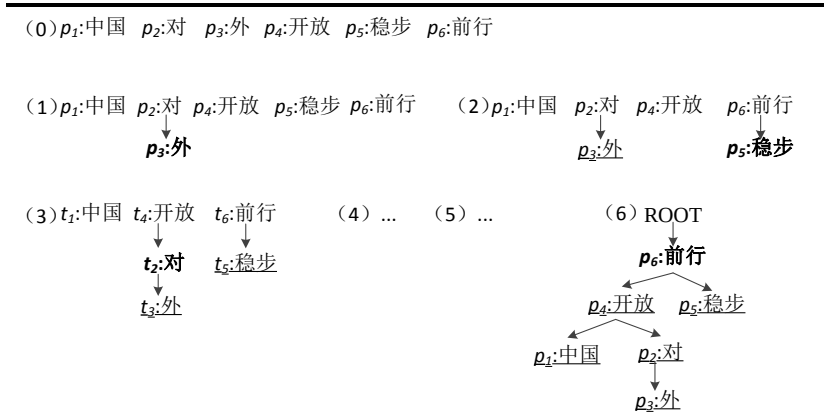


FIGURE 3 – A trace of easy-first parser for sentence "中国 对 外 开放 稳步 前行" ("China's opening up steadily advances")

2.4 The proposed easy-first joint POS tagging and dependency parsing method

In this section, we describe the joint tagging and dependency parsing method which is under the easy-first framework. Before we go into the details about the algorithm, one thing should be mentioned. That is, for dependency parsing, POS tag serves as indispensable information (Li et al., 2010). On the one hand, parsing barely based on bi-lexical features without POS tags hurt performance dramatically. On the other hand, if we tag the whole sentence first and then do dependency parsing, then syntactic features which are demonstrated to be effective for tagging disambiguation cannot be utilized. In order to provide POS tags for syntactic disambiguation and provide syntactic features for tagging disambiguation, we add a simple constraint to control the inference order: dependency relationships can only be extracted between two tagged words. Under such constraint, we can guarantee that at least POS tags for the pair of words from which dependency relation is to be extracted are already available. Also, syntactic feature can be utilize-

```
1   Set nProcessed ← 0,
2   p_1, ... p_n ← w_1, ... w_n
3   Set each p_i as unprocessed
4   repeat
5     Initialize valid
6     (p̄, l̄) ← argmax_{(p,l)∈valid} score(p, l)
7     labelling^Joint(p̄, l̄)
8     if (l̄ ∈ ACT)
9       Set p̄ as processed,
10        remove p̄ from pending
11      nProcessed ← nProcessed + 1
12  until nProcessed = |T|
```

FIGURE 4 – Main part of the joint tagging and dependency parsing algorithm

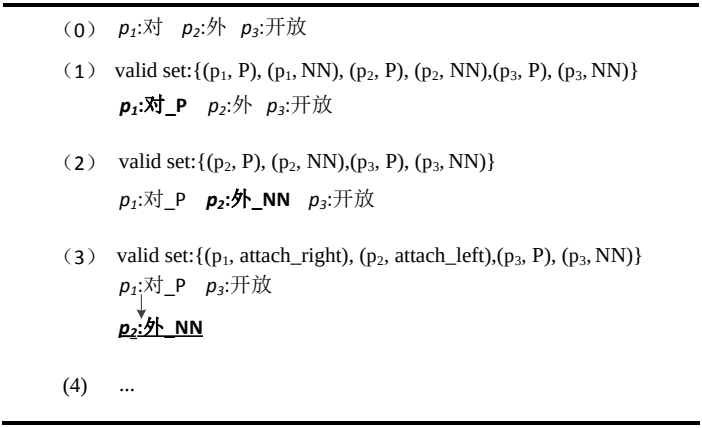


FIGURE 5 – A trace of easy-first joint tagging and parsing for "对 外 开放"

-d to process the untagged words. Under such constraint, the joint method can be constructed by a simple combination of easy-first POS tagging and easy-first dependency parsing.

Similar to previous tasks, for the joint task, t_i still corresponds to the i-th word of the input sentence w_i. Pending list is also used to record the partial dependency tree structures. The label set of the joint task is the union of **POS** and **ACT**. The labelling function of the joint task, $\text{labelling}^{\text{Joint}}$, behaves differently on the two types of labels. Particularly, $\text{labelling}^{\text{Joint}}$ (p_i, x) calls $\text{labelling}^{\text{POS}}$ (w_i, x) which associates w_i, the root of p_i, with POS tag x. $\text{labelling}^{\text{Joint}}$ (p_i, a) calls $\text{labelling}^{\text{Dep}}$ (p_i, a) which performs action a on p_i.

The main part of the joint method is shown in figure 4. Note that, to satisfy the constraint described above, at the beginning of each step, a **valid** set of (p, l) pairs are initialized (line 5). For the partial analysis p, if its root word is not yet tagged, only (p, x) are considered as valid where x is a certain POS tag. This enforces that partial analysis must be tagged first. For p which is already tagged, if its neighbour is also tagged, then attaches are allowed between the two neighbours.

After initializing the **valid** set, the joint algorithm computes scores of the (p, l) pairs from the **valid** set. Then the one with the highest score is selected and be fed to $\text{labelling}^{\text{Joint}}$. A partial analysis p is marked as processed only when p is attached to its neighbours or be set as the root of the sentence (line 8 to line 11). Figure 5 gives a toy example of the joint tagging and parsing procedure. The **valid** set of each step is also shown in the figure. For simplicity, we suppose that the POS tag set only contains two tags NN (noun) and P (preposition).

In summary, the key idea behind easy-first method is to always choose the easy items to process. The degree of easy or hard is determined by the scoring function. In the next section, we show how to learn the scoring function from training data.

3 Training

The scoring functions used in this work are all based on a linear model.

$$score(t, l) = \vec{w} \cdot \varphi(t, l)$$

Here, $\vec{w}$ is the model's parameter or weight vector and $\varphi(t, l)$ is the feature vector extracted from (t, l) pair. In this section, we first describe a general training algorithm which can be used to train models for all the tasks mentioned above and then introduce our training method which is specially designed for the joint model.

3.1 Easy-first training

The generalized easy-first training algorithm is shown in algorithm 2 which is based on the structured perceptron. Starting from a zero weight vector, the training process makes several iterations through the training set. Each iteration, algorithm 2 is utilized to process the training instances. A training instance consists of a sequence of items T together with its gold reference *R.* Here, R takes different forms in different tasks. For example, in POS tagging, R is the gold tag sequence of the input sentence. In dependency parsing, R is the gold dependency tree of the input sentence.

When $(\bar{t}, \bar{l})$ is not compatible[3] with R, the training algorithm updates the current parameter by punishing the features fired in the chosen $(\bar{t}, \bar{l})$ pair and rewarding the features fired in the (t, l) pair which is compatible with R. The algorithm will then move on to the next instance. Note that there might be more than one such pair that are compatible with R. For example, for figure 1 step (2), (w_2, P), (w_3, NN), (w_4, NN), (w_5, AD) and (w_6, VV) are all compatible with the gold tag sequence. Thus, at the beginning of each step, a **compatible** set of (*t, l*) pairs are initialized and

Algorithm 2: Easy-First Training

Input T: $t_1, \ldots, t_n$, R: gold reference of T, $labelling$: task specific function L:{ $l_1, \ldots, l_n$}, $\vec{w}$: feature weight vector, isAllowed: task specific function

Output $\vec{w}$: feature weight vector

$nProcessed \leftarrow 0$
set each t_i as unprocessed
repeat
  $\boldsymbol{compatible} \leftarrow \{(t, l) | isAllowed(t, l, R)_{\,t.state=unprocessed, l \in L} = true\}$
  $(\bar{t}, \bar{l}) \leftarrow \text{argmax}_{l \in L, t.state=unprocessed}\, score(t, l)$
  if $(\bar{t}, \bar{l}) \in \boldsymbol{compatible}$
    $labelling(\bar{t}, \bar{l})$,
    $\bar{t}.state \leftarrow processed$,
    $nProcessed \leftarrow nProcessed + 1$
  else
    $(\hat{t}, \hat{l}) \leftarrow \text{argmax}_{(t,l) \in \boldsymbol{goldAllowed}}\, score(t, l)$
    $\vec{w} \leftarrow \vec{w} + \varphi(\hat{t}, \hat{l}), \quad \vec{w} \leftarrow \vec{w} - \varphi(\bar{t}, \bar{l})$
    break
until $nProcessed = |T|$
return $\vec{w}$

[3] A (*t, l*) pair is compatible with *R* means that in *R*, *t* is labelled with *l*.

the one with the highest score is chosen to boost (Goldberg and Elhadad, 2010). Function isAllowed is used to check whether a candidate (t, l) pair belongs to the **compatible** set (line 4). Similar to the labelling function, isAllowed can be easily instantiated for different tasks. For POS tagging, $\text{isAllowed}^{\text{POS}}$ (w_i, x, R) checks whether x is the gold tag of w_i. For dependency parsing, $\text{isAllowed}^{\text{Dep}}$ (p_i, a, R) checks:

- Whether all p_i's children supposed by the gold dependency tree has already attached to p_i.
- Whether action a attaches p_i to the correct partial analysis.

For joint POS tagging and dependency parsing, $\text{isAllowed}^{\text{Joint}}$ behaves differently according to the types of labels:

- $\text{isAllowed}^{\text{Joint}}$ (p_i, x, R) returns $\text{isAllowed}^{\text{POS}}$ (w_i, x, R).
- $\text{isAllowed}^{\text{Joint}}$ (p_i, a, R) returns $\text{isAllowed}^{\text{Dep}}$ (p_i, a, R).

3.2 Training with additional loss

Algorithm 2 is a variant of the structured perceptron algorithm. The objective is to minimize the loss $\ell(R, \bar{R})$, usually hamming loss (Daumé III et al., 2009), between gold reference R and the predicted output $\bar{R}$. Whenever there is a positive loss, parameters are updated. For POS tagging, algorithm 2 minimizes $\ell^{POS}(R^{POS}, \bar{R}^{POS})$ where R^{POS} and $\bar{R}^{POS}$ are predicted and gold tag sequence respectively. For dependency parsing, algorithm 2 minimizes $\ell^{Dep}(R^{Dep}, \bar{R}^{Dep})$ where R^{Dep} and $\bar{R}^{Dep}$ are predicted and gold dependency tree, respectively.

For the joint task, algorithm 2 minimizes $\ell^{Joint}(< R^{POS}, R^{Dep} >, < \bar{R}^{POS}, \bar{R}^{Dep} >)$ where $< R^{POS}, R^{Dep} >$ denotes the pair of predicted tag sequence and dependency tree. Such loss has no distinction between tagging and parsing: either tagging error or parsing error will lead to non-zero loss and parameters are updated. This loss may not optimal in terms of achieving the best result of the individual tasks.

Inspired by (Hall et al., 2011), we slightly modify algorithm 2 to incorporate additional loss which only relates to one of the individual tasks so as to train the joint model towards the that task. Algorithm 3 shows the pseudo-code. The basic idea is that at each iteration of the training process, one of the three losses $\{\ell^{POS}, \ell^{Dep}, \ell^{Joint}\}$ is selected and parameters are updated acco-

Algorithm 3: Training with additional loss for joint POS tagging and parsing

Input T: $w_1, \ldots, w_n$, R:gold reference of T, $labelling^{Joint}$, $\boldsymbol{L^{Joint}}$, $\vec{w}$, $\text{isAllowed}^{\text{Joint}}$, ℓ: which can be either ℓ^{POS} or ℓ^{Dep} or ℓ^{Joint}.

Output $\vec{w}$: feature weight vector

1 Set $nProcessed \leftarrow 0$, $p_1, \ldots p_n \leftarrow w_1, \ldots w_n$
2 Set each p_i as unprocessed
3 **repeat**
4 Initialize $\boldsymbol{valid}$
5 $\boldsymbol{compatible} \leftarrow \{(p,l) | isAllowed^{Joint}(p,l)_{(p,l)\in \boldsymbol{valid}} = true\}$
6 **if** $(\ell = \ell^{POS})$
7 $\boldsymbol{compatible} \leftarrow \boldsymbol{compatible} \cup \{(p,l) | (p,l) \in \boldsymbol{valid} \wedge l \in \boldsymbol{ACT}\}$
8 **if** $(\ell = \ell^{Dep})$
9 $\boldsymbol{compatible} \leftarrow \boldsymbol{compatible} \cup \{(p,l) | (p,l) \in \boldsymbol{valid} \wedge l \in \boldsymbol{POS}\}$
10 $(\bar{p}, \bar{l}) \leftarrow \text{argmax}_{(p,l)\in \boldsymbol{valid}} score(p,l)$
11 **if** $(\bar{p}, \bar{l}) \in \boldsymbol{compatible}$
12 ... (*the following are exactly the same as algorithm 2*)

-rdingly: if ℓ^{Joint} is selected, then both tagging and parsing error cause parameter update; if ℓ^{POS} is selected, then only tagging errors can cause parameter update. This can be done by adding all valid attach actions to the **compatible** set regardless whether those actions are indeed compatible with the gold reference (line 6 to line 7); For ℓ^{Dep}, only parsing errors cause parameter update which can be achieved similar to the case of ℓ^{POS}.

4 Experiments

To make comparison with previous works, we use Penn Chinese Treebank 5.1 (CTB5) (Xue et al., 2005) to evaluate our method. We use the standard split of CTB5 as described in (Duan et al., 2007): section 001-815 and 1001-1136 are used as training set, section 886-931 and 1148-1151 are used as development set, section 816-885 and 1137-1147 are used as test set. Head finding rules of (Zhang and Clark 2008b) are used to convert the constituent trees into dependency trees. An Intel Core i7 870 2.93 GHz machine is used for evaluation. For POS tagging and dependency parsing, the number of training iterations are selected according to the model's performance on the development set. The model which achieves the highest score on the development set is selected to run on the test set.

4.1 POS tagging

Following Zhang and Clark (2008a), in the experiments, we also use a dictionary to limit the number of candidate tags for each word. That is, for a word which occurs more than M times in the training data, the tagger only considers the POS tags co-occurred with that word in the training data. We found 6 to be the optimal value on the development set. The feature templates we used in the experiments are shown in table 1. Z&C08 denotes the features templates of (Zhang and Clark, 2008a) which include uni-gram and bi-gram as well as some character based features. In addition to Z&C08, we also incorporate bi-directional POS tag features and the resulted feature set is denoted by feature set BD. For feature set OP, we include some order preference features with the goal to signal to the tagger that some words should be delayed to late stages until more surrounding tags are available and more informed prediction can be made.

Table 2 shows the performance for different feature set and also gives state-of-the-art results on the same data set. "Li-10" denotes the performance of the tagger of (Li et al., 2010) and Hatori-11 denotes the performance of the tagger of (Hatori et al., 2011). From table 2, we can see

Feature sets	Tagging Feature Templates
Z&C08	w_i, w_{i+1}, w_iw_{i+1}, w_iw_{i-1}, $FC(w_i)$, $LC(w_i)$, $TS(FC(w_i))$, $TS(LC(w_i))$, $C_n(w_i)$ (n=2…len(w_i) -1), $LC(w_{i-1})w_iFC(w_{i+1})$, $FC(w_i)C_n(w_i)$ (n=2…len(w_i)), $LC(w_i)C_n(w_i)$ (n=1…len(w_i) -1), $C_n(w_i)$ (if $C_n(w_i) = C_{n+1}(w_i)$), tag_{i-1}, $tag_{i-1}tag_{i-2}$
BD	Z&C08 + tag_{i+1}, $tag_{i+1}tag_{i+2}$, $tag_{i-1}tag_{i+1}$
OP^{POS}	BD + $w_itagged_{i+1}$, $w_itagged_{i-1}$, $w_itagged_{i-2}$, $w_itagged_{i+2}$ $w_itagged_{i-2}tagged_{i-1}$, $w_itagged_{i-1}tagged_{i+1}$, $w_itagged_{i+1}tagged_{i+2}$

TABLE 1 – Feature templates for easy-first POS tagging. Here w_i denotes the i-th word of the input sentence, tag_i denotes the POS tag of w_i, $FC(w_i)$/$LC(w_i)$ denotes the first/last character of w_i. $C_n(w_i)$ is the n-th character of w_i, TS(c) is the POS tag set that co-occurred with character c in the dictionary. $tagged_k$ denotes whether w_k has already been tagged

Feature Sets	Development Set			Test Set			Speed[4]
	Total	Seen	Unseen	Total	Seen	Unseen	
Z&C08	93.58	94.59	**77.16**	93.36	94.22	80.49	3001
BD	93.85	94.98	75.65	93.68	94.53	**80.83**	2733
OP	94.05	**95.13**	76.65	**93.84**	**94.73**	80.49	2070
Li-10	–	–	–	93.51	94.36	80.78	292
Hatori-11	**94.15**	–	–	93.82	–	–	210

TABLE 2 – POS tagging performance

that bi-directional features are effective for improving tagging performance. With bi-directional features, the tagger achieves the accuracy of 93.68 which is higher than Li-10 and slightly lower than Hatori-11. By incorporating order preference features, tagging accuracy increases to 93.84 better than both Li-10 and Hatori-11. Moreover, rather than using Viterbi or beam search, our easy-first tagger is deterministic which is easy to implement and runs in high speed, more than 2000 sentence per second.

4.2 Dependency parsing

We use root accuracy, complete match rate and word accuracy or dependency accuracy to evaluate the parser's performance. Feature templates for easy-first dependency parsing are shown in table 3. G&E10 denotes the feature templates used in (Goldberg and Elhadad, 2010) with some modification: the feature templates in the last row of G&E10 were originally designed to deal with English PP attachment ambiguity. Those templates are limited to be used only when

Feature sets	Parsing Feature Templates
G&E10	for p in p_{i-2}, p_{i-1}, p_i, p_{i+1}, p_{i+2}, p_{i+3}: len(p), nc(p)
	for p q in(p_{i-2}, p_{i-1}),(p_{i-1}, p_i),(p_i, p_{i+1}),(p_{i+1}, p_{i+2}),(p_{i+2}, p_{i+3}): dis(p, q), dis(p, q)$tag_p tag_q$
	for p in p_{i-2}, p_{i-1}, p_i, p_{i+1}, p_{i+2}, p_{i+3}: tag_p, w_p, $tag_p lc_p$, $tag_p rc_p$, $tag_p lc_p rc_p$
	for p q in(p_i, p_{i+2}),(p_{i-1}, p_i),(p_i, p_{i+1}),(p_{i-1}, p_{i+2}),(p_{i+1}, p_{i+2}): $tag_p tag_q$, $tag_p tag_q lc_p lc_q$, $w_p w_q$, $tag_p w_q$, $w_p tag_q$, $tag_p tag_q lc_p rc_q$, $tag_p tag_q rc_p lc_q$, $tag_p tag_q rc_p rc_q$
	$w_{pi-1} w_{pi} rc_{pi}$, $tag_{pi-1} w_{pi} rcw_{pi}$, $w_{pi-1} w_{pi+1} rc_{pi+1}$, $tag_{pi-1} w_{pi+1} rcw_{pi+1}$, $w_{pi} w_{pi+1} rc_{pi+1}$ $tag_{pi} w_{pi+1} rcw_{pi+1}$, $w_{pi+1} w_{pi+2} rc_{pi+2}$, $tag_{pi+1} w_{pi+2} rcw_{pi+2}$, $w_{pi} w_{pi+2} rc_{pi+2}$, $tag_{pi+1} w_{pi+2} rcw_{pi+2}$
VTT^{DEP}	for p in p_{i-2}, p_{i-1}, p_i, p_{i+1}, p_{i+2}: $w_p$vl(p) , $w_p$vr(p), tag_pvl(p), tag_pvr(p), $r_2 cw_p$, $r_2 c_p$, $l_2 cw_p$, $l_2 c_p$, $tag_p lc_p$ rc_p
	$tag_{pi-2} tag_{pi-1} tag_p$, $tag_{pi-1} tag_{pi} tag_{p+1}$, $tag_{pi} tag_{pi+1} tag_{p+2}$
OP^{DEP}	for p, q in (p_i, p_{i-1}),(p_i, p_{i-2}),(p_i, p_{i+1}),(p_i, p_{i+2}),(p_i, p_{i+3}):$w_p$nc(q), tag_pnc(q), $tag_p tag_q$nc(q),
	for p, q, r in (p_i, p_{i-1}, p_{i-2}),(p_i, p_{i-1}, p_{i+1}),(p_i, p_{i+1}, p_{i+2}): $w_p$nc(q) nc(r), tag_pnc(q) nc(r)

TABLE 3 – Feature template used for dependency parsing. For a partial dependency tree p, len(p) is the number of words in p. nc(p) denotes whether p is a leaf node. w_p and tag_p denote p's root word and the POS tag of p's root word, respectively. lc_p/rc_p denote the POS tag of p's left/right most child. lcw_p/rcw_p denotes the word form of p's left/right most child. $l_2 c_p$/$r_2 c_p$ denote the POS tag of p's second left/right most child. $l_2 cw_p$/$r_2 cw_p$ denotes the word form of p's second left/right most child

[4] Since experiments in this work and that in the other work are carried on different machines, speed is for reference only.

		H&S	Z&N	H&S-H	Z&N-H	Li-O2	Li-O3	G&E10	VTT	OP
GoldPOS	Word	85.20	86.00	85.12	85.96	**86.18**	86.00	84.62	85.18	85.22
	Root	78.32	–	78.30	**80.87**	78.58	77.59	74.70	75.38	75.48
	Compl	33.72	**36.90**	32.77	35.03	34.07	34.02	36.12	36.27	36.80
Tag Accuracy		–	–	93.82		93.51		93.84		
AutoPOS	Word	–	–	77.13	**78.04**	–	–	77.45	77.64	77.66
	Root	–	–	72.49	**75.55**	–	–	68.50	68.92	68.35
	Compl	–	–	25.13	26.07	–	–	**28.89**	28.19	28.45
J-N	Word	–	–	–	–	79.03	**79.29**	78.43	78.73	78.87
	Root	–	–	–	–	**74.70**	74.65	67.14	68.29	68.50
	Compl	–	–	–	–	27.19	27.24	28.98	**29.34**	29.29
Speed		–	–	32.7	9	5.8	2	**391**	385	355

TABLE 4 – Parsing performance. H&S-10 and Z&N-11 denote parsers in Huang and Sagae (2010) and Zhang and Nirve (2011), respectively. H&S-H and Z&N-H denote Hatori et al., (2011)'s re-implementation of H&S-10 and Z&N-11, respectively. Li-10-O2/O3 denotes the 2rd/3rd graph based model of Li et al., (2010)

either tag_{p_i}or $tag_{p_{i+1}}$or $tag_{p_{i+2}}$is a preposition. For Chinese, PP attachment ambiguity is not as prevalent as that of English (Huang et al., 2009) and we found that use these features without any limitation yields better results. VTT includes valence features, tri-gram features and third order features which were proved useful for transition based parsers (Zhang and Nivre, 2011). For OP, some additional order preference feature templates are added.

Parsing results are shown in table 4. "GoldPOS" denotes the input with gold standard POS tag. "AutoPOS" denotes that the training set are assigned with gold standard POS tag while the test set are tagged by our easy-first tagger. "J-N" denotes that we use 10-fold Jack-Nifing to train the model. "Tag Accuracy" denotes the test set tagging accuracy. From table 4, we can see that valence and tri-gram features are also effective for easy-first parser. For GoldPOS, word accuracy boosted from 84.62 to 85.18. For AutoPOS and J-N, word accuracy also increased about 0.2 and 0.3 point, respectively. Order preference features are not as effective as it were for tagging. After adding these features, parsing performance rarely changed. One reason might be that some of the features of G&E10 already capture order information and the order preference features list in table 3 redundant to some degree. Comparing with other state-of-the-art parsers on this data set, the performance of the easy-first parser is still lower. This may due to the fact that our parser is greedy, thus more vulnerable to error propagation. Interestingly, for AutoPOS and J-N, the easy-first parser achieves the highest complete match rate. This is consistent with (Goldberg and Elhadad, 2010). One thing should be mentioned is that, the deterministic easy-first parser is both easy to implement and runs very fast, more than 350 sentences per second.

4.3 Joint tagging and parsing

Similar to (Hatori et al., 2011), for the joint task, we choose the model which performs the best on the development set in terms of word accuracy to run on the test set. Feature templates for joint POS tagging and dependency parsing are shown in table 5. The feature set is the union of the feature set used for POS tagging and the feature set used for dependency parsing. Besides, in-

Feature sets	Feature templates for Joint tagging and parsing
Syn	nc(p_{i-1}) $w_{p_{i-1}}$, nc(p_{i-1}) $tag_{p_{i-1}}$ nc(p_{i+1}) $w_{p_{i+1}}$, nc(p_{i+1}) $tag_{p_{i+1}}$ nc(p_{i-1})w_i $tag_{p_{i-1}}$, nc(p_{i-1})$w_i w_{p_{i-1}}$, nc(p_{i-1}) $tag_{p_{i-1}} lc_{p_{i-1}}$ nc(p_{i-1}) $tag_{p_{i-1}} rc_{p_{i-1}}$ nc(p_{i+1})$w_i tag_{p_{i+1}}$ nc(p_{i+1})$w_i w_{p_{i+1}}$ nc(p_{i+1}) $tag_{p_{i+1}} lc_{p_{i+1}}$ nc(p_{i+1}) $tag_{p_{i+1}} rc_{p_{i+1}}$
VVT^{Joint}	Syn + VVT^{DEP} + OP^{POS}

Table 5 Feature templates for joint POS tagging and dependency parsing.

-spired by (Hatori et al., 2011), we also incorporate some syntactic features that aim to improve tagging accuracy and theses features are denoted by Syn.

Table 6 shows the results for the joint model with different losses. Parsing performance, especially word level accuracy, is largely affected by the different loss settings. When training the joint model with a single loss ℓ^{Joint}, word level accuracy is 79.12. When training with $\ell^{Joint}$and $\ell^{Parsing}$, word level accuracy increased to 79.91. These results demonstrate that our training method can bias the joint model towards the desired task. However, as we try different losses, tagging accuracy rarely changes. This may because that the tagging accuracy is already very high and it is quite difficult to achieve further improvement.

Comparing with previous results on joint POS tagging and dependency parsing, our method achieves the best result in terms of complete match rate and POS tagging accuracy[5]. The word accuracy is still below the best result. This may due to the fact that our joint decoder is deterministic thus suffers more from error propagation comparing with beam search based or dynamic programming based decoders. However, our joint method can also be enhanced with beam search and we leave it to future work.

Model	losses	POS	Word	Root	Compl	Speed
Joint	ℓ^{Joint}	94.25*	79.12	72.02	30.66	70+
	$\ell^{Joint}, \ell^{Parsing}$	94.26*	79.91*	72.81	**30.76**	70+
	ℓ^{Joint}, ℓ^{POS}	**94.27***	79.04	71.44	30.29	70+
Pipeline	–	93.84	78.73	68.29	29.34	

Other Methods	POS	Word	Root	Compl	Speed
B&N-12	93.24	81.42	–	–	–
Li-10-V1-O2	93.08	80.74	75.80	28.24	1.7
Li-10-V2-O3	92.80	80.79	75.84	29.11	0.3
Z&N-Hatori	93.94	**81.33**	**77.92**	29.90	1.5
H&S-Hatori	94.01	79.83	73.86	27.85	9.5

TABLE 6 – Joint tagging and parsing results. B&N-12 denotes the result of (Bohnet and Nivre, 2012). Li-10-V1-O2 and Li-10-V2-O3 denote results from (Li et al., 2010). Z&N-Hatori and H&S-Hatori denote results from (Hatori et al., 2011). "*" denotes statistical significant ($p < 0.05$ by MeNemar's test) comparing with the pipeline method.

[5] Comparing with easy-first tagger, the joint model does better at disambiguiting NN-VV, DEC-DEG while poorer at JJ-NN. This result is similar to previous result (Hatori et al, 2011). For limited space, we omit the confusion matrix.

5 Related Work

The easy-first dependency parsing algorithm was first proposed by (Goldberg and Elhadad, 2010), they applied the algorithm to English dependency parsing and by combining results with transition based parser and graph based parser, state-of-the-art performance is achieved. This work is a generalization to their algorithm, and we applied the generalized algorithm to Chinese POS tagging and dependency parsing and also achieves good results in terms of both speed and accuracy. We also proposed the first easy-first joint tagging and parsing algorithm. By incorporating additional loss during training, our method achieves the best tagging accuracy reported so far. Shen et al. (2007) proposed a bi-directional POS tagging algorithm which achieves state-of-the-art accuracy on English POS tagging. Comparing to their method, our tagging algorithm in this paper is much simpler and we are the first to use order preference features in POS tagging. Also this is the first work that applies easy-first tagging on Chinese.

For joint POS tagging and (unlabelled, projective) dependency parsing, Li et al. (2010) proposed the first graph based algorithm. Lee et al. (2011) proposed a graphical model to solve the joint problem. Hatori et al. (2011) proposed the first transition based algorithm. Bohnet and Nivre (2012) extended Hatori et al. (2011) to labelled non-projective dependency parsing. Different from the works talked above, our method is based on the easy-first framework. In addition, all previous joint methods optimize a single loss in the training phase while we are the first to train the joint model with additional loss.

Hall et al. (2011) proposed the augmented-loss training for dependency parser that aims at adapting the parser to other domains or to downstream tasks such as MT reordering. They extended structured perceptron with multiple losses each of which is associated with an external training set. Our method is directly inspired by Hall et al. (2011). However, rather than domain adaptation, our method aims at training the joint model to pursue the best result of one of the individual task. Moreover, our method optimizes all loss on a single training set.

6 Conclusion

In this paper, we generalize the method of (Goldberg and Elhadad, 2010) to a general framework and apply the framework to Chinese POS tagging and dependency parsing. We also proposed the first joint tagging and dependency parsing algorithm under the easy-first framework. We show that by using order preference features, an easy-first POS tagger can achieve the state-of-the-art accuracy. We show a deterministic easy-first parser can surpass the transition-based parser when the input is associated with automatically generated tags. We also illustrate that by incorporating additional loss in the training process, we can bias the joint model towards the desired task.

Acknowledgments

We would like to thank Mu Li, ChangNing Huang, Yova Goldberg, Nan Yang, Zhanghua Li and Jun Hatori for frequent discussions. We also thank Muhua Zhu, Wenliang Chen, Nan Yang and three anonymous reviewers for their insightful suggestions on earlier draft of this paper. This work was supported in part by the National Science Foundation of China (61100089, 61073140, 61272376, 61003159), specialized Research Fund for the Doctoral Program of Higher Education (20100042110031) and the Fundamental Research Funds for the Central Universities (N110404012).

References

Bohnet, B. and Nivre J. (2012) A Transition-Based System for Joint Part-of-Speech Tagging and Labeled Non-Projective Dependency Parsing. (EMNLP 2012).

Collins, M. (2002). Discriminative Training Methods for Hidden Markov Models: Theory and Experiments with Perceptron Algorithms.(EMNLP 2002), pages 1-8

Daumé III, H., Langford, J. and Marcu, D. (2009) Search-based structured prediction. In Journel of Machine Learning, 75(3): 297-325.

Duan, X., Zhao, J. and Xu, B. (2007) Probabilistic parsing action models for multilingual dependency parsing. (EMNLP-CoNLL 2007)

Goldberg, Y. and Elhadad, M. (2010) An Efficient Algorithm for Eash-First Non-Directional Dependency Parsing. (NAACL 2010), pages

Hall, K., McDonald, R., Katz-Brown, J. and Ringgaard, M. (2011) Training dependency parsers by jointly optimizing multiple objectives. (EMNLP 2011)

Hatori, J., Matsuzaki, T., Miyao, Y. and Tsujii, J. (2011) Incremental Joint POS Tagging and Dependency Parsing in Chinese. (IJCNLP 2011), pages 1216-1224, Chiang Mai, Thailand.

Huang, L. Jiang, W. and Liu, Q. (2009) Bilingually-Constrained (Monolingual) Shift-Reduce Parsing. (EMNLP 2009), pages 1222-1231, Singapore.

Huang, L. and Sagae, K. (2010) Dynamic programming for linear-time incremental parsing. (ACL 2010).

Lee, J., Naradowsky, J. and Smith, D.A. (2011) A discriminative model for joint morphological disambiguation and dependency parsing. (ACL 2011)

Li, Z., Zhang, M., Che, W., Liu, T. Chen, W. and Li, H. (2010) Joint Models for Chinese POS Tagging and Dependency Parsing (EMNLP 2010), pages 1180-1191, Edinburgh.

Rataparkhi, A. (1996) A Maximum Entropy Part-Of-Speech Tagger. (EMNLP 1996)

Shen, L., Satt, G. and Joshi, A. K. (2007) Guided Learning for Bidirectional Sequence Classification. (ACL 2007), pages 760-767, Prague.

Tsuruoka, Y. Tsujii, J. (2005). Bidirectional inference with the easiest-first strategy for tagging sequence data. (EMNLP 2005) pages 467-474.

Xue, N., Xia, F., Chiou, F. and Palmer, M. (2005) The Penn Chinese Treebank: Phrase structure annotation of a large corpus. In Natural Language Engineering, volume 11, pages 207-238.

Zhang, Y. and Clark, S. (2008a) Joint Word Segmentation and POS Tagging Using a Single Perceptron. (ACL 2008), Ohia.

Zhang, Y. and Clark, S. (2008b) A tale of two parsers: investigating and combining graph-based and transition-based dependency parsing using beam-search. (EMNLP 2008), Hawaii.

Zhang, Y. and Nivre, J. (2011) Transition-based Dependency Parsing with Rich Non-local Features. (ACL 2011), Portland.

Recognizing personal characteristics of readers using eye-movements and text features

Pascual Martínez-Gómez[1,2] *Tadayoshi Hara*[2] *Akiko Aizawa*[1,2]
(1) The University of Tokyo
(2) National Institute of Informatics
{pascual, harasan, aizawa}@nii.ac.jp

ABSTRACT
In the present work we raise the hypothesis that eye-movements when reading texts reveal task performance, as measured by the level of understanding of the reader. With the objective of testing that hypothesis, we introduce a framework to integrate geometric information of eye-movements and text layout into natural language processing models via image processing techniques. We evidence the patterns in reading behavior between subjects with similar task performance using principal component analysis and quantify the likelihood of our hypothesis using the concept of linear separability. Finally, we point to potential applications that could benefit from these findings.

KEYWORDS: eye-tracking, natural language processing, image recognition.

TITLE AND ABSTRACT IN JAPANESE

視線の動きとテキスト素性を用いた読み手の個性認識

本研究では、テキストを読む際の視線の動きから、テキストの理解度によって測定されるような読み手のタスクパフォーマンスが予測可能である、という仮説を立てる。この仮説を検証するため、我々はまず、画像処理技術を介して、視線の動きとテキスト配置に関する位置情報を、自然言語処理のモデルとして統合する枠組を導入する。次に我々は、近いタスクパフォーマンスの被験者間に共通した読解行動のパターンを主成分分析によって同定し、この線形分離可能性を求めることで我々の仮説の蓋然性を定量的に示す。最後に、我々はこれらの発見から恩恵を受け得る応用例について述べる。

KEYWORDS: 視線追跡、自然言語処理、画像認識.

Proceedings of COLING 2012: Technical Papers, pages 1747–1762,
COLING 2012, Mumbai, December 2012.

1 Introduction

Reading is a common activity that is part of the process of information transfer between humans. However, despite of the important role it has played in recent history and its current wide use, this process of information transfer is not well understood. The difficulty in modelling the reading activity stems from the direct unobservability of human mental states and how the information is decoded and integrated into the brain, or simply forgotten.

The reading act can be seen as an interaction between the reader and the writer where the information and other aspects of the communication are transferred via the document. If we could observe the detailed editing process of a writer, that would indeed give us valuable information about the writer, useful to interpret the message to be sent. However, information on the editing process is usually not available, but the final result in the form of a document is. On the other side of the communication channel, the reader does not necessarily evidence any reading actions except for the movements of the eyes, and that is all we have to understand the reading process.

Several psycholinguistic studies (Rayner, 1998; McDonald and Shillcock, 2003) have shown that document characteristics influence on cognitive processing and that they are reflected on eye-movements in an on-line manner. There have been attempts to also quantify the influence of textual linguistic characteristics on reading behavior for certain types of reading tasks (Martínez-Gómez et al., 2012b), noting that although linguistic features of documents can be used to explain eye-movements and reading behavior, there might be other influencing factors.

In our work, we consider the document and the eye-movements of the reader as the only *observable* variables in the reading act, and our general objective is to unveil the *hidden* variables intervening and influencing the interaction, such as writer's and reader's personal characteristics. Examples of writer's personal characteristics are writer's intention, concerns or emotional state, while reader's characteristics could be the reading objective, nationality, domain of expertise or literacy. We think that personal characteristics of readers define the mechanisms of their cognitive activity, and that when performing certain tasks, the eye-movements may reflect part of these personal characteristics. One of the hidden variables of the reader is the task performance achieved after reading a text, which is an intimate piece of information about the reader that could only be extracted so far by explicitly inquiring the subjects. Due to the interest and the broad range of applications that could benefit from the recognition of reading performance from eye-movements and document characteristics, we will narrow our study to this variable.

The objective of this work is then to use the spatiotemporal data that can be obtained by an eye-tracker when a subject reads a text and the linguistic information of the text itself, to capture common patterns in reading behavior across subjects with similar task performance. Thus, our hypothesis states that:

Hypothesis. *Subjects with a high performance in reading tasks have characteristic patterns of reading behavior and can be distinguished from the subjects with low performance.*

There are multiple ways of measuring task performance in reading tasks. One could argue that the reading objective defines how to measure performance. For example, the factors to measure success when reading a document with the objective of writing a review or preparing a presentation are clearly different. However, for the sake of simplifying and unifying our

method to measuring task performance, we will resort to measuring the level of understanding of subjects after reading a text.

In the next section, we describe the efforts from the psycholinguistic community in understanding the relationship between eye-movements and cognitive processing, and how the present work builds upon them. Section 3 introduces the models of reading behavior that are used to capture patterns in the eye-movements. A description of the data collection and experiment conditions can be found in Section 4 and the quantification of the likelihood of our hypothesis can be read in Section 5. In Section 6, we point at our next steps in recognizing reader's personal characteristics and suggest some applications that could benefit from the current line of investigation, followed by our conclusions at the end of the paper.

2 Related work

With the emergence of eye-tracking devices, the study on the relationship between eye-movements and cognitive processes experienced important advances. An interesting survey can be found in Rayner (1998) condensing the findings on characteristics of elemental eye-movements, lexical processing and integration of information during reading tasks. In McDonald and Shillcock (2003), it was shown that probabilistic language models implemented as bi-grams could help to explain the on-line cognitive processing of our brains and predict fixation times. That work suggested that statistical models can be used to model cognitive processing and that hypotheses could be tested using evidence extracted from the observation of eye-movements. Other models of eye-movements when reading were developed, being the E-Z reader (Reichle et al., 2003) one of the most comprehensive. Remarkably, a corpus of eye-movement data (Kennedy and Pynte, 2005) was also built to test previous computational models and paving the way to establish a common ground of computational model development. Following these ideas, we work under the assumption that eye-movements reflect cognitive processes and that the analysis of these eye-movements and the linguistic features of the text can be used to indirectly recognize the current mental state of the reader.

Cognitive load is an important variable that has received a significant amount of attention since it is a good signal of task difficulty and cognitive demand. In Tomanek et al. (2010), variations in eye-movements were used to build a cognitive cost model to predict human annotation costs of named entities, while in Doherty et al. (2010), variations in eye-movements were used to recognize hidden linguistic features from machine translation output such as sentence understandability. Although these two works appear to share the same idea, they point at different directions. The former uses linguistic features and eye-movements to unveil hidden cognitive costs, while the latter uses certainty on cognitive load and eye-movements to recognize textual characteristics such as sentence understandability. Both directions fall within our research interests, but in the present paper we follow the philosophy of Tomanek et al. (2010), in that we use observations of the text and the eye-movements to infer a hidden personal characteristic such as the level of understanding.

The authors in (Biedert et al., 2012) assume that eye-movements reflect difficulties in understanding the document being read and attempt to automatically recognize the quality of the text by integrating eye data from multiple readers. In that work, the authors investigate how different features of the eye-movements can reflect the quality of the text, but do not take into consideration the influence of lexical, syntactic and semantic complexity of the text on the eye-movements. This idea is central to our work, and we will attempt to combine data from eye-movements and linguistic features to obtain stronger predictors.

Eye-movements have also proved to characterize individuals, and there is a growing list of applications that would benefit from the refined user models that can be obtained by automatically processing eye-movements on text and other media (Xu et al., 2008; Buscher et al., 2008; Xu et al., 2009; Buscher and Dengel, 2009). With this motivation in mind, this paper contributes to better recognize personal characteristics of readers by using information from their eye-movements and the linguistic characteristics of the texts they read.

3 Methodology

3.1 Synthesizing images of linguistic and gaze evidence

With the objective of modelling the reading act, we first need to identify what variables are present in the process. The interaction between the reader and the document has latent and patent variables, where only the latter can be observed and measured. Nowadays, there is a large quantity of text that is being consumed in the form of digital content that is projected on a display, and it is reasonable to think that the structure, the contents and the linguistic features of the document can be observed and automatically extracted in the form of *statistical evidence*.

In the present paper, we work under the assumption that eye-movements can also be observed by means of an eye-tracker system. There are multiple eye-movements that have been recognized and categorized (see (Rayner, 1998) for details), but we can roughly group them into *fixations* and *saccades*. Fixations are periods of time where the subject looks still at a certain location, and it is known to be used for object recognition in general tasks or for lexical processing in reading tasks. Saccades, on the contrary, are sudden eye-movements that are used to change the fixation location and it is believed that cognitive processing is suppressed during the eye-movement.

There is a strong need of finding a suitable, unified representation of linguistic and gaze data in order to integrate these two sources of information. Gaze data can be naturally represented using synthesized images out of the (x, y) coordinates of the position where the eye-tracker "believes" the reader is gazing at. At a constant sampling rate, areas in the image where the reader gazed at longer will have a larger amount of pixels with high values, each pixel representing a gaze sample. There is, however, another aspect of the temporal data that can be obtained by an eye-tracker that is related to the sequential order of the gaze samples. The sequential order of the eye-movements is valuable to detect regressions, which are backward saccades usually used by readers for disambiguation, refreshing previously read passages or resolve apparent contradictions. Although these eye-movements are very interesting to infer the current state of mind of readers, they will not be included in our data representations due to their complex nature.

As it was pointed out by several authors(Hornof and Halverson, 2002; Hyrskykari, 2005), there are important variable and systematic errors in gaze location that need to be taken into account when working with eye-tracking systems. For the purpose of smoothing the effect that those errors may cause on our operations, we apply a gaussian convolution (Haralick and Shapiro, 1992) to blur the image representation of the gaze evidence and adjust the intensity of the pixel values to compensate for subjects that spent more or less amount of time on the document. The left plot in Figure 1 depicts a typical image representation of raw gaze data from an eye-tracking session of a subject reading a document, where fixations (little clusters of black dots) only occur on words or phrases (since those were the only objects of interest displayed on the screen), and text rows can be appreciated as fixations aligned horizontally. On the right plot, we depict the adjusted blurred version of the raw gaze data.

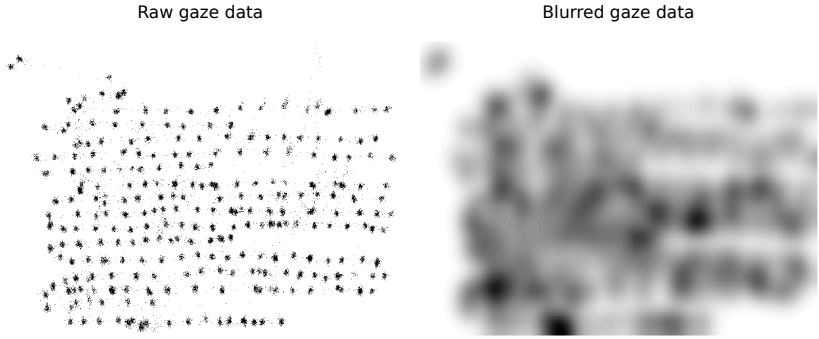


Figure 1: On the left, image representation of raw gaze data of an eye-tracking session. On the right, blurred image representation used to preserve uncertainty on the variable error introduced by the system. Pixel values are complemented for clarity.

Similarly, linguistic information can also be represented using synthesized images (Martínez-Gómez et al., 2012b), where the area of a word or phrase is filled with pixel values whose intensity is proportional to the quantification of a certain linguistic feature within the document. It should be noted at this stage that the image representations of linguistic features only quantify the presence of linguistic features within the document and does not take into account any gaze evidence. For this strategy of representation, there will be a synthesized image for each linguistic feature that we include in our model. The quantification of a linguistic feature can be normalized so that it falls within the $[0, 1]$ interval and the distribution of the pixel intensity values over that range can be adjusted in a similar manner as gaze images were, in order to compensate for linguistic features that occur too often or too rarely in the documents. An example of a binary feature that takes the values 1 or 0 to indicate whether a word is a noun or not can be seen on the left plot in Figure 2. The right plot in Figure 2 shows an image representation of a feature that quantifies the depth of a word in the parse tree of its corresponding sentence, normalized to fall within the interval $[0, 1]$.

In this work, we characterize reading behavior by the eye-movements of a subject when reading, which are in turn characterized by the distribution of fixations on parts of the text with certain linguistic features. It is thus useful to quantify this distribution of attention on each linguistic feature. Once we obtained the image representations of gaze evidence and the image representations of linguistic features, we combine these two sources of information by computing how well each linguistic feature explains the eye-movements. In line with our methodology, we will perform such combination by using image processing methods.

Within the image recognition field, image registration is the technique to find correspondences between two or more images with the purpose of estimating transformations for spatial alignment or detect temporal changes using pixel intensity differences. Although there is a wide variety of similarity measures that could be used to measure how well a certain linguistic feature explains the eye-movements, we opted for adapting precision, recall and F_1 scores as they are known in natural language processing, to the comparison of image representations.

Figure 2: On the left, image representation of a linguistic feature indicating what words are nouns (binary). On the right, image representation of a linguistic feature quantifying the word depth in a parse tree (normalized). Again, pixel values are complemented for clarity.

Computing the precision of how an image representation of a linguistic feature (source image) explains the image representation of gaze evidence (target image) consists in computing how much of pixel intensity in the source image is present in the target image, divided by the total amount of intensity across all pixels from the source image. Similarly, recall can be computed as the amount of pixel intensity from the source image that is present in the target image, divided by the amount of intensity across all pixels from the target image. Finally, the familiar expression $F_1 = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}$ can be used to compute the F_1 score, which is a value in the interval $[0, 1]$. An example of the computation of precision, recall and F_1 scores between images of only four pixels can be found in Figure 3.

3.2 Representation of observations

A reading session consists of a subject reading a document. From every reading session, we can obtain data from the eye-movements and data about the linguistic features of the text. As we described in Section 3.1, we can synthesize an image representation of the data from the eye-movements, and an image representation for each linguistic feature. Using the F_1 score, we computed how well the image representation of linguistic feature i match (or explains) the image representation of the eye-movements, and obtained a number $o_i \in \mathscr{R}$ such that $0 \leq o_i \leq 1$. If we match the image representation of every linguistic feature against the image representation of the eye-movements, we can obtain a fixed-size feature vector $\mathbf{o} = [o_1, \ldots, o_L]$ that has as many components (or dimensions) as linguistic features (L). An overview of the architecture can be found in Figure 4.

Thus in our model, the reading behavior of a subject reading a document is defined as $\mathbf{o}$, which represents the distribution of attention (eye-movements) on every linguistic feature. Using fixed-size feature vectors to represent observations is a widely used technique that, although it has a limited expressivity, it allows to use very well known efficient analysis and inference techniques. Using this formalism, we will say that two subjects have similar reading behavior

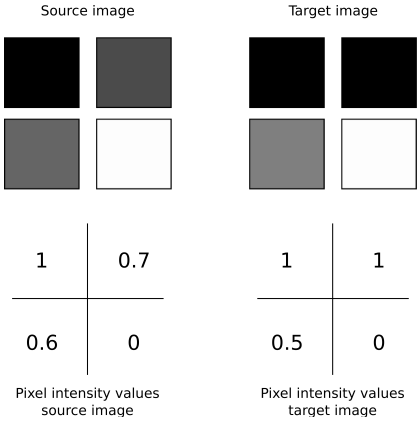


Figure 3: Example of computation of precision, recall and F_1 score between a source and a target image. On the top row, four pixels (complemented for clarity) are displayed, and their corresponding pixel values on the bottom row. Precision $= \frac{1+0.7+0.5}{1+0.7+0.6} = 0.96$. Recall $= \frac{1+0.7+0.5}{1+1+0.5} = 0.88$. Thus, $F_1 = 0.92$.

if they have similar feature vectors, as defined by a similarity metric in the feature space. For the sake of readability and ease of interpretation in posterior sections, we chose the euclidean distance between two observations $\mathbf{o}$ and $\mathbf{o}'$ as a similarity measure, defined as:

$$\text{dist}(\mathbf{o}, \mathbf{o}') = \sqrt{(f_1 - f_1')^2 + \cdots + (f_F - f_F')^2} \tag{1}$$

where the smaller the distance, the higher the similarity between $\mathbf{o}$ and $\mathbf{o}'$ is.

As it has been said, each reading session is represented by a feature vector $\mathbf{o}$ that defines a distribution on how well each linguistic feature helps to explain the gaze evidence from a certain subject. Our hypothesis is that this representation helps to discriminate between subjects with a low and a high task performance. In order to test this hypothesis, we will test for *linear separability*, that is, how well we can separate subjects with a hypersurface[1] in a certain reduced dimensionality.

4 Experimental framework

In order to collect data for experimentation purposes, 9 subjects were asked to read news and fiction documents in English following different reading strategies. Subjects were students from China (1), Indonesia (1), Japan (4), Spain (1), Sweden (1) and Vietnam (1) in Bachelor, Master, PhD and post-doctoral levels of education in computer science.

Three reading strategies were considered. The first strategy was *precise reading*, where subjects were told to read two documents with the objective of maximizing their comprehension and that their level of understanding would be tested after reading each document with yes-no questions,

[1] A straight line and a plane in 2 and 3 dimensions, respectively.

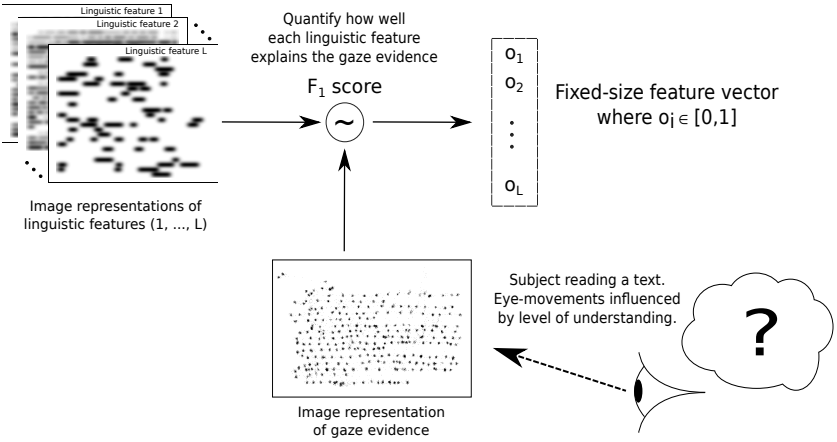


Figure 4: Schema of the methodology used in this work to estimate the distribution on how each linguistic feature explains subject's reading behavior. First, a collection of image representations of linguistic features is synthesized. Then, using the image F_1 score function, each linguistic feature is matched against the gaze data to obtain a measure on how well it explains reading behavior. The result is a fixed-size feature vector $\mathbf{o}$ representing a reading session.

multiple-choice questions and free-answer questions. The second strategy was *skimming*, with the objective of finding the answer to one or two questions (to each document, respectively) and the accuracy and completeness of their answer was measured after subjects claimed that they had found the necessary information to answer the questions. The third strategy was named *10-second reading*, where subjects were given 10 seconds to obtain as much information as possible from two documents (respectively), and were asked to show the amount of information they got and were scored proportionally to correctness and completeness. Table 1 summarizes basic statistics of the documents. The total duration of the reading sessions and question answering to obtain subject's level of understanding was, on average, 40 minutes.

Randomizing the presentation order of the documents is a standard practice in psycholinguistics. In the present work, however, the presentation order was kept constant, due to concerns about the effects of the randomization when working with a limited number of subjects. We believe that the results are not affected by this decision, since subjects were compared within the same document and reading strategy (thus, under the same experimental conditions).

In what follows, we assume that the score that subjects obtained in the questionnaires after reading every document represents the subject's level of understanding and that the highest score among all subjects represents a 100% of understanding. There were two rules used to partition subjects and test linear separability. In the first partition rule, we select the subjects whose understanding was below 33%, and try to find a linear decision boundary (in the form of a straight line or plane) that separates those subjects from the rest of the participants. In the second partition rule, we select the subjects whose understanding is above 66%, and proceed to test whether they can be linearly distinguished from the rest.

Reading strategy	Document	Avg. tokens / sent.	Num. sent.	Avg. reading time
10-second reading	doc. 1	23.4	11	10 s.
	doc. 2	22.6	14	10 s.
Skimming	doc. 3	20.9	13	70 ± 64 s.
	doc. 4	11.8	20	52 ± 38 s.
Precise reading	doc. 5	30.2	11	116 ± 49 s.
	doc. 6	20.3	12	95 ± 36 s.

Table 1: Statistics on the average number of tokens per sentence (Avg. tokens / sent.), number of sentences (Num. sent.) and average reading time (Avg. reading time). Statistics were collected across all subjects for the 6 documents used during the experiments, to record eye-movements and obtain the linguistic features of those documents.

With the intention of capturing all possible linguistic influence on reading behavior, we collected a set of lexical, syntactic and semantic linguistic features on every document. Among the lexical linguistic features, we measured word length, whether the word contains a digit or not, the presence of upper case letters or word unpredictability, as given by the perplexity from a 5-gram language model trained on a big corpus (Koehn, 2005) and smoothed using modified Kneser-Ney technique (Chen and Goodman, 1999). Syntactic features were also included in our model, such as whether a word is the head of a phrase, binary features indicating whether a word has a certain Part-of-Speech (POS) tag, total height of the parse tree of the sentence each word corresponds to, word position in the sentence, etc. We believe that semantic features also may influence greatly on the eye movements, since it is reasonable to think that an important part of the cognitive processing consists in an incremental integration of the information into the personal knowledge base, right after the lexical and syntactic processing happens. However, due to the difficulty in formally defining and properly quantifying semantic features, only two were considered, namely the word ambiguity, as given by the number of senses in WordNet (Miller, 1995) that a word may have, and a feature indicating whether a word or phrase is a named entity or not. The complete list of linguistic features included in our model can be found in Table 2, and will be used to synthesize image representations and compute how well they help to explain the gaze evidence.

Prior recording the eye-movements during the reading session, every subject was informed of the dynamics of the reading tasks. Tobii TX300 and Text2.0(Biedert et al., 2010) were used to capture the (x, y) coordinates of the gaze samples for every subject reading every document, and the eye-tracker was calibrated before every subject read every document. In order to avoid introducing tracking errors as much as possible, a chin rest was used for subjects to keep their heads stable. A text-gaze aligner (Martínez-Gómez et al., 2012a) based on an image registration method was also used to correct variable and systematic errors in the coordinates of the gaze samples, and further corrections were manually performed when necessary. Finally, the eye-tracking session data of a subject reading one of the documents in the 10-second reading task was discarded due to unrecoverable errors during the eye-tracking session.

Pixel intensity values were normalized in the image representations of the eye-movements and the image representations of the linguistic features, in order to compensate for subjects

Category	Linguistic feature	Type
Lexical	word length	Integer
	contains digit	Binary
	word unpredictability	Real
	contains uppercase	Binary
	is all uppercase	Binary
Syntactic	is head	Binary
	is POS `$tag` (23 features)	Binary
	height of parse tree of its sentence	Integer
	depth of the word in the parse tree	Integer
	word position in sentence	Integer
Semantic	is named entity	Binary
	ambiguity: number of senses from WordNet	Integer

Table 2: Lexical, syntactic and semantic linguistic features considered in this work. Examples of POS `$tag` are "Nouns", "Verbs" and "Prepositions". Heads and parse trees were obtained using an HPSG parser (Miyao and Tsujii, 2008).

spending different amount of time on the documents, and to account for rare or too-frequent linguistic features that have a different amount of total pixel intensity in the image. Thus, the intensity was adjusted in such a way that 1% of the highest and the lowest non-zero pixel values were saturated (i.e. totally dark and totally white).

5 Results

5.1 Patterns in the variance

In our hypothesis, we stated that subjects with different task performance would have different patterns in reading behavior. With the objective of revealing these differences at a preliminary stage, we proceed to analyze the variability of the distribution of attention on the linguistic features across all subjects participating in our experiments. Studying the variability of data is of special interest because patterns in variance are usually good signals to be considered to discriminate between subjects. Thus, Principal Component Analysis (PCA) (Jolliffe, 2002) will be used to obtain the directions in the feature space where the covariance matrix of the observations varies the most, as given by the eigen-vectors associated to the highest absolute eigen-values.

Then, we can project the observations (our subjects) onto the two or three directions that capture most of the variance. On the left plot of Figure 5, we can observe a projection onto the two dimensions with highest variability, capturing 74.8% of the total variance across all subjects in document 1 of the 10-second reading task. It can be appreciated that the three subjects with the lowest level of understanding (marked as x) are located on the left part of the plot, clearly separated from the rest of the subjects. On the right plot of Figure 5, subjects were projected on the three directions that capture most of the variability for the same document, capturing 85.6% of the total variance.

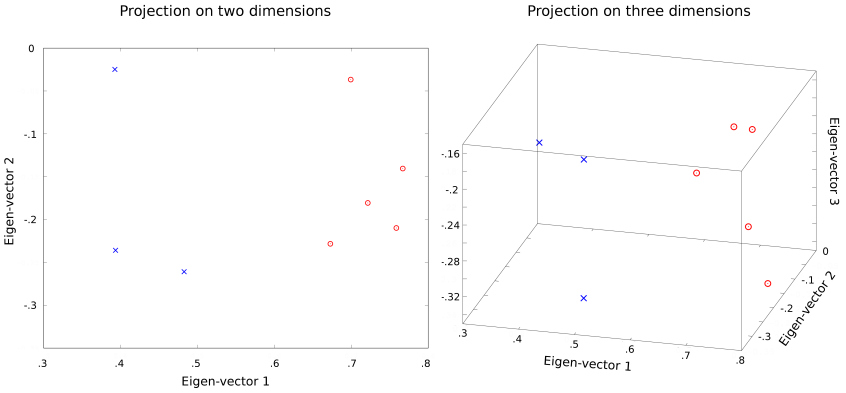


Figure 5: Principal Component Analysis on the covariance matrix of the subjects reading document 1 during the 10-second reading task. Subjects with low level of understanding are marked as x, and subjects with a high understanding are marked as o. On the left plot, a projection onto two dimensions capture 74.8% of the variance; on the right plot, a projection onto three dimensions capture 85.6%.

5.2 Separability

In order to reveal the differences in reading patterns (as described in Section 3.2), we will test for linear separability of subjects in a low dimensional space resulting from a principal component projection of the subjects. Despite of the apparently reduced number of subjects participating in the experimentation, linear separability of a number of subjects (i.e. 8 or 9) in a space of much smaller dimensionality (i.e. 2 or 3) is not likely to happen by pure chance.

The quantification of the linear separability can be found in Table 3. Subjects (observations **o**) were projected onto two and three dimensions using their two and three directions of highest variability respectively, following the PCA dimensionality reduction method. For every projection and every document, two partition rules (as described in Section 4) were considered to select the subset of subjects for which linear separability have to be tested. Then, a decision line (in two dimensions) or a plane (in three dimensions) was obtained such that maximizes the number of subjects from the subset of interest that are correctly separated from the rest of the subjects, without allowing subjects that do not belong to the subset of interest to be miss-classified. Every cell in Table 3 shows a fraction X/Y, where X denotes the number of subjects from the subset of interest (i.e. subjects with the lowest or the highest level of understanding) that were correctly separated from the rest of the subjects by using the best possible linear separation, and Y denotes the total number of subjects in the subset of interest.

In an instance of a positive example from Table 3, there were 8 subjects reading document 1 in 10-second reading task and those subjects were projected onto two dimensions. Using partition rule 1, the 3 subjects with the lowest level of understanding were selected, and all of them were correctly linearly separated. A negative example can be found at the 2-dimensional projection of the 9 subjects reading document 3 of the skimming task, where partition rule 1 is used to

Reading strategy	Document	Partition rule 1		Partition rule 2		Num. subjects
		2 Dim.	3 Dim.	2 Dim.	3 Dim.	
10-second reading	doc. 1	**3/3**	**3/3**	**2/2**	**2/2**	8
	doc. 2	**1/1**	**1/1**	0/1	0/1	9
Skimming	doc. 3	0/2	**2/2**	5/7	6/7	9
	doc. 4	**1/1**	**1/1**	**8/8**	**8/8**	9
Precise reading	doc. 5	1/4	2/4	**1/1**	**1/1**	9
	doc. 6	**1/1**	**1/1**	**1/1**	**1/1**	9

Table 3: Quantification of the separability for subjects reading two documents following 10-second, skimming and precise reading tasks. Subjects have been projected in two and three dimensions (Dim.), and linear separability have been tested for two different partition rules as described in Section 4. At every cell, X/Y denotes that Y subjects were selected by the partition rule to test their linear separability, and X of them were successfully separated from the rest of the subjects.

select the subset of subjects with the lowest level of understanding. From Table 3, it can be read that not a single subject with a low level of understanding can be linearly separated from the rest, thus not being distinguishable from the other subjects by using the patterns on reading behavior that are described in this paper. An illustration of these two examples can be found in Figure 6.

It can be observed that the subjects with a level of understanding below one third of the highest level of understanding can be linearly separated from the rest of the subjects in all documents of the 10-second and skimming reading strategies. This linear separation was feasible in two and three dimensions respectively[2]. At the precise reading strategy, only some subjects with the lowest level of understanding were found to be linearly separable from the rest of the subjects, indicating that the distribution of the attention over linguistic features only contains limited information about reading performance and that the eye-movements might be influenced by factors of different nature, which is consistent with (Martínez-Gómez et al., 2012b).

Table 3 also contains a quantification of how well subjects with a high level of understanding could be linearly differentiated from the rest of the subjects (columns corresponding to partition rule 2). Although it can be appreciated that results are not consistent across all reading tasks, there is a positive trend of linear separability. In document 1 of the 10-second reading task, two subjects were selected by the partition rule 2, and both of them were successfully linearly separated from the rest of the subjects in 2 dimensions. In document 2, however, the partition rule only selected one subject with the highest level of understanding, but neither projections in 2 nor 3 dimensions allowed for linear separability. In document 3 of the skimming task, 7 subjects (out of 9) were selected as having the highest level of understanding, and 5 and 6 of them were successfully linearly separated in 2 and 3 dimensions respectively. In document 4 of the same task, the partition rule 2 selected 8 subjects, and all of them were linearly separated from the remaining subject. Finally, in both documents of the precise reading task, only 1 subject in every document was selected as having the highest level of understanding, and the subject was positively linearly separated from the rest of the subjects in 2 and 3 dimensions.

[2]Note that if a subset is linearly separable in n dimensions, then it is also linearly separable in $n+1$ or more dimensions.

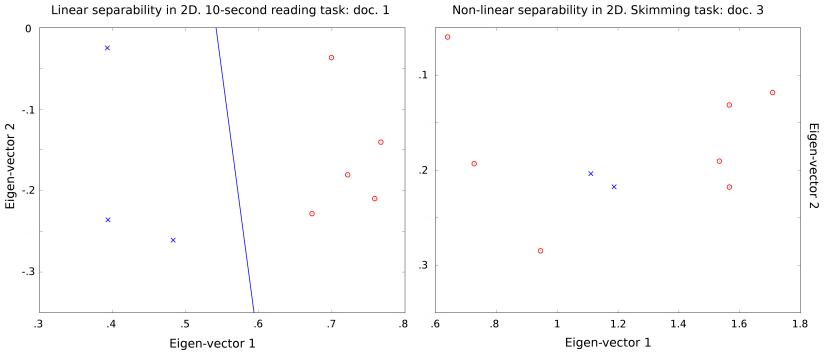


Figure 6: On the left, a positive example of linear separability in a projection on two dimensions of the subjects reading document 1 in the 10-second reading task. On the right, a negative example of a projection on two dimensions of the subjects reading document 3 of the skimming task, where subjects with low level of understanding (x) cannot be linearly differentiated from the rest of the subjects (o), using the patterns in reading behavior described in this work.

6 Future work and applications

The work presented in this paper is a step towards recognizing personal characteristics by using data extracted from the eye-movements in combination with the linguistic properties of the documents being read. We believe that there are other personal characteristics that can be extracted from eye-movements. Language ability is one of them, and it could be recognized by analyzing the proportions in the quantity of attention paid on words or phrases with certain linguistic features such as verbs or nouns. Subject's domain of expertise could also be recognized by analyzing fixation times on technical terms and comparing it to other subjects with different known domains of expertise. We also believe that the reading intention of users is also reflected on the eye-movements and it falls within our research road map. We are also interested in designing models to explain how the pupil size diameter depends on the linguistic characteristics and the amount of information contained in a text with the objective of quantifying the amount of surprise that readers received when exposed to the documents.

There are multiple applications that could benefit from refined user models that account for these personal characteristics such as user intention or domain of expertise. Information recommendation systems are a clear example, since recognizing users' intention is the first step in satisfying their information needs. Assistive technologies for reading and writing could also be developed since past records of users reading the same document would give hints on the text areas that require the highest amount of attention to maximize the understanding of the content, and writers could also use that information to optimize their documents for an efficient or pleasant reading experience. Applications for diagnosing learning difficulties in children and young people could also be developed following a similar strategy to the work it has been presented in this paper.

7 Conclusions

In the present work, we have introduced a method to represent subjects as fixed-size feature vectors that denote the distribution on how well each linguistic feature explains the eye-movements when reading a specific document. Information on gaze samples and linguistic features was integrated in a common framework by means of their encoding into synthesized images whose pixels quantify the strength of the statistical evidence. With the definition of image precision, recall and F_1 scores, we narrowed the gap between the image and natural language processing fields. Although traditional statistical models could be used with similar results, our method allows to include geometric information into our linguistic models in a natural manner.

Examples of image processing techniques that resulted useful were the image registration to perform text-gaze alignment, blurring images to carry the uncertainty of the error-correction into our subsequent models, capabilities to adjust intensity of pixel values to compensate for too common or too rare linguistic features, ease in visually analyzing our reading models and more importantly, estimating how well each linguistic feature explains eye-movements data without the need of testing for significant decreases in perplexity when those features are added into the traditional statistical models.

We analyzed the variability on the distribution over the personal feature vectors by projecting them onto a lower dimensional space for visual inspection. We observed patterns in the distribution of those feature vectors across all subjects, found that they are characteristic of every subject and that they relate to the subject's level of understanding. Finally, we tested the hypothesis that subjects with different levels of understanding can be distinguished from each other by using the information extracted from the combination of eye-movements captured by an eye-tracker and the linguistic information extracted from a document.

In order to test our hypothesis given the limited number of subjects, we used "linear separability", which is a very exigent condition to satisfy conditioned on the low dimensionality and the number of subjects that we presented. We consistently succeeded in linearly discriminating subjects with low level of understanding from the rest of the subjects at the 10-second and skimming reading tasks. However, subjects with low task performance were not consistently linearly separable for the task of precise reading, suggesting that other methods might be necessary to discriminate them. Linear separability of subjects with high level of understanding showed a positive but not decisive trend and we will say, for now, that subjects with lower level of understanding are easier to recognize by their eye-movements on the documents they read, when compared to subjects with high task performance. The ability to distinguish readers according to their level of understanding can further be accomplished in a less exigent scenario by relaxing the condition of separability into a higher dimensionality or non-linearity with a larger number of subjects.

Until now, recognizing the level of understanding of a subject when reading a document was only possible by requiring explicit feedback from subjects. The findings in this work demonstrate that the cognitive activity associated to a low or high level of understanding influences subject's eye-movements and that those eye-movements can be characterized in some readings tasks with the help of the linguistic characteristics of the text being read. Although the experiments presented in this paper are still limited, we have evidenced strong patterns in eye-movements that will allow us to unveil a larger portion of a person's state of mind.

References

Biedert, R., Buscher, G., Schwarz, S., Möller, M., Dengel, A., and Lottermann, T. (2010). The text 2.0 framework - writing web-based gaze-controlled realtime applications quickly and easily. In *Proceedings of the International Workshop on Eye Gaze in Intelligent Human Machine Interaction (EGIHMI)*.

Biedert, R., Dengel, A., Elshamy, M., and Buscher, G. (2012). Towards robust gaze-based objective quality measures for text. In *Proceedings of the Symposium on Eye Tracking Research and Applications*, pages 201–204. ACM.

Buscher, G. and Dengel, A. (2009). Gaze-based filtering of relevant document segments. In *International World Wide Web Conference (WWW)*.

Buscher, G., Dengel, A., and van Elst, L. (2008). Query expansion using gaze-based feedback on the subdocument level. In *ACM Special Interest Group on Information Retrieval (SIGIR)*.

Chen, S. F. and Goodman, J. (1999). An empirical study of smoothing techniques for language modeling. *Computer Speech and Language*, 4(13):359–393.

Doherty, S., O'Brien, S., and Carl, M. (2010). Eye tracking as an MT evaluation technique. *Machine Translation*, 24:1–13. 10.1007/s10590-010-9070-9.

Haralick, R. and Shapiro, L. (1992). *Computer and robot vision*, volume 1. Addison-Wesley.

Hornof, A. and Halverson, T. (2002). Cleaning up systematic error in eye-tracking data by using required fixation locations. *Behavior Research Methods*, 34:592–604. 10.3758/BF03195487.

Hyrskykari, A. (2005). Utilizing eye movements: Overcoming inaccuracy while tracking the focus of attention during reading. *Computers in Human Behavior*, 22:657–671.

Jolliffe, I. (2002). *Principal component analysis*, volume 2. Wiley Online Library.

Kennedy, A. and Pynte, J. (2005). Parafoveal-on-foveal effects in normal reading. *Vision Research*, 45:153–168.

Koehn, P. (2005). Europarl: A parallel corpus for statistical machine translation. In *Proceedings of the 10th Machine Translation Summit, 2005*, pages 79–86.

Martínez-Gómez, P., Chen, C., Hara, T., Kano, Y., and Aizawa, A. (2012a). Image registration for text-gaze alignment. In *Proceedings of the 2012 ACM international conference on Intelligent User Interfaces*, IUI '12, pages 257–260, New York, NY, USA. ACM.

Martínez-Gómez, P., Hara, T., Chen, C., Kyohei, T., Kano, Y., and Aizawa, A. (2012b). Synthesizing image representations of linguistic and topological features for predicting areas of attention. In *Proceedings of The Pacific Rim International Conference on Artificial Intelligence*, PRICAI '12.

McDonald, S. A. and Shillcock, R. C. (2003). Eye movements reveal the on-line computation of lexical probabilities during reading. *Psychological Science*, 14(6):648–652.

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

Miyao, Y. and Tsujii, J. (2008). Feature forest models for probabilistic HPSG parsing. *Computational Linguistics*, 34:35–80.

Rayner, K. (1998). Eye movements in reading and information processing: 20 years of research. *Psychological Bulletin*, 124:372–422.

Reichle, E., Rayner, K., and Pollatsek, A. (2003). The E-Z reader model of eye-movement control in reading: Comparisons to other models. *Behavioral and brain sciences*, 26:445–526.

Tomanek, K., Hahn, U., Lohmann, S., and Ziegler, J. (2010). A cognitive cost model of annotations based on eye-tracking data. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, ACL '10, pages 1158–1167, Stroudsburg, PA, USA. Association for Computational Linguistics.

Xu, S., Jiang, H., and Lau, F. C. (2008). Personalized online document, image and video recommendation via commodity eye-tracking. In *ACM Recommender Systems (RecSys)*.

Xu, S., Jiang, H., and Lau, F. C. (2009). User-oriented document summarization through vision-based eye-tracking. In *International Conference on Intelligent User Interfaces (IUI)*.

To Exhibit is not to Loiter: A Multilingual, Sense-Disambiguated Wiktionary for Measuring Verb Similarity

Christian M. Meyer[1] *Iryna Gurevych*[1,2]
(1) Ubiquitous Knowledge Processing Lab (UKP-TUDA)
Department of Computer Science, Technische Universität Darmstadt
(2) Ubiquitous Knowledge Processing Lab (UKP-DIPF)
German Institute for Educational Research and Educational Information
http://www.ukp.tu-darmstadt.de

ABSTRACT
We construct a new multilingual lexical resource from Wiktionary by disambiguating semantic relations and translations. For this task, we propose and evaluate an automatic disambiguation method that outperforms previous approaches significantly. We additionally introduce a method for inferring new semantic relations based on the disambiguated translations. Our resource fills the gap between expert-built resources suffering from high cost and small size and Wikipedia-based resources that are restricted to encyclopedic knowledge about nouns. We demonstrate this by applying our new resource to measuring monolingual and cross-lingual verb similarity. For the latter, our resource yields better results than Wikipedia and expert-built multilingual wordnets. We make our final resource and the evaluation datasets publicly available.

TITLE AND ABSTRACT IN GERMAN

Ein mehrsprachiges, lesartendisambiguiertes Wiktionary zur Bestimmung von Verbähnlichkeiten

Der vorliegende Beitrag beschreibt die Gewinnung einer neuen, mehrsprachigen lexikalischen Ressource aus Wiktionary-Daten, die durch Disambiguierung von semantischen Relationen und Übersetzungen entsteht. Zu diesem Zweck definieren und evaluieren wir eine automatische Methode zur Lesartendisambiguierung, die frühere Ansätze signifikant übertrifft. Wir stellen ferner eine Methode vor, um neue semantische Relationen auf Basis der disambiguierten Übersetzungen zu inferieren. Unsere Ressource schließt die Lücke zwischen von Experten erstellten Wissensquellen, die unter ihrer oft geringen Größe aber hohen Erstellungskosten leiden, und Wikipedia-basierten Ressourcen, die nahezu ausschließlich enzyklopädisches Wissen zu Substantiven enthalten. Beim Einsatz unserer neuen Ressource zur Bestimmung von einsprachigen und zweisprachigen Verbähnlichkeiten erreichen wir im letzteren Fall bessere Ergebnisse als für Wikipedia und die Expertenressourcen. Wir veröffentlichen unsere Ressource und die Evaluierungsdatensätze für zukünftige Forschungsarbeiten.

KEYWORDS: Wiktionary, Lexical Resource, Semantic Relation, Translation, Word Sense Disambiguation, Verb Similarity.

KEYWORDS IN GERMAN: Wiktionary, Lexikalische Ressource, Semantische Relation, Übersetzung, Lesartendisambiguierung, Verbähnlichkeit.

Proceedings of COLING 2012: Technical Papers, pages 1763–1780,
COLING 2012, Mumbai, December 2012.

1 Introduction

Motivation. The advancing globalization and the permeation of the internet in our daily lives raises a strong demand for multilingual applications, such as machine translation, cross-lingual question answering, or information retrieval. Traditional multilingual approaches are knowledge-based using bilingual dictionaries (Neff and McCord, 1990) or multilingual wordnets (Tufiş et al., 2004). To date, these approaches are getting more and more replaced by statistical translation models, although it has been found that multilingual resources have the ability to substantially contribute to the performance of a system (Oepen et al., 2007; Herbert et al., 2011). One reason for the knowledge-based approaches being rarely employed is the challenging construction process of multilingual resources. They are either manually compiled by professional translators or lexicographers or automatically generated from large amounts of unstructured data. The former usually results in small resources due to the time and cost intensive work, whereas the latter often reaches only a limited quality. Although Wikipedia has been found as a promising alternative for obtaining multilingual knowledge (Medelyan et al., 2009), it is almost entirely restricted to nouns and focuses on encyclopedic rather than lexical-semantic knowledge.

Contribution. In this paper, we will explore the collaborative online lexicon Wiktionary[1] and how it can be used as a multilingual resource. Similar to Wikipedia, the contents in Wiktionary are edited by a large community of Web users. This collaborative construction approach, known as the "Wisdom of Crowds", yields very large resources. At the same time, this assures a considerable quality, as the numerous authors can quickly revise erroneous or unclear entries. Wiktionary offers a broad range of lexical-semantic knowledge including sense definitions, semantic relations, and translations. It fills the gap between the small, expert-built wordnets and the large Wikipedia-based resources restricted to nouns.

The contribution of our paper is threefold: (i) We propose and evaluate a method for disambiguating semantic relations and translations in Wiktionary; (ii) we infer new semantic relations based on the disambiguation result and create a novel sense-disambiguated Wiktionary that we make freely available; (iii) we demonstrate the usefulness of our new sense-disambiguated resource by employing it for calculating cross-lingual verb similarity. Measuring verb similarity is often a crucial technique for information extraction or (cross-lingual) question answering systems. In this paper, we experiment with English and German even though our methods can generally be adapted to over 170 languages covered by Wiktionary.

Overview. The English Wiktionary consists of about 475,000, the German Wiktionary of about 73,000 *word senses*.[2] For each of these word senses, multiple *semantic relations* (i.e., synonymy, antonymy, hyponymy, etc.) and *translations* may be encoded. We will use *relation* henceforth to refer to both semantic relations and translations and use the terms *source* and *target* to denote the endpoints of a relation. The Wiktionary entry for *(to) hang* distinguishes, for instance, fifteen word senses. The eighth word sense is defined as *"to exhibit (an object)"* with synonymy relations targeting at *exhibit* and *show* and translations into German *ausstellen*, French *exposer*, Dutch *ophangen*, and other languages.

The target of a relation is encoded using word forms. Thus, it remains underspecified which word sense a relation is pointing to. The synonym *exhibit* of the eighth word sense of *hang* can, for example, refer to the meaning of displaying something (e.g., exhibiting a drawing) or

[1] `http://www.wiktionary.org`

[2] All statistics are based on Wiktionary data of April 2011 accessed using JWKTL (Zesch et al., 2008a).

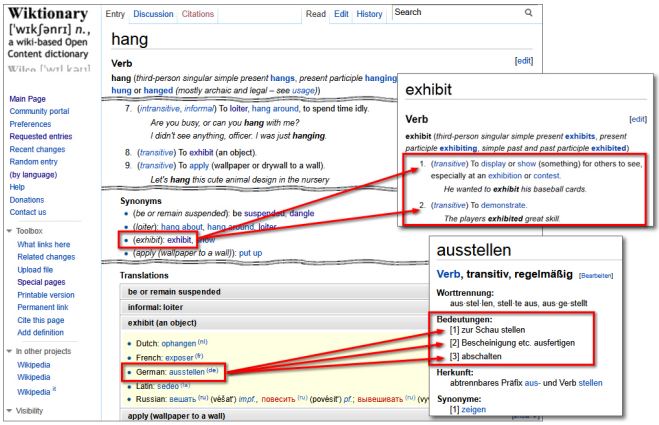


Figure 1: The synonym *(to) exhibit* of the English Wiktionary entry *(to) hang* and its German translation *ausstellen* have multiple possible target word senses.

demonstrating a skill (e.g., exhibiting a talent in acting). For humans, it is easy to recognize that *hang* is synonymous to the first word sense of *exhibit*, but not to the second. Natural language processing applications, however, cannot disambiguate such relations easily. The same applies to translations: The German *ausstellen* has, for instance, a meaning of (1) exhibiting an object, (2) certificating a document and (3) turning off smth. Figure 1 illustrates this kind of underspecification. In Section 3, we propose a solution to this issue by automatically disambiguating the semantic relations and translations in Wiktionary. Sense-disambiguated relations are a necessary precondition for many applications, such as computing semantic relatedness by measuring path lengths (Budanitsky and Hirst, 2006): if undisambiguated relations were used, then *exhibit* and *loiter* would be highly related as they both have a relation to *hang*.

Besides the information inherently found in Wiktionary, we infer new semantic relations based on our disambiguated translations. This is particularly useful for the English Wiktionary, which encodes only about 26,000 semantic relations (compared to 290,000 in the German edition). With our inference method, we are able to increase the number of semantic relations for the English language by almost ten times. Section 4 describes our inference method and provides statistics of our new resource. In Section 5, we apply this new resource to calculating monolingual and cross-lingual verb similarity as one example use case in the scope of our work. Thereby, we show that our resource is comparable to expert-built resources in the monolingual experiments and that it outperforms them in the cross-lingual setting by a large margin.

2 Related Work

The most closely related areas of work are the construction of multilingual resources, the disambiguation of relations and the inference of new semantic relations.

Multilingual resource construction. The most prominent multilingual resources are EuroWordNet (Vossen, 1998), BalkaNet (Stamou et al., 2002), and MultiWordNet (Pianta et al.,

2002). All of them are professionally crafted and provide a well-structured network of word senses and relations. Despite their high quality, the sizes vary largely. For English and German, there are, for instance, only 16,347 shared word senses in EuroWordNet (the only one encoding this language pair).[3] Another drawback of these wordnets is their high development cost which hinders the large-scale manual extension of their contents. Wikipedia-based multilingual resources is another strand of research. Well-known works are Yago (Suchanek et al., 2007), DBpedia (Bizer et al., 2009), and WikiNet (Nastase et al., 2010), which mostly differ in their structure and the way they extract the data. The bulk of knowledge in Wikipedia is, however, of encyclopedic nature, whereas our work aims at lexical-semantic knowledge.

The two most closely related research efforts to ours are Universal WordNet (de Melo and Weikum, 2009) and BabelNet (Navigli and Ponzetto, 2010). The former uses WordNet for bootstrapping a multilingual resource based on combined evidence found in existing wordnets, parallel corpora, and machine-readable dictionaries. It incorporates (undisambiguated) Wiktionary translations, but solely relies on semantic relations taken from WordNet. BabelNet aligns WordNet and Wikipedia at the level of word senses. Although this yields a large resource, the additional information from Wikipedia is almost entirely about nouns – there are hence no translations for verbs, adjectives, or the like. Our work provides a viable option towards closing this gap, as it makes use of lexical-semantic knowledge covering any part of speech.

Relation disambiguation. The task of disambiguating semantic relations (also called *sense linking* and *relation anchoring*) has been previously described in the context of machine-readable dictionaries (Krovetz, 1992) and ontology learning (Pantel and Pennacchiotti, 2008). Meyer and Gurevych (2010) discussed relation disambiguation for the German Wiktionary using a disambiguation method based on textual similarity. In Section 3.3, we will compare this approach to our system.

The disambiguation of all words in a sense definition (i.e. *gloss disambiguation*), as it has been done in the WordNet 2/eXtendend WordNet project (Harabagiu et al., 1999; Mihalcea and Moldovan, 2001), is very similar to the disambiguation of semantic relations. Therefore, many of the features defined in Section 3.1 are similar to those proposed by Moldovan and Novischi (2004). Note however that we use explicitly defined semantic relations rather than sense definitions as our disambiguation subjects. In addition to that, we adapt our method to Wiktionary instead of using WordNet-specific features and also extend this work to a cross-lingual setting. Very recently, Flati and Navigli (2012) proposed a graph-based method to gloss disambiguation outperforming previous approaches. While this method could in general be adapted to disambiguating relations, we observe that the graph induced by Wiktionary's semantic relations is very sparse. This would hinder finding the cycles and quasi-cycles required by the method.

The disambiguation of translations has been studied in the context of bilingual dictionaries and corpora (Kikui, 1999; Tsunakawa and Kaji, 2010). Mausam et al. (2009) discovered new translations in Wiktionary using a graph-based inference algorithm for Wiktionary translations. Although this also involves a disambiguation of translations, their work is not directly comparable to ours, since they do not strictly use the word senses encoded in Wiktionary but define them based on the translations shared across multiple languages. In contrast to that, we aim at exploiting a wide range of lexical-semantic knowledge and therefore need to rely on the word senses actually encoded in Wiktionary.

[3] `http://www.illc.uva.nl/EuroWordNet/finalresults-ewn.html` (accessed 2012-07-11)

Inference of relations. New semantic relations have been previously inferred when bootstrapping wordnets, i.e. translating the word senses and their definitions to a new language and reusing the relations from an existing wordnet. This has been done, for example, for constructing the Spanish (Atserias and Villarejo, 2004), French (Sagot and Fišer, 2008), and Thai (Thoongsup et al., 2009) wordnets. Such approaches differ from our work in that they do not require a disambiguation of relations. Huang et al. (2002) studied the cross-lingual inference of semantic relations when using imprecise translations. They measure an error rate of 11% for the inference of Chinese semantic relations based on the English WordNet.

3 Disambiguation of Wiktionary's Semantic Relations and Translations

In this section, we describe and evaluate our method for automatically disambiguating semantic relations and translations in Wiktionary.

3.1 Feature Definition

Let $t_j \in t$ be one of multiple possible target word senses for a relation (either a semantic relation or a translation) $r = (s_i, t)$. We define the following features based on our analysis of 200 Wiktionary relations (referred to as *development data*).

Definition overlap. A widely used method for word sense disambiguation is based on counting word overlaps between sense definitions (Lesk, 1986). Let $\text{gloss}(s_i)$ and $\text{gloss}(t_j)$ be the lemmatized and stop-word-filtered sense definitions of s_i and t_j. Their overlap is the number of shared words:

$$f_{\text{Lesk}} := |\text{gloss}(s_i) \cap \text{gloss}(t_j)|.$$

We additionally define f_{ExtLesk} by employing the extension by Banerjee and Pedersen (2003), i.e. we assign squared scores to consecutive sequences of words. If both definitions contain, for example, *"large carnivorous animal"*, we assign a score of $3^2 = 9$.

Source lemma. A special case of overlapping definitions is that the lemma of the source word sense is contained in the definition of the target word sense:

$$f_{\text{src}} := \text{lemma}(s_i) \in \text{gloss}(t_j).$$

This happens frequently, since a definition usually contains synonymous words or follows the *genus-differentia* pattern – i.e., providing a more specialized term (the *genus*) and the properties that distinguish the word from its co-hyponyms (the *differentia*). Consider, for instance, two word senses for *peck*: (i) "[…] *a dry measure of eight quarts"* and (ii) *"a great deal; a large or excessive quantity"*. The second one happens to be the correct disambiguation for the synonymy relation between *deal* and *peck* as it contains the source lemma *deal*.

Linguistic labels. Many word senses are domain-specific, such as the use of *host* as a certain kind of server in computer science. In dictionaries, domain-specific word senses are often marked by linguistic labels stating the domain, register, time, etc. this word sense is normally used in. An example is the sense *"(UK, pejorative) A working-class youth* […]*"* of *chav*. Relations usually connect two word senses of the same domain, register, etc. Hence, we add a feature

$$f_{\text{lbl}} := |\text{label}(s_i) \cap \text{label}(t_j)|$$

counting the number of labels shared by s_i and t_j. Since Wiktionary's linguistic labels are very heterogeneous and fine-grained, we manually grouped similar labels into broader categories; *zoology* and *ornithology* are, for instance, grouped into *biology*.

Inverse relation. Consider a relation between two polysemous words, such as the antonymy relation between $fall_i$ and *increase*. If there is a word sense j of *increase* for which an inverse antonymy relation ($increase_j$, *fall*) is encoded, then it is very likely that j is the correct disambiguation. Let relations(t_j) be the set of related lexical items of t_j. We define

$$f_{\text{inv}} := \text{lemma}(s_i) \in \text{relations}(t_j)$$

as the feature checking for inverse relations.

Relation overlap. The idea of inverse relations can be further extended by finding relations to other words shared by both the source and the target sense. A relation (*sweater*, *cloth*) can, for instance, be disambiguated by finding that one of their word senses shares a relation to *pullover* (a synonym of *sweater* and a hyponym of *cloth*). We define

$$f_{\text{rel}} := \frac{|\text{relations}(s_i) \cap \text{relations}(t_j)|}{|\text{relations}(s_i) \cup \text{relations}(t_j)|},$$

which is similar to the link-based similarity measure proposed by Milne and Witten (2008), who use hyperlinks from Wikipedia.

Commonness and Monosemy. The word senses of a lexicon are often ordered according to their usage frequencies in a corpus or the intuitions of the lexicographers. This has led to a very strong baseline for word sense disambiguation by always choosing the first sense. The same applies to the disambiguation of relations when choosing the first target sense. Therefore, we introduce a feature $f_{\text{idx}} := j$ that is set to the index of the target sense t_j.

Finally, we add a feature f_{mono} that is true if the target word has only one word sense, i.e. if it is monosemous. In these cases, it is most likely that this sense is the correct disambiguation; e.g., for the synonymy relation between *eggplant* and the monosemous word *brinjal*.

Cross-lingual features. Most of the features described above are also applicable in a multilingual setting when using translations instead of semantic relations. In order to also use the features based on sense definitions, we automatically translate them using the Bing translation[4] service. This opens up interesting research opportunities, since the definition of either the source or the target sense can be translated, i.e.

$$f_{\text{Lesk,TL}} := |\text{gloss}(\text{translate}(s_i)) \cap \text{gloss}(t_j)| \text{ or } f_{\text{Lesk,SL}} := |\text{gloss}(s_i) \cap \text{gloss}(\text{translate}(t_j))|.$$

There can even be a combined feature:

$$f_{\text{Lesk,SL\&TL}} := \tfrac{1}{2}(f_{\text{Lesk,SL}} + f_{\text{Lesk,TL}}).$$

Regarding the linguistic label feature f_{lbl}, we manually mapped English and German labels that represent the same meaning (e.g., *biology* and *Biologie*). This yielded a list of 19 label groups covering 1,267 distinct linguistic labels from two languages.

Constraints. In addition to the features introduced above, we can apply a threshold to convert a numeric feature into a boolean one. The notation $f_{\text{Lesk}\geq k}$ defines, for instance, a feature that is true if the sense definitions share at least k words. We use the notation $\hat{f}$ when only the target word sense with the highest feature value is used. The feature $\hat{f}_{\text{Lesk}\geq k}$ is thus true if, and only if, f_{Lesk} is higher than k and the maximum f_{Lesk} of all possible target word senses t.

[4] `http://www.microsofttranslator.com/`

3.2 Disambiguation Method

Let F be a set of features. Based on the notation introduced above, we define a generic relation disambiguating method

$$D\colon\ (r, t_j, F) \mapsto \{0, 1\},$$

returning 1 if t_j is a correct disambiguation for r and 0 otherwise. A basic method $D[f] = f$ uses only a single boolean feature $f \in F$. Thereby, we can model a most frequent sense baseline $MFS = D[f_{\text{idx}=1}]$ always using the first target word sense. One way of combining features is to concatenate them using a backoff strategy, i.e. a method

$$D[f_1 \circ f_2] = \begin{cases} D[f_1] & \text{if } f_1 \in F \\ D[f_2] & \text{otherwise} \end{cases}$$

relying on feature f_1 (if present) and f_2 otherwise. For example, $D[f_{\text{inv}} \circ f_{\text{idx}=1}]$ disambiguates those relations that have an inverse relation using f_{inv}. The remaining relations are disambiguated using a most frequent sense approach.

Based on the features introduced above, we now propose our disambiguation method

$$WKTWSD = D[f_{\text{mono}} \circ f_{\text{lbl}\geq 1} \circ f_{\text{rel}\geq 0.5} \circ f_{\text{src}} \circ f_{\text{inv}} \circ \hat{f}_{\text{ExtLesk}\geq 2} \circ f_{\text{idx}=1}]$$

that concatenates all features introduced above. For the cross-lingual datasets, we use $\hat{f}_{\text{ExtLesk}\geq 2,\text{SL\&TL}}$ instead of $\hat{f}_{\text{ExtLesk}\geq 2}$. The ordering and the thresholds have been chosen based on our analysis of the development data.

3.3 Empirical Evaluation

Comparison to previous work. Our experimental setup is directly comparable to the disambiguation of semantic relations in the German Wiktionary reported by Meyer and Gurevych (2010). They use a publicly available dataset, which consists of 250 manually disambiguated Wiktionary relations. Table 1 shows the performance of our proposed method in comparison with their text-similarity-based method *MG10*. Note that Meyer and Gurevych (2010) evaluated their system by measuring the agreement between the method and each of the two human raters. We therefore report A_O and Cohen's κ (Artstein and Poesio, 2008) following the original experimental setup. The inter-rater agreement serves as an upper bound and the most frequent sense baseline *MFS* is used as a lower bound. Our *WKTWSD* method outperforms their approach by a large margin. The improvement is statistically significant.[5]

Gold standard datasets. To our knowledge, there are no other evaluation datasets for disambiguating Wiktionary relations. That is why we create four new annotated datasets that consist of English semantic relations ($R_{\text{en:en}}$), German semantic relations ($R_{\text{de:de}}$), English–German translations ($R_{\text{en:de}}$), and German–English translations ($R_{\text{de:en}}$). The relations are sampled according to their type, the part of speech, and the number of candidates (i.e., possible target word senses) in order to create a balanced dataset.[6] Balancing out the datasets is very useful for being able to evaluate our approach separately for each sample group and to avoid datasets with a strong bias (e.g., on synonyms between nouns). None of the sampled relations occurs in our development data. Table 2 shows the numbers of sampled relations and the possible target senses (i.e., the number of annotations required).

[5] McNemar's test; $p < .05$

[6] Our sampling procedure is explained in detail in the supplementary material that is published with the datasets.

Method	$A_{O,1}$	$A_{O,2}$	κ_1	κ_2
MFS	.78	.79	.45	.50
MG10	.79	.82	.48	.57
WKTWSD	**.84**	**.85**	**.59**	**.65**
Human	.89	.89	.73	.73

Table 1: Comparison of our system to previous work

	$R_{\text{en:en}}$	$R_{\text{de:de}}$	$R_{\text{en:de}}$	$R_{\text{de:en}}$
Relations	394	459	204	204
Annotations	1,117	1,119	614	656
A_O	.91	.92	.89	.90
κ	.82	.85	.73	.75
F_1	.89	.92	.80	.83

Table 2: Statistics on our evaluation datasets

We then asked two human raters to annotate the monolingual datasets $R_{\text{en:en}}$ and $R_{\text{de:de}}$ and three raters to annotate the cross-lingual datasets $R_{\text{en:de}}$ and $R_{\text{de:en}}$. The raters should annotate each possible target word sense as being a correct ($D = 1$) or incorrect ($D = 0$) disambiguation for the given relation, for example:

s_i = *phenomenal*	D	t_j = *awesome*
(colloquial) Very remarkable; highly extraordinary; amazing.	0	Causing awe or terror; inspiring wonder or excitement.
(colloquial) Very remarkable; highly extraordinary; amazing.	1	(informal) Excellent, exciting, remarkable.

It was allowed to rate all target senses of a relation as incorrect (e.g., if the correct target sense has not yet been encoded in Wiktionary) or to rate more than one target sense as correct (e.g., if the target senses are more fine-grained than the source sense). Each rater was allowed to consult external sources such as lexicons, encyclopedias, etc. (and in particular Wiktionary itself). They were, however, not allowed to contact each other. The raters are native in German and speak English fluently. They have been trained using some example cases and an annotation guidebook that we publish along with the paper.

To estimate the reliability of our datasets, we measure the inter-rater agreement. Table 2 shows the observed agreement A_O and the kappa statistics κ for each dataset. We report Cohen's κ for the two rater case and Fleiss' κ (multi-π) for the three rater case (Artstein and Poesio, 2008). The raters agree on about 90% of the cases. The κ statistics of over .80 for the monolingual datasets suggests good reliability. The cross-lingual datasets have a slightly lower agreement. The disambiguation of translations hence seems to be more difficult for our raters. However, the κ scores are well above .67 and therefore allow us to draw tentative conclusions (Artstein and Poesio, 2008). We also provide F_1 scores for our datasets as suggested by Hripcsak and Rothschild (2005), which serve as upper bounds for our methods.

Finally, we create gold standard datasets based on the majority vote of the raters. As a tie breaker for the monolingual datasets, an additional adjudicator has been asked for a final decision. All datasets including analyses are freely available from our homepage.

Evaluation results. Table 3 shows the performance of our disambiguation method on the four gold standard datasets. We have counted the number of correct decisions $TP + TN$, the number of false positives *FP* and false negatives *FN*, which we use to report accuracy $A = \frac{TP+TN}{N}$, precision $P = \frac{TP}{TP+FP}$ (proportion of correctly disambiguated relations in the system result), recall $R = \frac{TP}{TP+FN}$ (proportion of correctly disambiguated relations in the gold standard), and the $F_1 = \frac{2PR}{P+R}$ score (Manning and Schütze, 1999). As a lower bound, we use the most frequent sense method *MFS*. The upper bound is human performance (*Human*) estimated by the inter-rater agreement A_O and the inter-rater F_1 score introduced above. Our *WKTWSD* method significantly outperforms the *MFS* baseline for each dataset. The only exception being the precision on the $R_{\text{de:en}}$ dataset, which is slightly lower than the precision of *MFS*.

Besides the lower and upper boundaries, we trained a number of machine learning classifiers for our set of features using the Weka toolkit (Hall et al., 2009). We report the results for a Naïve Bayes (*NB*) and a *J48* decision tree (a C4.5 clone) here, although we tried other classifiers as well, which generally yielded similar results. The training was done in a 5-fold cross validation. Note that we did not optimize the configuration in order to avoid overfitting to the datasets. In general, our *WKTWSD* method reaches a similar or even better performance than the machine learning classifiers. The main reason for this is the largely varying number of possible target word senses. While one relation might have only a single possible target sense, another one might have ten or even more. This tends to cause more false negatives in the machine learning methods and thus less relations that can be disambiguated. The finding is in line with previous work on gloss disambiguation: Moldovan and Novischi (2004) note that compiling a sufficient set of training examples is not possible in many cases. Despite this, the machine learning methods mostly achieve a slightly higher precision. *J48* even yields $P = .82$ for the $R_{\text{de:en}}$ dataset. However, this always comes at the cost of a lower recall.

Feature and error analysis. Table 4 shows the precision P and coverage C (proportion of items covered by this feature) of using each feature $f \in F$ individually. With the exception of $f_{\text{idx=1}}$ (most frequent sense strategy), none of the features is able to disambiguate the whole dataset, but most of them achieve a very high precision on the covered items. It is not surprising that f_{mono} performs extremely well ($P \in [.88, .96]$), since there is only one target word sense available for these cases. The feature f_{src} performs well on the monolingual datasets ($P \in [.87, .97]$), but does not work at all on the cross-lingual task ($P \in [.38, .50]$). The reasons for this are ambiguities in the sense definitions that are often not resolved by the machine translation service. Parallel ambiguities such as *commission* and *Kommission*, which both mean either a group of people or a transaction fee of a broker, is a main source of errors here. Similar errors also occur for f_{inv}. The word overlap feature $\hat{f}_{\text{ExtLesk}}$ generally shows a high precision. It is, in particular, higher than usually reported for word sense disambiguation tasks (Navigli, 2009). The reason might be that we do not compare a sense definition with context words, but two definitions with each other and hence benefit from comparing texts that are specially crafted to characterize word senses. Interestingly, the imprecise translation of certain words noted for f_{src} is less problematic for $\hat{f}_{\text{ExtLesk}\geq 2,\text{SL\&TL}}$, as there are usually at least some correctly translated words in the sense definition. In our experiments, we found that $\hat{f}_{\text{ExtLesk}\geq 2,\text{SL}}$ outperforms $\hat{f}_{\text{ExtLesk}\geq 2,\text{TL}}$, whereas $\hat{f}_{\text{ExtLesk}\geq 2,\text{SL\&TL}}$ is only marginally better than $\hat{f}_{\text{ExtLesk}\geq 2,\text{SL}}$. The English Wiktionary is very sparse in encoding semantic relations. The coverage of $f_{\text{rel}\geq 0.5}$ is therefore very low for all datasets involving English data.

Since we ordered the features manually for our *WKTWSD* method, we additionally define a method *BestOrder* which concatenates the features in descending order of their precision on

Method	$R_{\text{en:en}}$ A	P	R	F_1	$R_{\text{de:de}}$ A	P	R	F_1
MFS	.81	.75	.74	.74	.79	.78	.76	.77
WKTWSD	.84	.78	**.80**	**.79**	.84	.83	**.83**	**.83**
NB	.85	**.81**	.78	**.79**	.84	**.84**	.81	.82
J48	.83	**.81**	.71	.76	.84	.83	.82	**.83**
BestOrder	.85	.79	.80	.80	.85	.84	.83	.84
Human	.91			.89	.92			.92

Method	$R_{\text{en:de}}$ A	P	R	F_1	$R_{\text{de:en}}$ A	P	R	F_1
MFS	.79	.62	.72	.67	.79	.64	.66	.65
WKTWSD	.81	.64	**.75**	**.69**	.79	.62	**.71**	**.67**
NB	.81	.67	.69	.68	.82	.74	.61	**.67**
J48	.79	**.69**	.53	.60	.82	**.82**	.53	.64
BestOrder	.80	.63	.75	.69	.81	.67	.73	.70
Human	.89			.80	.90			.83

Table 3: Performance of our disambiguation methods on the four evaluation datasets

Feature	$R_{\text{en:en}}$ P	C	$R_{\text{de:de}}$ P	C	$R_{\text{en:de}}$ P	C	$R_{\text{de:en}}$ P	C
f_{mono}	.91	.21	.94	.22	.96	.08	.88	.08
f_{inv}	.78	.13	.89	.31	.68	.49	.67	.41
$f_{\text{lbl}\geq 1}$	.82	.07	.90	.05	.86	.02	.60	.04
f_{src}	.87	.10	.97	.07	.50	.20	.38	.18
$f_{\text{rel}\geq 0.5}$	.94	.04	.90	.14	.33	.01	.75	.01
$\hat{f}_{\text{ExtLesk}\geq 2}$	.89	.27	.99	.12	.87	.15	.93	.17
$f_{\text{idx}=1}$	.75	1.0	.78	1.0	.62	1.0	.64	1.0

Table 4: Precision and coverage of each feature

each dataset. The rationale behind this is that we make use of the best feature before moving to the next one. By comparing *WKTWSD* to *BestOrder*, we can measure the influence of our manually chosen ordering. Note, however, that *BestOrder* needs to be considered as an upper bound for *WKTWSD* rather than a separate method, because it made use of our analysis of the test data. The results can be found in Table 3. We observe that the order of the features plays only a minor role: *WKTWSD* and *BestOrder* are only slightly different although they concatenate the features in totally different ways. The largest difference accounts to .03 for the $R_{\text{de:en}}$ dataset and is mostly due to the low performance of f_{src}.

Summary. We conclude that our approach is better suited for disambiguating Wiktionary relations than previous works using textual similarity. The features are effectively applied using a concatenation method. The training of machine learning classifiers could not improve these results in our experiments.

	Our resource		Wordnets	
	English	German	WordNet	GermaNet
Lexical entries	379,694	85,574	156,584	85,257
Word senses	474,128	73,500	206,978	96,690
Semantic relations	215,353	300,724	1,398,868	512,653
…Synonyms	*70,199*	*78,133*	*315,984*	*74,552*
…Antonyms	*35,291*	*33,391*	*7,979*	*3,359*
…Hypo-/Hypernyms	*54,494*	*87,246*	*658,804*	*397,335*
…Other types	*55,269*	*101,954*	*416,101*	*37,407*
Translations	79,382		16,347	

Table 5: Statistics on our new resource in comparison to WordNet and GermaNet

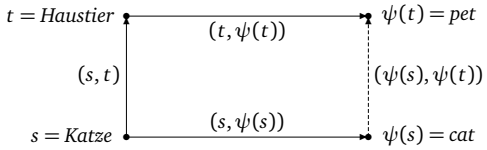


Figure 2: Cross-lingual inference of the semantic relation $(\psi(s), \psi(t))$

4 A Multilingual, Sense-Disambiguated Wiktionary

Resource construction. We create our new multilingual resource by using all word senses encoded in a given set of Wiktionary language editions (English and German in our experiments). Then, we perform the automatic disambiguation of the semantic relations and translations to obtain a fully disambiguated resource. We use the *WKTWSD* method for this task, as it performed well on our evaluation datasets. The disambiguated translations allow for extracting lexical-semantic knowledge in multiple languages. The first sense of *(to) stroll* is, for instance, *"to wander on foot* […]*"* in the English Wiktionary. When following its translations, we are able to extract the German equivalent *spazieren*: *"gemächlich gehen* […]*"*. In this way, we can also obtain multilingual example sentences, linguistic labels, etc.

Inference of relations. For semantic relations, we can even further benefit from their disambiguated target senses: Let (s, t) be a disambiguated semantic relation in one of the Wiktionary language editions and let $(s, \psi(s))$ and $(t, \psi(t))$ be disambiguated translations of s and t into another language. Assuming a correct disambiguation of these three relations, we can infer a fourth relation $(\psi(s), \psi(t))$, since the meaning of s and t is preserved under the disambiguated translations. Figure 2 shows an example: For the German hypernym (*Katze*, *Haustier*) and the corresponding translations (*Katze, cat*) and (*Haustier, pet*), we can infer the English hypernymy relation (*cat, pet*) that is currently not encoded in the English Wiktionary. Note that the inferred relation is also sense-disambiguated, i.e. both *cat* and *pet* refer to the animal sense.

Size of our resource. Our final resource contains 215,353 English and 300,724 German semantic relations. The English Wiktionary benefits most from inferring new semantic relations: We increased the number of relations found in the original Wiktionary (26,965) by almost an order of magnitude. But also for the German language, we were able to infer 10,705 new semantic relations. In addition to that, our resource consists of 474,128 English and

73,500 German word senses as well as 79,382 translations (45,246 English–German and 34,136 German–English). Table 5 shows detailed statistics of our resource including the most common types of the encoded semantic relations. We compare our resource with the Princeton WordNet (Fellbaum, 1998) and GermaNet (Kunze and Lemnitzer, 2002) and their inter-lingual index (which is a part of EuroWordNet). Our resource surpasses the number of translations by a large margin, but contains less semantic relations than in the expert-built wordnets. The coverage of lexical entries and word senses is comparable or higher.

5 Measuring Verb Similarity

To demonstrate the usefulness of our resource, we carry out two experiments employing the newly created resource in a monolingual and cross-lingual verb similarity task. Judging verb similarity is of particular interest for applications such as cross-lingual word sense disambiguation (Lefever and Hoste, 2010), lexical substitution (Mihalcea et al., 2010), or question answering (Magnini et al., 2005). State of the art knowledge-based systems rely heavily on Wikipedia, which predominantly encodes encyclopedic knowledge about nouns. The large amount of multilingual lexical-semantic knowledge in Wiktionary let us expect good results not only for nouns, but also for other parts of speech and verbs in particular.

Monolingual verb similarity. Yang and Powers (2006) introduced an evaluation dataset for verb similarity that consists of 130 English verb pairs taken from TOEFL and ESL (English as a second language) questions. For each of them, a numerical score is provided expressing the human intuitions of their similarity. These scores are averaged over six human annotators that were asked to rate the similarity of each pair on a graded scale from 0 (*not at all related*) to 4 (*inseparably related*). Yang and Powers (2006) report a correlation of $r = 0.866$ between the raters. An example from their dataset is the verb pair (*approve, support*) with a score of 3.

To the best of our knowledge, Zesch et al. (2008b) reports the latest evaluation results on this dataset as shown in column *Z08* of Table 6. They use explicit semantic analysis, a method based on concept vectors (Gabrilovich and Markovitch, 2007) built from WordNet, Wikipedia, and the undisambiguated Wiktionary. Each entry from these resources (synsets in WordNet and wiki pages in Wikipedia and Wiktionary) is regarded as one concept. For a given word pair, two concept vectors are then created that consist of the word's tf-idf scores over the concepts. The similarity for this word pair is then expressed by the cosine of the two concept vectors. Although Zesch et al. (2008b) find Wiktionary to yield best results for computing semantic relatedness between nouns, the performance for verb similarity is substantially lower than using WordNet. One reason for that is the high degree of polysemy of verbs, which is not dealt with by their approach. Since our resource is completely sense-disambiguated, we can, in contrast, compute sense-disambiguated concept vectors using each word sense as one concept.

We reproduced the results of Zesch et al. (2008b), also using WordNet, Wikipedia, and the undisambiguated Wiktionary, and show them in the column $V_{\text{en:en}}$ of Table 6. Note that we use all 130 verb pairs, whereas Zesch et al. (2008b) used only the 80 pairs that were covered by all three similarity metrics they tried. Therefore, our scores slightly differ from *Z08*. In addition to the three resources, we report the performance when using the sense-disambiguated concept vectors derived from our resource. Using our resource yields better results than using Wikipedia or the undisambiguated Wiktionary. The previously best resource WordNet is slightly outperformed by our resource. This difference is, however, not statistically significant. All four concept-vector-based methods cover 100% of the dataset and are thus directly comparable.

Resource	*Z08*	$V_{\text{en:en}}$	$V_{\text{de:de}}$	$V_{\text{en:de}}$	$V_{\text{de:en}}$
WN/GN	**.71**	.69	**.57**	.31	.23
Wikipedia	.29	.27	.33	.23	.28
Wiktionary	.65	.63	.36	—	—
Our resource	—	**.73**	.52	**.53**	**.51**
Coverage	62%	100%	92%	95%	97%

Table 6: Evaluation results on the four verb similarity datasets using concept vectors from WordNet/GermaNet (WN/GN), Wikipedia, Wiktionary, and our new resource in comparison to previous work by Zesch et al. (2008b) (*Z08*). Performance is measured by Spearman's rank correlation coefficient using Horn's correction for tied ranks (Horn, 1942). All correlations significantly differ from random (two-tailed paired t-test; $p < .05$).

We also study German verb similarity and therefore translate the $V_{\text{en:en}}$ dataset. The verb pair (*approve, support*) is, for instance, translated to (*annehmen, unterstützen*) keeping its similarity score of 3. Table 6 shows the results for this new $V_{\text{de:de}}$ dataset. To create the concept vectors, we use GermaNet instead of WordNet as well as the German editions of Wikipedia and Wiktionary. We use only the 120 verb pairs covered by all four resources. Our resource is again able to outperform Wikipedia and the undisambiguated Wiktionary by a wide margin. The performance competes with the expert-built GermaNet, but is slightly lower than that. As opposed to the English language, GermaNet and the German part of our resource are similar in size (see Table 5), which can explain these results. This is why we expect better results with the growth of the German Wiktionary. Furthermore, we conclude that our resource can be a promising alternative for languages with less developed expert-built resources.

Cross-lingual verb similarity. Based on the English and the German verb pairs, we create two cross-lingual verb similarity datasets that use the first English verb together with the second German verb from each corresponding verb pair $V_{\text{en:de}}$ and, vice versa, the first German verb together with the second English verb $V_{\text{de:en}}$. For the example introduced above, this yields the two verb pairs (*approve, unterstützen*) and (*annehmen, support*), both with a score of 3.

Table 6 shows the evaluation results using these two datasets. To create the cross-lingual concept vectors, we use the inter-lingual index between WordNet and GermaNet, the interwiki links from Wikipedia, and the disambiguated translations from our new resource. Since the translations of the original Wiktionary are not sense-disambiguated, they cannot be used to build cross-lingual concept vectors.[7] As noted in Section 2, the inter-lingual index of WordNet and GermaNet (which is part of EuroWordNet) is very small. Consequently, we observe that the expert-built wordnets yield a substantially lower performance for $V_{\text{en:de}}$ and $V_{\text{de:en}}$ than in the monolingual setting. Wikipedia likewise yields low scores because of its lack of the knowledge about verbs, whereas our resource significantly outperforms ($p < .01$) both the expert-built wordnets and Wikipedia.

Our error analysis shows that many of the judgments derived from our resource are useful. The predominant problem is still the coverage of the translations. The similarity of the English–German verb pair *(concoct, ausarbeiten)* is, for instance, not yet backed up by a translation in Wiktionary and is hence underestimated by the system. While this is essentially the same problem as for the wordnets, the problem is much less severe for our resource.

[7] Wiktionary also encodes interwiki links for each wiki page, but they link to the same form (e.g. from *walk* in the English Wiktionary to *walk* in the German Wiktionary) rather than to translations and thus cannot be used.

Summary. Wikipedia-based resources are not very appropriate for computing verb similarity as they focus on encyclopedic knowledge about nouns. Expert-built wordnets work well for computing monolingual verb similarity, because they have a sufficient coverage and encode thoroughly elaborated lexical-semantic knowledge. Our new disambiguated Wiktionary-based resource competes with their quality. Since Wiktionary is available in over 170 languages, our approach is, however, also applicable to the languages lacking large expert-built resources. In a cross-lingual setting, this shows a different picture: Expert-built multilingual wordnets suffer from their small size. Since the disambiguated translations in our resource let us build cross-lingual concept vectors, they can be effectively utilized in this task.

Conclusion and perspectives

We have created a new multilingual, sense-disambiguated resource using the word senses from Wiktionary and interconnecting them by means of disambiguated semantic relations and translations. For the automatic disambiguation of the relations, we proposed and evaluated a rule-based method using seven different features. Our features are similar to those used by Moldovan and Novischi (2004), whereas we adjusted them to our specific task and generalized them to the cross-lingual setting. We found our method to significantly outperform a previous approach based on textual similarity (Meyer and Gurevych, 2010). In a second evaluation based on four newly created datasets, we obtained promising results exceeding the baseline in every case. Using the disambiguated relations, we inferred a large number of new semantic relations and thereby yielded almost a tenfold increase in the number of relations for the English language. Our final resource fills the gap between small expert-built multilingual wordnets and Wikipedia-based resources, which are mostly restricted to the encyclopedic knowledge about nouns. The new resource and all evaluation data is publicly available for research.[8]

We also employed our new resource in a monolingual and cross-lingual verb similarity task. Besides the standard dataset by Yang and Powers (2006), we created a novel German and two cross-lingual verb similarity datasets. Our resource competes with expert-built wordnets in the monolingual setting. Since Wiktionary is available in many languages, this allows for computing verb similarity also for languages lacking large expert-built resources. In the cross-lingual setting, our sense-disambiguated resource outperforms both Wikipedia and the expert-built wordnets. The former suffers from the small amount of knowledge about verbs and the latter lack coverage of the inter-lingual index.

In future work, we plan to combine our resource with BabelNet or UBY (Gurevych et al., 2012) in order to benefit from the heterogeneous knowledge found in WordNet, Wikipedia, and our resource. Extending our resource to other languages and exploring alternative disambiguation algorithms such as CQC are further promising options. We will also consider providing our inferred semantic relations to the Wiktionary community to contribute to the harmonization of Wiktionary data. Besides verb similarity, our sense-disambiguated resource has the potential to improve other natural language processing tasks as well, for instance, question answering.

Acknowledgments

This work has been supported by the Volkswagen Foundation as part of the Lichtenberg-Professorship Program under grant No. I/82806. We thank Christian Kirschner, Dr. Judith Eckle-Kohler, and Dr. Torsten Zesch for their contributions to this project as well as Dongqiang Yang and David M. W. Powers for sharing their verb similarity dataset.

[8] http://www.ukp.tu-darmstadt.de/data/lexical-resources/wiktionary/

References

Artstein, R. and Poesio, M. (2008). Inter-Coder Agreement for Computational Linguistics. *Computational Linguistics*, 34(4):555–596.

Atserias, J. and Villarejo, Luís Rigau, G. (2004). Spanish WordNet 1.6: Porting the Spanish WordNet Across Princeton Versions. In *Proceedings of the Fourth International Conference on Language Resources and Evaluation*, pages 161–164, Lisbon, Portugal.

Banerjee, S. and Pedersen, T. (2003). Extended Gloss Overlaps as a Measure of Semantic Relatedness. In *Proceedings of the 18th International Joint Conference on Artificial Intelligence*, pages 805–810, Acapulco, Mexico.

Bizer, C., Lehmann, J., Kobilarov, G., Auer, S., Becker, C., Cyganiak, R., and Hellmann, S. (2009). DBpedia – A Crystallization Point for the Web of Data. *Journal of Web Semantics: Science, Services and Agents on the World Wide Web*, 7(3):154–165.

Budanitsky, A. and Hirst, G. (2006). Evaluating WordNet-based Measures of Lexical Semantic Relatedness. *Computational Linguistics*, 32(1):13–47.

de Melo, G. and Weikum, G. (2009). Towards a Universal Wordnet by Learning from Combined Evidence. In *Proceedings of the 18th ACM Conference on Information and Knowledge Management*, pages 513–522, Hong Kong, China.

Fellbaum, C., editor (1998). *WordNet: An Electronic Lexical Database (Language, Speech, and Communication)*. Cambridge, MA: MIT Press.

Flati, T. and Navigli, R. (2012). The CQC Algorithm: Cycling in Graphs to Semantically Enrich and Enhance a Bilingual Dictionary. *Journal of Artificial Intelligence Research*, 43:135–171.

Gabrilovich, E. and Markovitch, S. (2007). Computing Semantic Relatedness using Wikipedia-based Explicit Semantic Analysis. In *Proceedings of the 20th International Joint Conference on Artificial Intelligence*, pages 1606–1611, Hyderabad, India.

Gurevych, I., Eckle-Kohler, J., Hartmann, S., Matuschek, M., Meyer, C. M., and Wirth, C. (2012). UBY – A Large-Scale Unified Lexical-Semantic Resource Based on LMF. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 580–590, Avignon, France.

Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., and Witten, I. H. (2009). The WEKA Data Mining Software: An Update. *ACM SIGKDD Explorations Newsletter*, 11(1):10–18.

Harabagiu, S. M., Miller, G. A., and Moldovan, D. I. (1999). WordNet 2 - A Morphologically and Semantically Enhanced Resource. In *Proceedings of the ACL Special Interest Group on the Lexicon Workshop on Standardizing Lexical Resources*, pages 1–7, College Park, MD, USA.

Herbert, B., Szarvas, G., and Gurevych, I. (2011). Combining Query Translation Techniques to Improve Cross-Language Information Retrieval. In Clough, P., Foley, C., Gurrin, C., Jones, G. J., Kraaij, W., Lee, H., and Murdoch, V., editors, *Advances in Information Retrieval: 33rd European Conference on IR Research*, volume 6611 of *Lecture Notes in Computer Science*, pages 712–715. Berlin/Heidelberg: Springer.

Horn, D. (1942). A Correction for the Effect of Tied Ranks on the Value of the Rank Difference Correlation Coefficient. *Journal of Educational Psychology*, 33(9):686–690.

Hripcsak, G. and Rothschild, A. S. (2005). Agreement, the F-Measure, and Reliability in Information Retrieval. *Journal of the American Medical Informatics Association*, 12(3):296–298.

Huang, C.-R., Tseng, I.-J. E., and Tsai, D. B. S. (2002). Translating lexical semantic relations: the first step towards multilingual wordnets. In *Proceedings of the COLING '02 Workshop on 'Building and Using Semantic Networks'*, Taipei, Taiwan.

Kikui, G. (1999). Resolving Translation Ambiguity Using Non-parallel Bilingual Corpora. In *Proceedings of the ACL '99 Workshop on 'Unsupervised Learning in Natural Language Processing'*, pages 31–36, College Park, MD, USA.

Krovetz, R. (1992). Sense-Linking in a Machine Readable Dictionary. In *Proceedings of the 30th Annual Meeting of the Association for Computational Linguistics*, pages 330–332, Newark, DE, USA.

Kunze, C. and Lemnitzer, L. (2002). GermaNet — representation, visualization, application. In *Proceedings of the Third International Conference on Language Resources and Evaluation*, volume 5, pages 1485–1491, Las Palmas, Canary Islands, Spain.

Lefever, E. and Hoste, V. (2010). SemEval-2010 Task 3: Cross-Lingual Word Sense Disambiguation. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, pages 15–20, Uppsala, Sweden.

Lesk, M. (1986). Automatic Sense Disambiguation Using Machine Readable Dictionaries: How to Tell a Pine Cone from an Ice Cream Cone. In *Proceedings of the 5th Annual International Conference on Systems Documentation*, pages 24–26, Toronto, ON, Canada.

Magnini, B., Vallin, A., Ayache, C., Erbach, G., Peñas, A., de Rijke, M., Rocha, P., Simov, K., and Sutcliffe, R. (2005). Overview of the CLEF 2004 Multilingual Question Answering Track. In Peters, C., Clough, P., Gonzalo, J., Jones, G. J., Kluck, M., and Magnini, B., editors, *Multilingual Information Access for Text, Speech and Images: 5th Workshop of the Cross-Language Evaluation Forum,*, volume 3491 of *Lecture Notes in Computer Science*, pages 371–391. Berlin/Heidelberg: Springer.

Manning, C. D. and Schütze, H. (1999). *Foundations of Statistical Natural Language Processing*. Cambridge, MA: The MIT Press.

Mausam, Soderland, S., Etzioni, O., Weld, D., Skinner, M., and Bilmes, J. (2009). Compiling a Massive, Multilingual Dictionary via Probabilistic Inference. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 262–270, Singapore.

Medelyan, O., Legg, C., Milne, D., and Witten, I. H. (2009). Mining Meaning from Wikipedia. *International Journal of Human-Computer Studies*, 67(9):716–754.

Meyer, C. M. and Gurevych, I. (2010). Worth its Weight in Gold or Yet Another Resource — A Comparative Study of Wiktionary, OpenThesaurus and GermaNet. In Gelbukh, A., editor, *Computational Linguistics and Intelligent Text Processing: 11th International Conference*, volume 6008 of *Lecture Notes in Computer Science*, pages 38–49. Berlin/Heidelberg: Springer.

Mihalcea, R. and Moldovan, D. (2001). eXtended WordNet: Progress Report. In *Proceedings of the NAACL '01 Workshop 'WordNet and Other Lexical Resources: Applications, Extensions and Customizations'*, pages 95–100, Pittsburgh, PA, USA.

Mihalcea, R., Sinha, R., and McCarthy, D. (2010). SemEval-2010 Task 2: Cross-Lingual Lexical Substitution. In *Proceedings of the 5th International Workshop on Semantic Evaluations*, pages 9–14, Uppsala, Sweden.

Milne, D. and Witten, I. H. (2008). An Effective, Low-Cost Measure of Semantic Relatedness Obtained from Wikipedia Links. In *Proceedings of the AAAI '08 Workshop 'Wikipedia and Artificial Intelligence: An Evolving Synergy'*, pages 25–30, Chicago, IL, USA.

Moldovan, D. and Novischi, A. (2004). Word sense disambiguation of WordNet glosses. *Computer Speech and Language*, 18(3):301–317.

Nastase, V., Strube, M., Börschinger, B., Zirn, C., and Elghafari, A. (2010). WikiNet: A very large scale multi-lingual concept network. In *Proceedings of the 7th International Conference on Language Resources and Evaluation*, pages 1015–1022, Valetta, Malta.

Navigli, R. (2009). Word sense disambiguation: A survey. *ACM Computing Surveys*, 41(2):1–69.

Navigli, R. and Ponzetto, S. P. (2010). BabelNet: Building a Very Large Multilingual Semantic Network. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 216–225, Uppsala, Sweden.

Neff, M. S. and McCord, M. C. (1990). Acquiring Lexical Data from Machine-Readable Dictionary Resources for Machine Translation. In *Proceedings of the Third International Conference on Theoretical and Methodological Issues in Machine Translation of Natural Languages*, pages 85–90, Austin, TX, USA.

Oepen, S., Velldal, E., Lønning, J. T., Meurer, P., Rosén, V., and Flickinger, D. (2007). Towards Hybrid Quality-Oriented Machine Translation: On Linguistics and Probabilities in MT. In *Proceedings of the 11th International Conference on Theoretical and Methodological Issues in Machine Translation*, pages 144–153, Skövde, Sweden.

Pantel, P. and Pennacchiotti, M. (2008). Automatically Harvesting and Ontologizing Semantic Relations. In Buitelaar, P. and Cimiano, P., editors, *Ontology Learning and Population: Bridging the Gap between Text and Knowledge*, pages 171–198. Amsterdam: IOS Press.

Pianta, E., Bentivogli, L., and Girardi, C. (2002). MultiWordNet: Developing an aligned multilingual database. In *Proceedings of the First International WordNet Conference*, pages 293–302, Mysore, India.

Sagot, B. and Fišer, D. (2008). Building a free French wordnet from multilingual resources. In *Proceedings of the LREC '08 Workshop 'Ontologies and Lexical Resources'*, pages 14–19, Marrakech, Morocco.

Stamou, S., Oflazer, K., Pala, K., Christoudoulakis, D., Cristea, D., Tufiş, D., Koeva, S., Totkov, G., Dutoit, D., and Grigoriadou, M. (2002). BALKANET: A Multilingual Semantic Network for the Balkan Languages. In *Proceedings of the First International WordNet Conference*, pages 12–14, Mysore, India.

Suchanek, F. M., Kasneci, G., and Weikum, G. (2007). Yago: A Core of Semantic Knowledge. In *Proceedings of the 16th International World Wide Web Conference*, pages 697–706, Banff, AB, Canada.

Thoongsup, S., Charoenporn, T., Robkop, K., Sinthurahat, T., Mokarat, C., Sornlertlamvanich, V., and Isahara, H. (2009). Thai WordNet Construction. In *Proceedings of the 7th ACL/IJCNLP '09 Workshop on 'Asian Language Resources'*, pages 139–144, Singapore.

Tsunakawa, T. and Kaji, H. (2010). Augmenting a Bilingual Lexicon with Information for Word Translation Disambiguation. In *Proceedings of the Eighth COLING '10 Workshop on 'Asian Language Resources'*, pages 30–37, Beijing, China.

Tufiş, D., Ion, R., Barbu, E., and Barbu, V. (2004). Cross-Lingual Validation of Multilingual Wordnets. In *Proceedings of the Second Global WordNet Conference*, pages 332–340, Brno, Czech Republic.

Vossen, P. (1998). Introduction to EuroWordNet. *Computers and the Humanities*, 32(2–3):73–89.

Yang, D. and Powers, D. M. W. (2006). Verb Similarity on the Taxonomy of WordNet. In *Proceedings of the Third International WordNet Conference*, pages 121–128, Jeju Island, Korea.

Zesch, T., Müller, C., and Gurevych, I. (2008a). Extracting Lexical Semantic Knowledge from Wikipedia and Wiktionary. In *Proceedings of the 6th International Conference on Language Resources and Evaluation*, pages 1646–1652, Marrakech, Morocco.

Zesch, T., Müller, C., and Gurevych, I. (2008b). Using Wiktionary for Computing Semantic Relatedness. In *Proceedings of the Twenty-Third AAAI Conference on Artificial Intelligence*, pages 861–867, Chicago, IL, USA.

Using Distributional Similarity for Lexical Expansion in Knowledge-based Word Sense Disambiguation

Tristan Miller[1] *Chris Biemann*[1] *Torsten Zesch*[1,2] *Iryna Gurevych*[1,2]
(1) Ubiquitous Knowledge Processing Lab (UKP-TUDA)
Department of Computer Science, Technische Universität Darmstadt
(2) Ubiquitous Knowledge Processing Lab (UKP-DIPF)
German Institute for Educational Research and Educational Information
http://www.ukp.tu-darmstadt.de/

ABSTRACT
We explore the contribution of distributional information for purely knowledge-based word sense disambiguation. Specifically, we use a distributional thesaurus, computed from a large parsed corpus, for lexical expansion of context and sense information. This bridges the lexical gap that is seen as the major obstacle for word overlap–based approaches. We apply this mechanism to two traditional knowledge-based methods and show that distributional information significantly improves disambiguation results across several data sets. This improvement exceeds the state of the art for disambiguation without sense frequency information—a situation which is especially encountered with new domains or languages for which no sense-annotated corpus is available.

TITLE AND ABSTRACT IN GERMAN

Über die Bestimmung lexikalischer Expansionen mittels distributioneller Ähnlichkeit und deren Einsatz in der wissensbasierten Lesartendisambiguierung

Wir untersuchen den Einfluss distributioneller Informationen auf die rein wissensbasierte Lesartendisambiguierung. Basierend auf einem distributionellen Thesaurus, den wir aus einem großen geparsten Korpus erzeugen, erweitern wir die Definition der Lesart und deren Kontext mit lexikalischen Expansionen. Dadurch schließen wir die 'lexikalische Lücke', die sich als Haupthindernis für Ansätze basierend auf Wortgemeinsamkeiten herausgestellt hat. Wir erweitern zwei klassische wissensbasierte Ansätze um lexikalische Expansionen und zeigen, dass dadurch die Qualität der Lesartendisambiguierung deutlich erhöht wird. Wir erzielen die bisher besten veröffentlichten Ergebnisse für Disambiguierung ohne Nutzung der Lesartenhäufigkeiten, was besonders für Domänen oder Sprachen relevant ist, für die keine Lesarten-annotierten Korpora zur Verfügung stehen.

KEYWORDS: word sense disambiguation, distributional thesaurus, lexical expansion.

KEYWORDS IN GERMAN: Lesartendisambiguierung, distributioneller Thesaurus, lexikalische Expansion.

Proceedings of COLING 2012: Technical Papers, pages 1781–1796,
COLING 2012, Mumbai, December 2012.

1 Introduction

Word sense disambiguation (WSD)—the task of determining which sense a word carries in a particular context—is a longstanding core research problem in computational linguistics. Approaches to WSD can be classified according to what lexical resources are used: *knowledge-based* techniques rely only on machine-readable dictionaries (MRDs), lexical semantic resources (LSRs), and untagged corpora, whereas *supervised* approaches instead or additionally use manually annotated training examples. Though supervised systems generally perform better, their use is restricted to scenarios where a sufficient amount of hand-crafted training data is available. Estimates for the amount of time required to produce such training data are pessimistic (Mihalcea and Chklovski, 2003); this knowledge acquisition bottleneck is the principal motivation behind research into semi-supervised and knowledge-based WSD. The latter have the advantage that, unlike manually annotated corpora, MRDs and LSRs do exist for many languages and domains.

In the past, however, knowledge-based approaches have suffered from a variant of the lexical gap problem: when matching a sense description to a given context of a disambiguation target, it is often the case that the description and context do not have much vocabulary in common. We propose a new method to bridge this lexical gap which is based on statistics collected from a large, unannotated background corpus. Specifically, we enrich the textual information from the context and the MRD with lexical expansions produced by a distributional thesaurus.

We examine the contribution of these expansions to two popular knowledge-based algorithms, including one which tries to address the lexical gap through LSR-based augmentation of the sense description. We show that, especially in situations for which no sense frequency information is available, improvements from adding more knowledge and from adding lexical expansions add up, allowing us to improve over the state of the art for knowledge-based all-words disambiguation.

2 Background

MRD-based word sense disambiguation began with Lesk (1986), who proposed that two or more words in context could be simultaneously disambiguated by looking up their respective definitions in a dictionary and finding the maximum overlap between each combination of their senses. A popular variant is the "simplified" Lesk algorithm (Kilgarriff and Rosenzweig, 2000), which disambiguates one word at a time by comparing each of its definitions to the context in which the word is found. This variant avoids the combinatorial explosion of word sense combinations the original version suffers from when trying to disambiguate multiple words in a text.

Both the original and simplified versions of the Lesk algorithm suffer from low coverage due to the lexical gap problem: because the context and definitions are usually quite short, it is often the case that there are no overlapping content words at all. Various solutions to the problem have been proposed, with varying degrees of success. Lesk himself proposed increasing the size of the context window, though Vasilescu et al. (2004) found that performance was generally better for smaller contexts. Lesk also proposed augmenting the definitions with example sentences provided by some dictionaries; Kilgarriff and Rosenzweig (2000) found that including them (for simplified Lesk) led to significantly better performance than using the definitions alone. Banerjee and Pedersen (2002) observed that, where there exists a lexical resource like WordNet (Fellbaum, 1998) which also provides semantic relations between senses,

these can be used to augment definitions with those from related senses (such as hypernyms and hyponyms); their "extended" Lesk algorithm was found to be a great improvement over the original algorithm. Subsequent researchers (e.g., Ponzetto and Navigli (2010)) have combined the "simplified" and "extended" approaches into a "simplified extended" algorithm, in which augmented definitions are compared not with each other, but with the target word context.

Many successful approaches to automatic WSD in recent years rely on distributional information to model the "topicality" of the context and the sense definition.[1] They include using vector-space dimensionality reduction techniques like LSA (Gliozzo et al., 2005) or LDA (Cai et al., 2007; Li et al., 2010), additionally collected text material per sense as in topic signatures (Martínez et al., 2008), and clustering for word sense induction as features (Agirre et al., 2006; Biemann, 2012); the importance of bridging the lexical gap is reflected in all those recent advances, be it in knowledge-based or supervised WSD scenarios.

In this paper, we employ a source of semantic similarity whose application to automatic WSD has never before been explored: using a distributional thesaurus, or DT (Lin, 1998), we expand the lexical representations of the context and sense definition with additional terms. On this expanded representation, we are able to apply the well-known overlap-based methods to text similarity without any modification. Lexical expansion has already proven useful in semantic text similarity evaluations (Bär et al., 2012), which is a task related to matching sense definitions to contexts.

The intuition behind our approach is depicted in Figure 1: say we wish to disambiguate the word *interest* in the sentence, "The loan interest is paid monthly." The correct sense definition from our MRD ("a fixed charge for borrowing money") has no words in common with the context, and thus would not be selected by an overlap-based WSD algorithm. But with the addition of ten lexical expansions per content word (shown in smaller text), we increase the number of overlapping word pairs (shown in boldface) to seven.

Observe also that this expansion of linear text sequences into a two-dimensional representation makes conceptual associations (cf. the associative relations of de Saussure (1916)) explicit, allowing for purely symbolic matching instead of using a vector-space representation such as LSA. The main differences to vector-space approaches are the following: On the one hand, vector-space approaches usually use dimensionality reduction in order to handle sparsity, which results in a fixed number of topics/dimensions. While very salient collection-specific topics are handled well by this approach, rare topics are either conflated into a single rump topic, or distributed amongst the salient topics. Our DT-based expansion technique has no notion of dimensions since it works on the word level, and thus does not suffer from this kind of sampling error that is inevitable when representing a large vocabulary with a small fixed number of dimensions or topics. On the other hand, while vector-space models do a good job at ranking candidates according to their similarity,[2] they fail to efficiently generate a top-ranked list of possible expansions: due to its size, it is infeasible to rank the full vocabulary every time. Lexical expansion methods based on distributional similarity, however, generate a short list of highly similar candidates.

[1] Distributional information was also used in a much older, semi-automatic approach by Tugwell and Kilgarriff (2001). In their technique, "word sketches" consisting of common patterns of usage of a word were extracted from a large POS-tagged corpus and presented to a human operator for manual sense annotation. The pattern–sense associations were then used as input to bootstrapping WSD algorithm.

[2] See Rapp (2004) for an early success of vector-space models on a semantic task.

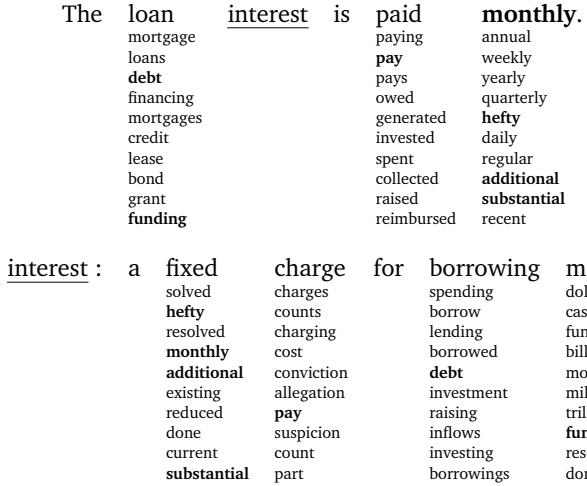


Figure 1: Example showing the intuition behind lexical expansion for matching a context (top) to a sense definition (bottom). The term to be disambiguated is underlined and the matching terms are in boldface.

The lexical expansions shown in Figure 1 were generated by the same DT used in our experiments. However, for the general case, we make no assumptions about the method that generates the lexical expansions, which could just as easily come from, say, translations via bridge languages, paraphrasing systems, or lexical substitution systems.

3 Experiments

Our experiments measure the contribution of various lexical expansion schemes to the simplified and simplified extended variants of the Lesk algorithm. We chose these algorithms because of their simplicity and transparency, making it easy for us to trace through their operation and see exactly how and where the lexical expansions help or hinder disambiguation. Furthermore, Lesk variants perform remarkably well despite their simplicity, making them popular choices as baselines and as starting points for developing more sophisticated WSD algorithms.

Our experiments with the simplified Lesk algorithm use only the definitions provided by WordNet; they are intended to model the case where we have a generic MRD which provides sense definitions, but no additional lexical semantic information such as example sentences or semantic relations. Such scenarios are typical of many languages and domains, where there is no WordNet-like resource and no manually sense-annotated corpus which could be used for supervised WSD or for a backoff to the most frequent sense. Accurate WSD systems that rely on the existence of an MRD only could pave the way to wider application of lexical disambiguation in NLP applications.

By contrast, the experiments with the simplified extended Lesk algorithm assume the existence of a WordNet-like resource with a taxonomic structure; the definition text for a sense is therefore constructed from the gloss, synonyms, and example sentences provided by WordNet, plus the

same information for all senses in a direct semantic relation. This setup specifically targets situations where such a resource serves as the sense inventory but no large sense-annotated corpus is available for supervised WSD (thus precluding use of the most frequent sense backoff). This is the case for many languages, where wordnets but not manually tagged corpora are available, and also for domain-specific WSD using the English WordNet. Whereas other approaches in this setting (Ponzetto and Navigli, 2010; Henrich et al., 2012) aim at improving WSD accuracy through the combination of several lexical resources, we restrict ourselves to WordNet and bridge the lexical gap with non-supervised, data-driven methods.

How one computes the overlap between two strings was left unspecified by Lesk; we therefore adopt the simple approach of removing occurrences of the target word, treating both strings as bags of case-insensitive word tokens, and taking the cardinality of their intersection. We do not preprocess the texts by lemmatization or stop word filtering, since the terms in the distributional thesaurus are likewise unprocessed (as in Figure 1), and because preliminary experiments showed that such preprocessing brought no benefit. We use the sentence containing the target word as the context. The sense with the highest overlap with the context is assigned a probability of 1; when $k \geq 2$ senses are tied for the highest overlap count, these senses are assigned a probability of $1/k$. All other senses are assigned a probability of 0. The probabilities are then used for scoring during evaluation (see §3.3).

3.1 Use of distributional information

We now describe the creation and the use of our distributional thesaurus. In the fashion of Lin (1998), we parsed a 10M sentence English news corpus from the Leipzig Corpora Collection[3] (Biemann et al., 2007) with the Stanford parser (de Marneffe et al., 2006) and used collapsed dependencies to extract features for words: each dependency triple (w_1, r, w_2) denoting a directed dependency of type r between words w_1 and w_2 results in a feature (r, w_2) characterizing w_1, and a feature (w_1, r) characterizing w_2. Words are thereby represented by the concatenation of the surface form and the POS as assigned by the parser. After counting the frequency of each feature for each word, we apply a significance measure (log-likelihood test (Dunning, 1993)), rank features per word according to their significance, and prune the data, keeping only the 300 most salient features per word. The similarity of two words is given by the number of their common features (which we will shortly illustrate with an example). The pruning operation greatly reduces run time at thesaurus construction, rendering memory reduction techniques like Goyal et al. (2012) unnecessary. Despite its simplicity and the basic count of feature overlap, we found this setting to be equal to or better than more complex weighting schemes in word similarity evaluations. Across all parts of speech, the DT contains five or more similar terms for a vocabulary of over 150 000 words.

To illustrate the DT, Table 1 shows the top three most similar words to the noun *paper*, together with the features which determine the similarities. Amongst their 300 most salient features as determined by the significance measure, *newspaper* and *paper* share 45, *book* and *paper* share 33, and *article* and *paper* share 28; these numbers constitute the terms' respective similarity scores.

The DT is used to expand the context and the sense definitions in the following way: For each content word (that is, adjectives, nouns, adverbs, and verbs) we retrieve the n most similar terms from the DT and add them to the textual representation. Since our overlap-based

[3] Available at `http://corpora.uni-leipzig.de/`; data for 229 languages and dialects is published.

term	score	shared features
newspaper|NN	45	told|VBD|-dobj column|NN|-prep|in local|JJ|amod editor|NN|-poss edition|NN|-prep|of editor|NN|-prep|of hometown|NN|nn industry|NN|-nn clips|NNS|-nn shredded|JJ|amod pick|VB|-dobj news|NNP|appos daily|JJ|amod writes|VBZ|-nsubj write|VB|-prep|for wrote|VBD|-prep|for wrote|VBD|-prep|in wrapped|VBN|-prep|in reading|VBG|-prep|in reading|VBG|-dobj read|VBD|-prep|in read|VBD|-dobj read|VBP|-prep|in read|VB|-dobj read|VB|-prep|in record|NN|prep|of article|NN|-prep|in reports|VBZ|-nsubj reported|VBD|-nsubj printed|VBN|amod printed|VBD|-nsubj printed|VBN|-prep|in published|VBN|-prep|in published|VBN|partmod published|VBD|-nsubj sunday|NNP|nn section|NN|-prep|of school|NN|nn saw|VBD|-prep|in ad|NN|-prep|in copy|NN|-prep|of page|NN|-prep|of pages|NNS|-prep|of morning|NN|nn story|NN|-prep|in
book|NN	33	recent|JJ|amod read|VB|-dobj read|VBD|-dobj reading|VBG|-dobj edition|NN|-prep|of printed|VBN|amod industry|NN|-nn described|VBN|-prep|in writing|VBG|-dobj wrote|VBD|-prep|in wrote|VBD|rcmod write|VB|-dobj written|VBN|rcmod written|VBN|-dobj wrote|VBD|-dobj pick|VB|-dobj photo|NN|nn co-author|NN|-prep|of co-authored|VBN|-dobj section|NN|-prep|of published|VBN|-dobj published|VBN|-nsubjpass published|VBD|-dobj published|VBN|partmod copy|NN|-prep|of buying|VBG|-dobj buy|VB|-dobj author|NN|-prep|of bag|NN|-nn bags|NNS|-nn page|NN|-prep|of pages|NNS|-prep|of titled|VBN|partmod
article|NN	28	authors|NNS|-prep|of original|JJ|amod notes|VBZ|-nsubj published|VBN|-dobj published|VBD|-dobj published|VBN|-nsubjpass published|VBN|partmod write|VB|-dobj wrote|VBD|rcmod wrote|VBD|-prep|in written|VBN|rcmod wrote|VBD|-dobj written|VBN|-dobj writing|VBG|-dobj reported|VBD|-nsubj describing|VBG|partmod described|VBN|-prep|in copy|NN|-prep|of said|VBD|-prep|in recent|JJ|amod read|VB|-dobj read|VB|-prep|in read|VBD|-dobj read|VBD|-prep|in reading|VBG|-dobj author|NN|-prep|of titled|VBN|partmod lancet|NNP|nn

Table 1: Illustration of a DT entry with features, showing the most similar terms to the noun *paper*.

approaches treat contexts and sense definitions as unordered bags of words, we do not need to take precautions with respect to the positions of words and expansions within the texts. The bags of words are filtered by removing occurrences of the disambiguation target. Then, we count the overlaps as usual between the expanded context and sense definitions. In our experiments we test $n = 10, 20, \ldots, 100$.

We had the intuition that the optimal number of expansions may depend on the part of speech of the word to be disambiguated, and perhaps also on the parts of speech of the words being expanded. Therefore, we parameterized our expansion procedure such that the part of speech of the target word determined the number of expansions, and also whether all words were expanded or only those of a certain part of speech.

3.2 Data sets

Data sets for WSD can generally be classified as *fine-grained* or *coarse-grained* according to the granularity of the sense inventory used for the annotations. Another common distinction is between the *all-words* task, in which the aim is to provide an annotation for every content word

in long running texts, and the *lexical sample* task, where several instances from the same small set of target words are annotated in (usually very short) contexts. We tested our systems on several coarse- and fine-grained data sets, and in both the all-words and lexical sample settings. However, most of our analysis will focus on the coarse-grained all-words scenario, as all-words provides a wider and more natural distribution of target words and senses, and because the fine sense distinctions of WordNet are considered a major obstacle to accurate WSD (Navigli, 2009). Additionally, as we discuss below, the fine-grained data sets available to us have various issues which render them unsuitable for comparisons with the state of the art.

Our coarse-grained data set is from the SemEval-2007 English all-words disambiguation task (Navigli et al., 2007). It consists of five non-fiction documents from various sources, where each of the 2269 content words (362 adjectives, 1108 nouns, 208 adverbs, and 591 verbs) has been annotated with clusters of WordNet 2.1 senses. For this data set only, we make a slight modification to our algorithm to account for this clustering: instead of choosing the WordNet sense with the highest overlap, we add up the overlap counts of each cluster's constituent senses, and then select the best cluster.

For our fine-grained experiments, we used the all-words and lexical sample tasks from Senseval-2 (Palmer et al., 2001; Kilgarriff, 2001) and Senseval-3 (Snyder and Palmer, 2004). With these data sets, however, several factors hinder direct comparison to previously published results. There are a number of errors in the gold standard annotations, and the methodology of the original task is different from what has subsequently become common. Specifically, not all of the target words have a corresponding entry in the sense inventory, and systems were originally expected to mark these "unassignable" senses as such. In the case of Senseval-2, the gold standard annotations were made using an unpublished (and now lost) version of WordNet. Subsequent researchers have adopted a variety of mutually incompatible methods for dealing with these issues. For our runs, we use Rada Mihalcea's WordNet 3.0 conversions of the corpora[4] and remove from consideration all "unassignable" target word instances. We do not fix the erroneous annotations, which means that even our baselines cannot achieve 100% coverage.

3.3 Baselines and measures

We use the evaluation metrics standard in word sense disambiguation research (Palmer et al., 2006; Navigli, 2009). Each disambiguation target receives a *score* equal to the probability the system assigned to the correct sense.[5] *Coverage* is the proportion of target word instances for which the system attempted a sense assignment, *precision* (P) is the sum of scores for the correct sense assignments divided by the number of target word instances for which the system made an attempt, and *recall* (R, also known as *accuracy*) is the sum of scores for the correct sense assignments divided by the number of target word instances. The *F-measure* is the harmonic mean of precision and recall: $F_1 = 2PR \div (P + R)$. Note that according to these definitions, $P \leq R$, and when coverage is 100%, $P = R = F_1$. In this paper we express all these measures as a percentage (i.e., in the range $[0, 100]$).

[4] `http://www.cse.unt.edu/~rada/downloads.html#sensevalsemcor`

[5] Where the probability is less than 1, this is mathematically equivalent to the average score which would have been obtained, over repeated runs, of choosing a sense at random to break any ties. It is effectively a backoff to a random sense baseline, ensuring 100% coverage even when there is no overlap.

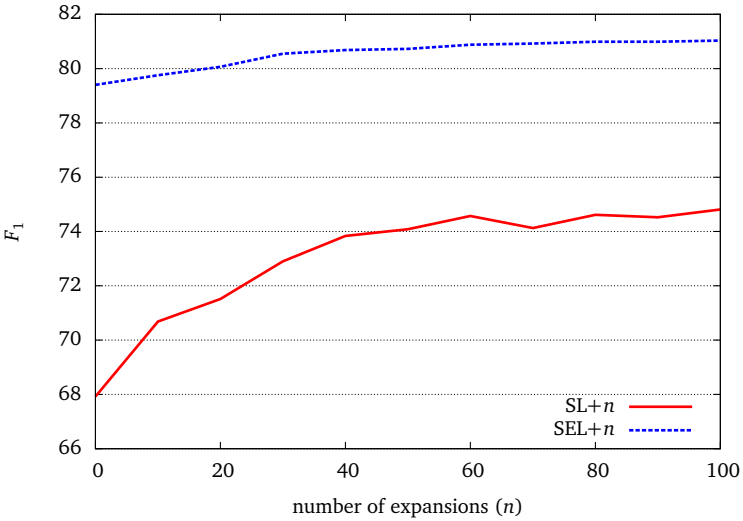


Figure 2: Results (F_1) on the SemEval-2007 corpus by number of lexical expansions

Our systems were compared against a computed random baseline which scores

$$P = R = F_1 = \frac{1}{|W|} \sum_{i=1}^{|W|} \frac{1}{|S(w_i)|},$$

where $W = \{w_1, w_2, \ldots\}$ is the set of target word instances in the corpus and $S(w_i)$ is the set of candidate senses for some target word w_i. This is equivalent to the score, averaged over repeated runs, of randomly choosing one of the candidate senses for each target word.

We also report accuracy of the most frequent sense (MFS) baseline, which always chooses the sense which occurs most frequently in SemCor (Mihalcea, 2008), a very large manually annotated corpus. Note that unlike our knowledge-based systems, MFS is a supervised baseline, and cannot actually be applied to the use cases for which our non-supervised systems are intended. Nonetheless, it is included here as it gives some idea of what accuracy could be achieved, at minimum, were one to go to the considerable expense of creating a manually tagged training corpus. Note that MFS is a notoriously difficult baseline to beat even for supervised systems.

3.4 Results

On the SemEval-2007 data set, the basic configuration of simplified Lesk (SL+0)—i.e., without any lexical expansions—achieves an overall F_1 of 67.92, which is already much better than the random baseline ($F_1 = 61.28$). When we tried adding a fixed number of lexical expansions to all content words, we observed that accuracy generally increased sublinearly with the number of

	part of speech				
system	**adj.**	**noun**	**adv.**	**verb**	**all**
MFS baseline	84.25	77.44	87.50	75.30	78.89
random baseline	68.54	61.96	69.15	52.81	61.28
SL+0	75.32	69.71	69.75	59.46	67.92
SL+100	82.18	76.31	78.85	66.07	74.81
SEL+0	87.19	81.52	74.87	**72.26**	79.40
SEL+100	**88.40**	**83.45**	80.29	72.25	**81.03**
TKB-UO	78.73	70.76	74.04	62.61	70.21
MII+ref	82.04	80.05	**82.21**	70.73	78.14
WN^{++}-DC	—	79.4	—	—	—

Table 2: Results (F_1) on the SemEval-2007 corpus by part of speech

expansions. The highest accuracy was obtained by using 100 expansions (the maximum number we tried); we denote this configuration SL+100. SL+100's F-measure of 74.81 represents a relative increase of more than 10% over SL+0. The simplified extended Lesk configuration also benefitted from lexical expansions, though the effect was less pronounced: the basic version without expansions (SEL+0) achieves $F_1 = 79.40$, and adding 100 lexical expansions (SEL+100) yields a relative performance increase of just over 2%, to $F_1 = 81.03$. As with simplified Lesk, accuracy increased sublinearly with the number of expansions. This effect is visualized in Figure 2, which plots the F-measure for the two algorithms according to the number of lexical expansions used.

Table 2 shows the F-measure of our two baselines (top), our algorithms (middle), and some state-of-the-art knowledge-based systems (bottom), broken down by target word part of speech. In each column the best result, excluding the supervised MFS baseline, is shown in boldface. TKB-UO (Anaya-Sánchez et al., 2007) was the best-performing knowledge-based system at the SemEval-2007 competition; it is a clustering-based system which uses WordNet 2.1 (but not its sense frequency information) as its sole source of knowledge. Among later knowledge-based systems using this data set, MII+ref (Li et al., 2010), a topic model approach, achieves the highest result we are aware of. This work maps sense descriptions and target word contexts to a topic distribution vector as sampled by LDA (Blei et al., 2003). WN^{++}-DC (Ponzetto and Navigli, 2010) uses an altogether different approach: it disambiguates nouns in a sentence by building a graph of candidate senses linked by semantic relations, and then for each target word selecting the sense with the highest vertex degree. When using semantic relations from WordNet alone the method achieves $F_1 = 74.5$, but when WordNet is enriched with additional semantic relations from an online encyclopedia performance increases to $F_1 = 79.4$. Note that, uniquely among the results in the table, WN^{++}-DC does not achieve full coverage ($P = 87.3$, $R = 72.7$).

POS-optimized results. We also tried using different expansion strategies for target words of different parts of speech: for each target word POS, we tried expanding only adjectives, only nouns, etc., and tried each of these scenarios for the same eleven values of n as previously. Because this procedure involved tuning on the test data, we do not include the results for comparison in Table 2. However, they are interesting as they give an upper bound on per-

system	Senseval-2 lexical sample	Senseval-2 all-words	Senseval-3 all-words
MFS baseline	41.56	65.36	65.63
random baseline	15.46	39.54	32.89
SL+0	17.10	39.02	35.41
SL+100	20.92	45.69	37.17
SEL+0	28.60	54.22	48.76
SEL+30	32.72	57.77	53.09

Table 3: Results (F_1) on the Senseval-2 and -3 corpora

formance for the case where the expansions-per-POS parameters are optimized on a set of manually annotated training examples—that is, a mildly supervised variant of our otherwise knowledge-based algorithms.

For simplified Lesk, we found that accuracy for nouns, adverbs, and verbs remained highest when all content words were given 100 expansions, but adjectives fared better when all content words were given only 60 expansions. With this configuration we achieve an overall F_1 of 74.94. The best simplified extended Lesk configuration achieves $F_1 = 81.27$ when for adjectives we apply 20 expansions to all content words; for nouns, 60 expansions to all content words; for adverbs, 80 expansions to all content words; and for verbs, 30 expansions to adverbs only. That verbs benefit from adverb expansions is not surprising, given that the latter often serve to modify the former. Why the optimal *number* of expansions should vary with the target word part of speech is not as clear. In any case, the extra performance gains from POS expansion optimization were quite small, not exceeding a quarter of a percentage point over the non-optimized versions.

Fine-grained results. As with the coarse-grained task, we found that using lexical expansions resulted in an improvement in accuracy in the fine-grained tasks. However, in this setting we did not observe the same continuously improving accuracy from using more and more expansions; in all but one case, adding expansions helped to a point, after which accuracy started to decrease. This effect was particularly noticeable with simplified extended Lesk, where peak accuracy was achieved with around 30 expansions. For simplified Lesk, the optimum was less stable across the corpora, ranging from 60 to 100 expansions. We believe that this is because the expanded terms provided by the DT reflect broad conceptual relations which, taken in aggregate, do not precisely map to the narrow sense distinctions of the sense inventory. This is not a problem when we stick to the first highly salient expansions provided by the DT, but beyond this the conceptual relations become too tenuous and fuzzy to facilitate disambiguation.

Table 3 shows the results of our systems and baselines on the Senseval-2 lexical sample and all-words tasks and the Senseval-3 all-words task. For simplified extended Lesk we show the results of using 30 expansions (SEL+30); as simplified Lesk had no consistent peak accuracy we stick with 100 expansions (SL+100). The results, while quite expectedly lower than the coarse-grained scores in absolute terms, nonetheless validate the utility of our approach in fine-grained tasks. Not only does the use of expansions significantly increase the accuracy, but in the case of the Senseval-2 corpora, the relative increase is much higher than that of the coarse-grained tasks. For SL+100, the relative improvements over the unexpanded algorithms

for the lexical sample and all-words data sets are 22.3% and 17.1%, respectively, and for SEL+100 they are 14.4% and 6.5%, respectively.

4 Discussion

In this section, we discuss our results and put them in the perspective of applicability of WSD systems. Our lexical expansion mechanism leads to a relative improvement of up to 22% in the fine-grained evaluation and 10% in the coarse-grained evaluation for the "simple" setup. This is achieved by merely adding lexical items to the representation of the sense description and context, and without changing the algorithm. Especially in situations where there exists a reasonably coarse-grained MRD for the language or domain, this is a major improvement over previous approaches on applications where one is not in the comfortable situation of having sense frequency information. In our opinion, this scenario has been neglected in the past, despite occurring in practice much more often than the case where one has access to a rich LSR, let alone sufficient training data for supervised disambiguation.

The expansions from distributional similarity are complementary to those coming from richer knowledge resources, as our results for fitting simplified extended Lesk with DT expansions show: even in the situation where a richer lexical resource allows for bridging the lexical gap via descriptions from related senses, we still see an additional relative improvement of 2% to 14% when comparing the F-measure of the SEL+n system against the SEL+0 baseline. Not only does this system outperform all previous approaches to coarse-grained WSD without MFS backoff, it is also able to outperform the MFS baseline itself, both generally and for certain parts of speech.

We emphasize that while the DT uses additional text data for computing the similarity scores used in the lexical expansion step, the overall system is purely knowledge-based because it is not trained on sense-labelled examples; the DT similarities are computed on the basis of an automatically parsed but otherwise unannotated corpus. This marks an important difference from the system described in Navigli and Velardi (2005) which, although it also uses collocations extracted from large corpora, avails itself of manual sense annotations wherever possible.

While the comparison of results to other methods on the same coarse-grained data sets suggests that lexical expansion using a distributional thesaurus leads to more precise disambiguation systems than word or topic vectors, our point is somewhat different: Realizing lexical expansions and thus explicitly generating associated terms to a textual representation opens up a new way of thinking about bridging lexical gaps and semantic matching of similar meaning. In light of the fact that distributional similarity (Lin, 1998) and overlap-based approaches to WSD (Lesk, 1986) have existed for a long time now, it is somewhat surprising that this avenue had not been explored earlier.

4.1 Error analysis

In order to better understand where and how our system is succeeding and failing, we now present an error analysis of the results, both in aggregate and for some individual cases. To begin, we computed a confusion matrix showing the percentage of the 2269 SemEval-2007 target word instances for which the SL+0 and SL+100 algorithms made a correct disambiguation, made an incorrect disambiguation, or failed to make an assignment at all without resorting to the random choice backoff (see Table 4). Table 5 shows the same confusion matrix for the SEL+0

		SL+100			
		unassigned	**incorrect**	**correct**	**total**
SL+0	**unassigned**	0.2	8.1	9.7	18.0
	incorrect	0.1	14.1	6.2	20.4
	correct	0.0	2.8	58.8	61.6
	total	0.4	25.0	74.7	100.0

Table 4: Confusion matrix for SL+0 and SL+100

		SEL+100			
		unassigned	**incorrect**	**correct**	**total**
SEL+0	**unassigned**	0.1	3.5	4.0	7.6
	incorrect	0.0	14.9	2.8	17.7
	correct	0.0	1.9	72.9	74.7
	total	0.1	20.3	79.6	100.0

Table 5: Confusion matrix for SEL+0 and SEL+100

and SEL+100 algorithms.[6] As can be seen, the pattern of contingencies is similar. Because of the sheer size of the expanded sense descriptions and contexts with this task, however, in the following analysis we stick to the simplified Lesk scenario.

As we hypothesized, using lexical expansions successfully bridges the lexical gap: whereas the basic simplified Lesk was able to make a sense assignment (be it correct or incorrect) in only 82.0% of cases, SL+100 could do so 99.6% of the time. SL+100 was able to correctly disambiguate over half of all the target words for which SL+0 failed to make any sense assignment. This contingency—some 9.7% of all instances—accounts for the majority of SL+100's improvement over SL+0. However, in 6.2% of cases SL+100's improvement resulted from successfully revising an incorrect answer of SL+0. We randomly selected ten of these cases and found that in all of them, all the overlaps for SL+0 were from a small number of non-content words (*the*, *of*, *in*, etc.), with the chosen sense achieving only one or two more overlaps than the runners-up; thus, the lexical gap is still at fault here. By contrast, the expanded sense definitions and contexts used by SL+100 for these cases always contained dozens of overlapping content words, and the overlap count for the chosen sense was markedly higher than for the runners-up.

What is also interesting to consider is the 0.2% of cases where both algorithms neglected to make a sense assignment, apparently signifying SL+100's failure to bridge the lexical gap. We manually examined all of these instances and found that for all but one, the systems failed to disambiguate the target words because the sentences containing them were extremely short, usually with no other content words apart from the target word. It is unlikely that any knowledge-based algorithm restricting itself to sentential context could succeed in such cases, and no reasonable number of lexical expansions is likely to help. Our choice to use sentential context was motivated by simplicity and expediency; a more refined WSD algorithm could, of course, using a sliding or dynamically sized context window and thereby avoid this problem.

[6] Totals in both tables may not add up exactly due to rounding.

The remaining case was a sentence of normal length where SL+0 found no overlapping content words between the definition and the context, but SL+100 produced a two-way tie between two of the clusters, one of which was the correct one.

It is also of interest to know why SL+0 was able to correctly disambiguate some words which SL+100 could not; these represent 2.8% of the instances. Again, we drew a random sample of these instances, and observed that in all of them, the only overlaps found by SL+0 were for non-content words; the fact that it happened to choose the correct sense cluster can therefore be chalked up to chance.

Though it has been relatively easy to identify the reasons behind SL+100's correct assignments, and behind its failures to make any assignment at all, it is not so easy to deduce the causes of its incorrect assignments. We observe that the system had disproportionate difficulties with verbs, which constitute 35% of the incorrect disambiguations but only 26% of all target words in the corpus. Particularly troublesome were verbs such as *be*, *go*, *have*, and *do*, which are often used as auxiliaries. On their own they contribute little or no semantic information to the sentence, and their dictionary definitions tend to explain their grammatical function, so there is little opportunity for meaningful lexical or conceptual overlaps. A related problem was observed for adverbs and adjectives: the problematic cases here were often generic terms of restriction, intensification, or contrast (e.g., *different*, *just*, *only*, *so*) which are used in a wide variety of semantic contexts and whose dictionary definitions focus on usage, or else constitute concise rephrasings using equally generic terms. Purely definition-based disambiguation approaches are unlikely to help in any of these cases; an accurate knowledge-based approach would probably need to be aware and make use of information beyond the lexical-semantic level, such as verb frames and semantic roles, or incorporate the grammatical structure around the target word for matching.

Conclusion and further work

We have proposed a new method for word sense disambiguation based on word overlap between sense descriptions and the target word context. Our method uses lexical expansions from a distributional thesaurus, which is computed over dependency-context similarities over a large background corpus. We found that applying our conceptually simple extension to two traditional knowledge-based methods successfully bridged the lexical gap, resulting in performance gains exceeding that of state-of-the-art knowledge-based systems that do not make use of sense frequency information, and approaching or even exceeding the MFS baseline. The concept of lexical expansion is a promising avenue to enrich classic, word-based NLP algorithms with additional lexical material. The intuitions of overlap-based approaches are thereby complemented by a method that makes associations explicit and bridges the lexical gaps for semantically similar contexts that are expressed in a different wording.

There are a number of ways how our method could be improved. First of all, since a DT is static and thus not dependent on the context, it generates spurious expansions, such as the similar terms for *charge* in Figure 1, which is obviously dominantly used in its "criminal indictment" sense in the background corpus. At best, these expansions, which implicitly capture the sense distribution in the background corpus, result in less overlap with the correct sense description—but they might well result in assigning incorrect senses. A straightforward improvement would alter the lexical expansion mechanism as to be sensitive to the context—something that is captured, for example, by LDA sampling (Blei et al., 2003). A further extension would be to have the number of lexical expansions depend on the DT similarity score (be it static or

contextualized) instead of the fixed number we used here.

In the future, we would like to examine the interplay of lexical expansion methods in WSD systems with richer knowledge resources (e.g., Navigli and Ponzetto (2010); Gurevych et al. (2012)) and apply our approach to other languages with fewer lexical resources. Also, it seems promising to apply lexical expansion techniques to text similarity, text segmentation, machine translation, and semantic indexing.

Acknowledgments

We thank Richard Steuer for computing and providing us access to the distributional thesaurus.

This work has been supported by the Hessian research excellence program *Landes-Offensive zur Entwicklung Wissenschaftlich-ökonomischer Exzellenz (LOEWE)* as part of the research center *Digital Humanities*, and also by the Volkswagen Foundation as part of the Lichtenberg Professorship Program under grant № I/82806.

References

Agirre, E., Martínez, D., de Lacalle, O. L., and Soroa, A. (2006). Evaluating and optimizing the parameters of an unsupervised graph-based WSD algorithm. In *Proceedings of the First Workshop on Graph Based Methods for Natural Language Processing (TextGraphs-1)*, pages 89–96. (New York, NY, USA).

Anaya-Sánchez, H., Pons-Porrata, A., and Berlanga-Llavori, R. (2007). TKB-UO: Using sense clustering for WSD. In *Proceedings of the Fourth International Workshop on Semantic Evaluations (SemEval-2007)*, pages 322–325. (Prague, Czech Republic).

Banerjee, S. and Pedersen, T. (2002). An adapted Lesk algorithm for word sense disambiguation using WordNet. In *Proceedings of the Third International Conference on Intelligent Text Processing and Computational Linguistics (CICLing 2012)*. (New Delhi, India).

Bär, D., Biemann, C., Gurevych, I., and Zesch, T. (2012). UKP: Computing semantic textual similarity by combining multiple content similarity measures. In *Proceedings of the Sixth International Workshop on Semantic Evaluation (SemEval-2012)*, pages 435–440. (Montreal, Canada).

Biemann, C. (2012). Creating a system for lexical substitutions from scratch using crowdsourcing. *Language Resources and Evaluation*. DOI 10.1007/s10579-012-9180-5.

Biemann, C., Heyer, G., Quasthoff, U., and Richter, M. (2007). The Leipzig Corpora Collection: Monolingual corpora of standard size. In *Proceedings of the Corpus Linguistics Conference (CL2007)*. (Birmingham, United Kingdom).

Blei, D. M., Ng, A. Y., and Jordan, M. I. (2003). Latent Dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022.

Cai, J. F., Lee, W. S., and Teh, Y. W. (2007). NUS-ML: Improving word sense disambiguation using topic features. In *Proceedings of the Fourth International Workshop on Semantic Evaluations (SemEval-2007)*, pages 249–252. (Prague, Czech Republic).

de Marneffe, M.-C., MacCartney, B., and Manning, C. D. (2006). Generating typed dependency parses from phrase structure parses. In *Proceedings of the Fifth International Conference on Language Resources and Evaluations (LREC 2006)*. (Genoa, Italy).

de Saussure, F. (1916). *Cours de linguistique générale*. Librairie Payot & Cie, Paris.

Dunning, T. (1993). Accurate methods for the statistics of surprise and coincidence. *Computational Linguistics*, 19(1):61–74.

Fellbaum, C., editor (1998). *WordNet: An Electronic Lexical Database*. MIT Press, Cambridge, MA.

Gliozzo, A., Giuliano, C., and Strapparava, C. (2005). Domain kernels for word sense disambiguation. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics (ACL '05)*, pages 403–410. (Ann Arbor, MI, USA).

Goyal, A., Daumé III, H., and Cormode, G. (2012). Sketch algorithms for estimating point queries in NLP. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP–CONLL 2012)*, pages 1093–1103. (Jeju Island, South Korea).

Gurevych, I., Eckle-Kohler, J., Hartmann, S., Matuschek, M., Meyer, C. M., and Wirth, C. (2012). Uby – A large-scale unified lexical-semantic resource. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics (EACL 2012)*, pages 580–590. (Avignon, France).

Henrich, V., Hinrichs, E., and Vodolazova, T. (2012). WebCAGe – A Web-harvested corpus annotated with GermaNet senses. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics (EACL 2012)*, pages 387–396. (Avignon, France).

Kilgarriff, A. (2001). English lexical sample task description. In *Proceedings of Senseval-2: Second International Workshop on Evaluating Word Sense Disambiguation Systems*, pages 17–20. (Toulouse, France).

Kilgarriff, A. and Rosenzweig, J. (2000). Framework and results for English SENSEVAL. *Computers and the Humanities*, 34:15–48.

Lesk, M. (1986). Automatic sense disambiguation using machine readable dictionaries: How to tell a pine cone from a ice cream cone. In *Proceedings of the Fifth Annual International Conference of Systems Documentation (SIGDOC '86)*, pages 24–26. (Toronto, Canada).

Li, L., Roth, B., and Sporleder, C. (2010). Topic models for word sense disambiguation and token-based idiom detection. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics (ACL '10)*, pages 1138–1147. (Uppsala, Sweden).

Lin, D. (1998). Automatic retrieval and clustering of similar words. In *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics (ACL '98) and the 17th International Conference on Computational Linguistics (COLING 1998)*, volume 2, pages 768–774. (Montreal, Canada).

Martínez, D., López de Lacalle, O., and Agirre, E. (2008). On the use of automatically acquired examples for all-nouns word sense disambiguation. *Journal of Artificial Intelligence Research*, 33(1):79–107.

Mihalcea, R. (2008). Semcor 3.0. `http://lit.csci.unt.edu/~rada/downloads/semcor/semcor3.0.tar.gz`.

Mihalcea, R. and Chklovski, T. (2003). Open Mind Word Expert: Creating large annotated data collections with Web users' help. In *Proceedings of Fourth International Workshop on Linguistically Interpreted Corpora (LINC-03)*. (Budapest, Hungary).

Navigli, R. (2009). Word sense disambiguation: A survey. *ACM Computing Surveys*, 41:10:1–10:69.

Navigli, R., Litkowski, K. C., and Hargraves, O. (2007). SemEval-2007 Task 07: Coarse-grained English All-words Task. In *Proceedings of the Fourth International Workshop on Semantic Evaluations (SemEval-2007)*, pages 30–35. (Prague, Czech Republic).

Navigli, R. and Ponzetto, S. P. (2010). BabelNet: Building a very large multilingual semantic network. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics (ACL '10)*, pages 216–225. (Uppsala, Sweden).

Navigli, R. and Velardi, P. (2005). Structural semantic interconnections: A knowledge-based approach to word sense disambiguation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27:1075–1086.

Palmer, M., Fellbaum, C., Cotton, S., Delfs, L., and Dang, H. T. (2001). English tasks: All-words and verb lexical sample. In *Proceedings of Senseval-2: Second International Workshop on Evaluating Word Sense Disambiguation Systems*, pages 21–24. (Toulouse, France).

Palmer, M., Ng, H. T., and Dang, H. T. (2006). Evaluation of WSD systems. In Agirre, E. and Edmonds, P., editors, *Word Sense Disambiguation: Algorithms and Applications*, volume 33 of *Text, Speech, and Language Technology*. Springer.

Ponzetto, S. P. and Navigli, R. (2010). Knowledge-rich word sense disambiguation rivaling supervised systems. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics (ACL '10)*, pages 1522–1531. (Uppsala, Sweden).

Rapp, R. (2004). A freely available automatically generated thesaurus of related words. In *Proceedings of the Fourth International Conference on Language Resources and Evaluations (LREC 2004)*, pages 395–398. (Lisbon, Portugal).

Snyder, B. and Palmer, M. (2004). The English all-words task. In Mihalcea, R. and Edmonds, P., editors, *Senseval-3: Third International Workshop on the Evaluation of Systems for the Semantic Analysis of Text*, pages 41–43. (Barcelona, Spain).

Tugwell, D. and Kilgarriff, A. (2001). WASP-Bench: A lexicographic tool supporting word sense disambiguation. In *Proceedings of Senseval-2: Second International Workshop on Evaluating Word Sense Disambiguation Systems*, pages 151–154. (Toulouse, France).

Vasilescu, F., Langlais, P., and Lapalme, G. (2004). Evaluating variants of the Lesk approach for disambiguating words. In *Proceedings of the Fourth International Conference on Language Resources and Evaluations (LREC 2004)*, pages 633–636. (Lisbon, Portugal).

Revising the Compositional Method for Terminology Acquisition from Comparable Corpora

Emmanuel Morin *Béatrice Daille*
Université de Nantes, LINA UMR CNRS 6241
2, rue de la Houssinière, BP 92208
F-44322 Nantes cedex 03
{emmanuel.morin,beatrice.daille}@univ-nantes.fr

ABSTRACT
In this paper, we present a new method that improves the alignment of equivalent terms monolingually acquired from bilingual comparable corpora: the Compositional Method with Context-Based Projection (CMCBP). Our overall objective is to identify and to translate high specialized terminology made up of multi-word terms acquired from comparable corpora. Our evaluation in the medical domain and for two pairs of languages demonstrates that CMCBP outperforms the state-of-art compositional approach commonly used for translationally equivalent multi-word term discovery from comparable corpora.

KEYWORDS: Comparable corpora, bilingual lexicon, compositionality, multi-word term, context information.

1 Introduction

The automatic compilation of bilingual dictionaries has received considerable attention in recent years for *language for special purposes (LSP)* (especially coming from scientific domains). LSP is characterised by the small amount of available textual data compared with general language, and a high proportion of specialised terms which are not be found in general language monolingual or bilingual dictionaries. For LSP, a specialised term could be either a single-word term (SWT) or a multi-word term (MWT), the latter being highly productive (Sag et al., 2002). A term is a lexical unit which represents a concept within a domain. As an example, in the medical domain, *cancer* is an SWT, *breast cancer* is an MWT.

Comparable corpora that are sets of texts in two or more languages without being translations of each other, seem to be the right solution to solve the textual scarcity of LSP: as monolingual productions, they are authentic texts, and the babel web ensures that there is a sufficient number of multilingual documents. The comparability of the corpus should be ensured by using various shared characteristics across languages that are checked during the compilation phase (McEnery, 2007). For LSP, the domain and sub-domain are requested as well as the communicative settings and the textual genre to identify reliable translations (Bowker and Pearson, 2002).

To build highly-specialised terminologies, the terms are first of all extracted monolingually from the comparable corpus. To collect close candidate terms across languages, it is necessary to use a term extraction program that applies the same method in the source and in the target languages. The translation of MWTs is the main need as they constitute around 80% of the domain-specific terms. See Nakagawa and Mori (2003) for the Japanese language.

Proceedings of COLING 2012: Technical Papers, pages 1797–1810,
COLING 2012, Mumbai, December 2012.

Our goal is to find the right translation for a source MWT in the set of MWT candidates in the target language. The simplest method assumes that the right translation of an MWT could be obtained by translating each component individually thanks to a general dictionary, by generating all the combinations of word positions, and then filtering the translated expressions using either the list of target MWTs (Morin and Daille, 2010), the target corpus (Robitaille et al., 2006) or the web (Grefenstette, 1999). This method is limited to the subset of MWTs that share the same compositional property - 48.7% were reported by Baldwin and Tanaka (2004) for English/Japanese N N compounds. However, even if the MWT is characterised by the compositional property, the translation is not found when some words, which are part of the MWTs, do not belong to the general bilingual dictionary or when the translated combinations do not exist or have not been extracted by the term extraction program in the target language.

Within this context, we propose to improve the compositional approach by using context information collected from LSP comparable corpora when one or several of the components, part of the MWTs, are not found in the dictionary. We demonstrate that the use of context information when performing terminology translation helped us to learn a significant number of additional correct lexical entries that could not be identified by the compositional method.

The remainder of this paper is organised as follows: Section 2 shows the intricate problems with the translations of MWTs. Section 3 presents the compositional method used for the automatic translation of MWTs. Section 4 introduces CMCBP that takes advantage of the context information to improve the compositional method. Section 5 describes the linguistic resources and the open-terminology extraction tool used for our experiments. Section 6 evaluates the influence of CMCBP on the quality of bilingual terminology extraction through experiments involving French as a source language, and English and German as target languages. Section 7 discusses works related to this study. Finally, Section 8 presents our conclusions.

2 Translation of MWTs

If MWTs are less polysemous (Savary and Jacquemin, 2003) and more representative (Nomura and M., 1989; Nakagawa and Mori, 2003) of domain specialities than SWTs, pinpointing their translations poses specific problems that are well-known, such as fertility, non-compositionality, or term variation[1]:

Fertility is known as a problem of difference of length between the source and the target MWT (Brown et al., 1993): for instance, the German SWT *axilladissektion* (1 content word) is translated into English by the MWT *axillary dissection* (two content words); the French MWT *dépistage du cancer du sein* (three content words) is translated into English by the MWT *breast screening* (two content words).

Non-compositionality is illustrated when the target MWT is not typically composed of the translation of its parts (Melamed, 2001). For instance the French MWT *curage axillaire* is translated into the English language as *axillary dissection* whereas the English word *dissection* is not the translation of the French word *curage*. Baldwin and Tanaka (2004) report that at least 50% of the Japanese N N compounds are not translated through a compositional strategy into English.

[1] The French/English/German examples in this paper are extracted from the specialized medical comparable corpus described in Section 5.

Term variation refers to an MWT that appears in texts in different forms reflecting either graphical, syntactic, morphological or semantic differences: for example, the French MWTs *cancer du sein* and *cancer mammaire* are both translated by the same English MWT *breast cancer*. Source and target MWTs can appear in different syntactic structures. For example, the French MWT *prolifération tumorale* of N A pattern is translated by the English MWT *tumour proliferation* of N N pattern, where the French adjective *tumorale* is linked through morphological derivation to the English noun *tumour*. The term variations could also involve paradigmatic variation when one element of the MWT is substituted by a synonym or a hypernym such as *tumour size* → *diameter tumour* in the source language and not in the target language such as *taille tumorale* (lit. 'tumour size').

It is quite difficult to design a general framework that can address all these problems simultaneously (Robitaille et al., 2006) and for any language. The non-compositionality has to be solved during the translation process as it involves an MWT and its translation. The term variant problem is generally handled at the monolingual level during the term extraction task. This is done in two steps: term variant extraction and term variant grouping. A sophisticated variant recognition and conflation program will handle several types of variants: graphical, but also morphological and syntactic variation, ideally paradigmatic variants and acronyms. Using a term variant program allows us to cluster a set of term-like sequences reflecting base or variant forms. This clustering could be interpreted as a terminology normalization in the same way as lemmatisation at the morphological level. Handling term variation could indirectly solve part of the fertility problem using the syntactic variant of MWTs: as an example, in French, the term *dépistage du cancer du sein* (lit. *breast cancer screening*) could be collected as a syntactic variant of the term *dépistage du sein* (lit. *breast screening*) and thus provides a word-to-word translation. In German, the fertility problem could be solved by establishing an equivalence relation between a morphological compound of the type $N_1|N_2$ where | is the concatenation operator, and a syntagmatic compound of N_1N_2 pattern: the noun *axilladissektion* that is morphologically analyzed as *axilla|dissektion* will be a variant of the MWT *axilläre dissektion*.

The compositional method with context-based projection that we introduce in Section 4 will take into account the non-compositionality and term variation problems and indirectly the fertility problem through the term variant and the German compound splitting treatments.

3 Compositional Approach

Compositionality is defined as the property where *"the meaning of the whole is a function of the meaning of the parts*" (Keenan and Faltz, 1985, p. 24-25): a *frying pan* is indeed a *pan* used for *frying*. The implementation of the principle of translation compositionality from a comparable corpus relies on the following steps (Grefenstette, 1999; Tanaka, 2002; Robitaille et al., 2006):

Translation of the source MWT For an MWT of the source language to be translated, each component of the MWT is translated by looking it up in a dictionary. The lexical form is examined without checking the part-of-speech (POS). For example, for the French MWT *examen clinique* (*clinical examination*), there are six English translations for *examen* (*consideration*/N, *examen*/N, *examination*/N, *inspection*/N, *review*/N, *test*/N) and two translations for *clinique* (*clinic*/N, and *clinical*/A).

Generation of the candidate translations All possible mappings are constructed regardless of word order with a total of $O(\prod_{i=1}^{p} t_i n!)$ possible mappings (where t_i is the number of translations of the content word i, and n the number of content words). In the above example, 24 combinations are obtained. The number of generated translations can be reduced using MWT POS patterns in the source and the target languages. For instance, Tanaka and Baldwin (2003) defined the following templates to filter translation candidates: N_1 N_2 Japanese structure is translated by N_1 N_2 (33.2% of the cases), A_1 N_2 (28.4%), N_2 of (the) N_1 (4.4%) English structures.

Selection of the candidate translations From the set of translation candidates, the most likely translations are selected according to term frequency in the target language. In the above example, the translations are MWTs of the target language identified by the terminology extraction system.

4 Compositional Method with Context-Based Projection (CMCBP)

The compositional approach that finds translations of multi-word terms is easy to implement, but it fails when:

1. At least one element of an MWT is not found in the bilingual dictionary and thus cannot be translated.
2. The translated combination is valid but is not provided by the term extraction program for the target language. One explanation could be that the target MWT does not occur in the comparable corpus, or the source concept occurs in the target corpora but under a non terminology-like form, or an error during the preprocessing of the corpora induces that the terminology extraction program misses the MWT.
3. The translated combination is not valid. One of the MWT translation problems has been encountered (see Section 2).

When there is no translation candidates for an MWT, a first solution would be to find its synonyms in the source language. Similar words are predicted by Pekar et al. (2006) for low-frequency words and by Sharoff et al. (2009) for wrong translations. CMCBP that deals with term variants performs a clustering of synonymic terms. The translations that are proposed are for the set of synonymic term variants.

CMCBP is designed to identify MWT translations in a comparable corpus on a large scale and is able to solve points 1 and 3. It includes the use of the context of the words (which are parts of the MWT to be translated) when the compositional approach fails. We refer to the two monolingual parts of the comparable corpus as the source and target corpus. CMCBP uses four steps:

Computing the context of the MWT For an MWT or a morphological compound in the source corpus defined as $C_{s1}C_{s2}\cdots C_{sk}$ to be translated (where k is the number of content words or autonomous morphemes), we look up each component C_{si} in the bilingual dictionary. When a component is not found in the bilingual dictionary, we replace it by co-occurrence information. We compute the co-occurrence information between a component C_{si} and the words that co-occur in a window of w words around C_{si} from

the source corpus. Mutual information or Likelihood-ratio are good measures of the co-occurrence relationship between 2 words. The co-occurrence information is expressed with a vector representation called context vector (V_{si}). As an example, let us consider the French MWT *antécédent familial* ($C_{s1}C_{s2}$). If the first component *antécédent* (C_{s1}) is not found in the bilingual dictionary then this component is replaced by its vector context: V_{s1} (see Figure 1).

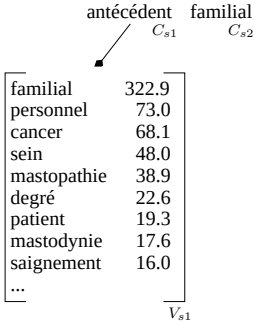


Figure 1: Computing context in the source corpus

Transfer of the MWT At this level, we are able to identify two situations depending on whether or not the components of the MWT are translated:

1. If the component C_{si} is found in the dictionary, we compute the co-occurrence information of each translation in the target corpus and store it in a context vector: V'_{si}.
2. If the component C_{si} is not found in the dictionary, we use the context vector of the source corpus V_{si}. The elements of V_{si} are projected into the target corpus using the bilingual dictionary and the transferred context vector becomes: V'_{si}. If the bilingual dictionary provides several translations for an element, all of them are used but the different translations are weighted according to their frequency in the target language. If an element is not found in the bilingual dictionary it is discarded.

In the previous example, if we find two English translations for the component *familial* (C_{s2}) such as *familial* and *family* in the target language then we obtain two context vectors: V'_{s2_1} and V'_{s2_2} (see Figure 2).

Generation of candidate translations Each MWT of the target language, for which each component C_{ti} is described by its context vector V_{ti}, is then compared to the transferred MWT through a similarity measure such as Cosine or Weighted Jaccard. For an MWT composed of two context vectors V_{t1} and V_{t2} in the target language and a transferred MWT composed of two context vectors V'_{s1} and V'_{s2}, two pairs of similarity scores corresponding to the possible mappings are computed: $sim(V_{t1}, V'_{s1})$ with $sim(V_{t2}, V'_{s2})$,

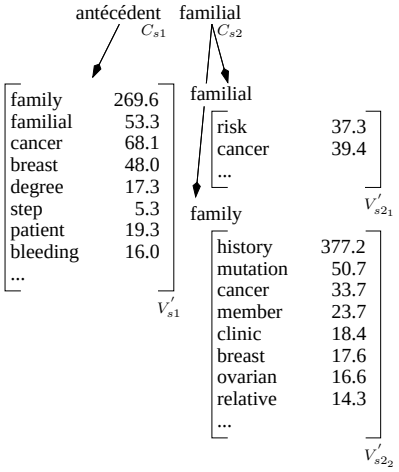


Figure 2: Projection in the target corpus

and $sim(V_{t1}, V'_{s2})$ with $sim(V_{t2}, V'_{s1})$. The combination score for each pair is then defined as the geometric mean of each similarity score: $\sqrt{sim(V_{t1}, V'_{s1}).sim(V_{t2}, V'_{s2})}$ and $\sqrt{sim(V_{t1}, V'_{s2}).sim(V_{t2}, V'_{s1})}$. Figure 3 illustrates this comparison for the previous example.

Ranking of candidate translations We rank the candidate translations in decreasing order of their combination score (see Figure 4).

5 Resources

In this section, we describe the different resources used for our experiments: the comparable corpus, the bilingual dictionary, and the multi-word term test set.

5.1 Comparable Corpora

The documents comprising the specialised comparable corpora were taken from the medical domain within the sub-domain of 'breast cancer'. These documents have been automatically selected from scientific paper websites where the title or the keywords of the articles contain the MWT 'breast cancer' in English, 'cancer du sein' in French and 'brustkrebs' in German. The compilation of the comparable corpus fulfils the requirements of an LSP comparable corpus: domain, sub-domain, communicative settings (experts-to-experts) and textual genre are common characteristics across languages. In this way, we collected 118 documents in English, 130 in French and 103 in German (about 530,000 words for English and French languages and 220,000 words for German language).

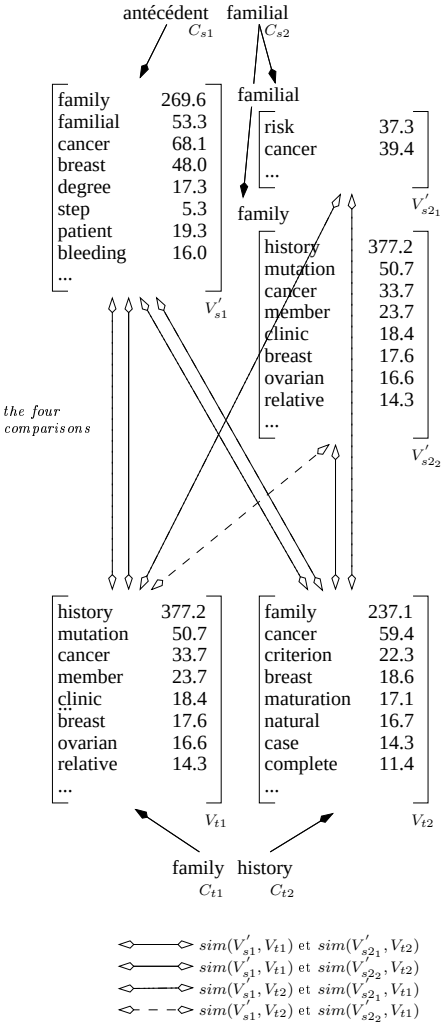


Figure 3: Comparison between the MWT to be translated and an MWT of the target corpus

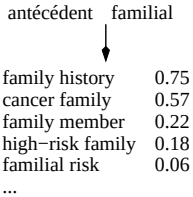


Figure 4: Rank list of candidate translations

5.2 Bilingual Dictionary

The bilingual dictionaries used in our experiments are the French/English dictionary ELRA-M0033 and the French/German dictionary ELRA-M0034 available from the ELRA catalogue[2]. The French/English dictionary contains 243,539 translations and the French/German dictionary 170,967 translations. These are two general language dictionaries which contain only a few terms related to the medical domain.

5.3 Multi-Word Term Test Set

Terms are extracted monolingually from the comparable corpora. To collect close candidate terms across languages, it is necessary to use a term extraction program that is multilingually designed. We choose the TTC TermSuite (Rocheteau and Daille, 2011)[3] that applies the same term extraction method to several languages including French, German and English. TermSuite first normalises the texts through the following linguistic pre-processing steps: tokenisation, part-of-speech tagging and lemmatisation using TreeTagger (Schmid, 1995). TermSuite then extracts SWTs and MWTs whose syntactic patterns correspond either to a canonical or a variation structure. The patterns are expressed using MULTEXT part-of-speech tags and are provided for each language. The main patterns, whatever the language is, are N and A for SWTs. The main patterns of MWTs are for:

- **French** N N: *ganglion sentinelle* (*sentinel lymph node*); N Sp N: *cancer du sein* (*breast cancer*); N A: *curage axillaire* (*axillary dissection*);
- **English** N N: *breast cancer*; A N: *far therapy*; N Sp N: ;
- **German** A N: *thromboembolischer vorfall* (*thromboembolic incident*); N Sp N: *patientin mit mammakarzinom* (*patient with breast cancer*); N D : g N: *erfahrung der früherkennung* (*experience of early detection*).

The variants handled for MWTs are graphical, morphological, and syntactic. Both SWTs and MWTs accept variants but some are more likely to concern one main type such syntactic variants for MWTs. TermSuite defines a morphological variant as a morphological modification of one of the components of the MWT, and a syntactical variant as the adding of another word at the frontier or inside the MWT. For example, in the French part of the comparable corpus, the MWT candidate *cancer du sein* (*breast cancer*) appears in the following forms where shared items are numbered with the same values.

[2] http://www.elra.info/
[3] http://code.google.com/p/ttc-project

- **base form** of N_1 $S_1 p$ N_2 pattern: *cancer du sein* (*breast cancer*);
- **inflexional variant**: *cancers du sein* (*breast cancers*);
- **syntactic variant** (insertion inside the base form of a modifier): N_1 A $S_1 p$ N_2 *cancer primitif du sein* (*primary breast cancer*);
- **syntactic variant** (expansion coordination of base form): N_1 Sp N $S_1 p$ N_2 *cancer des ovaires et du sein* (*ovarian and breast cancer*).

In German, it is necessary to reconsider the rough distinction between single- multi-word terms in order to take morphological compounds into account. The common German compounds of the type $N_1|N_2$ ("|" is the concatenation operation) are often translated by N Sp N patterns in French: *Produktionsstandort* ⟷ *site de production* or by the N N patterns in English. Morphological compounds are identified by tokenisation programs as single-word terms but they look quite similar to multi-word terms. We use the morphological splitter which is combined with a dictionary look-up developed by Weller and Heid (2012) in order to get the MWT syntagmatic equivalence of a German morphological compound.

In order to build the test set, we have selected the French MWTs extracted by TermSuite for which the number of occurrences is greater than or equal to 5. The test set is composed of 976 French MWTs for which 90% of the base forms are only composed of two content words.

6 Experiments

In this section, we evaluate the performance of the dictionary look-up, the compositional method and CMCBP on the quality of bilingual terminology extraction.

6.1 Dictionary Look-up

First of all, we count the number of terms of the test set directly translated by looking them up in the bilingual dictionaries. From the 976 French MWTs to be translated, 51 are recorded in the French/English dictionary and 12 in the French/German dictionary. Here, the MWTs correctly translated are mainly generic terms that are not specific to the thematic of breast cancer such as *traitement médical/medical treatment* and *acide aminé/amino acid:* in French/English and *analyse statistique/statistische untersuchung* (*statistical analysis*) and *effet secondaire/begleiterscheinung* (*side effect*) in French/German. In this instance we were unable to generate any translations for 836 French MWTs in English and for 964 French MWTs in German.

6.2 Compositional Method

We then evaluate the quality of the translations provided by the compositional method (the MWTs found in the dictionary are not used). Table 1 shows the results obtained for the translation from French/English and German/English. The first column indicates the number of French MWTs that are translated. Since the compositional approach can give several target translations for one French MWT, the last two columns indicate the Top_1 and Top_5 accuracy. To evaluate the Top_n accuracy, we first keep for each French word to be translated its n first candidate translations and then measure the accuracy of the ranked lists obtained, *i.e.* the proportion of lists comprising the expected translation. Here, the candidate translations are

ranked according to their frequency in the target part of the comparable corpus. The results of this experiment show that 140 of the 836 French MWTs are translated into English for the Top_5 with a high level of accuracy: 79.1%, and 87 of the 964 French MWTs are translated into German for the Top_5 with a high level of accuracy: 95.7%. Here, we were unable to generate any translations for 785 French MWTs in English and 877 French MWTs in German.

	# trans.	Top_1	Top_5
French/English	140	73.2%	79.1%
French/German	87	88.8%	95.7%

Table 1: Results for the compositional method

6.3 Compositional Method with Context-Based Projection

We now apply CMCBP (here again the MWTs found in the dictionary are not used). In this experiment, the parameters required for our approach are as follows: the size of the context window w is up to 3 (i.e. a seven-word window), the association measure is Mutual Information, and the distance measure is Cosine. Other combinations of parameters were assessed but the previous parameters gave the best performance. Table 2 presents the percentage of French terms for which the correct translation is obtained among the $Top_{1, 5, 10,}$ and $_{20}$ candidates translations from French to English and German. Table 2 shows that 514 of the 836 French MWTs are translated into English with the CMCBP with an accuracy of 42.1% for the Top_1 and 57.1% for the Top_{20} and 510 of the 964 French MWTs are translated into German with an accuracy of 44.3% for the Top_1 and 51.2% for the Top_{20}. These results indicate that the majority of the correct translated MWTs are in fact obtained from the Top_5. Moreover, the CMCBP retains the advantages of the compositional method. All translations obtained with the compositional method are found in the same rank with the CMCBP.

	# trans.	Top_1	Top_5	Top_{10}	Top_{20}
French/English	514	42.1%	55.4%	56.8%	57.1%
French/German	510	44.3%	49.4%	51.2%	51.2%

Table 2: Results for the compositional method enhanced with context alignments

From the MWTs that are correctly translated and not found by the compositional approach, we found a large majority of French MWTs involving a relational adjective. However, the French MWT *dépistage mammographique* is not translated by the compositional approach since the French relational adjective *mammographique* is not found in the dictionaries. In contrast, the correct English translation *mammographic screening* is found in the Top_3 with the CMCBP because we have associated the French context vector of *mammographique* with the English context vector of *mammographic* and the French/English pair *dépistage/screening* is found in the dictionary. The other French MWTs correctly translated are mainly MWTs with a compositional structure for which one element is not found in the dictionary such as: *amélioration significative/significant benefit* (Top_1), and *caractéristique tumoral/tumor charakteristik* (tumor characteristic) (Top_1) or without a compositional structure such as: *bras témoin/control arm* (Top_1),

and *curage axillaire/axillary dissection* (Top_{11}). From the MWTs incorrectly translated, we can point out two main cases. First, we find target MWTs semantically close to the French MWTs to be translated such as: *postmenopausalen frau* (*postmenopausal women*) (Top_5) and *prämenopausalen frau* (*premenopausal women*) (Top_7) for *femme ménopausé* (*menopausal women*). *Postmenopausalen frau* and *prämenopausalen frau* are morphological variants of *menopausalen frau* and should have been identified as thus by the term extraction program, but unfortunately the canonical form *amenopausalen frau* does not occur in the comparable corpora. Secondly, we found only a sub-part of the English MWTs such as: *node dissection* for *curage ganglionnaire* (*lymph node dissection*). This case needs further work as it deals with a fertility case that is not able to be solved with the term variant recognition program.

7 Related Work

The principle of translation compositionality is restrictive. Several studies have concentrated on enhancing the compositional approach: Robitaille et al. (2006) proposed a backing-off method: if there is insufficient data in the dictionary to translate an MWT of *n* content words, a scaled MWT with a length less than, or equal to, *n* is used instead. Morin and Daille (2010) proposed an extended compositional method that bridges the gap between MWTs of different syntactic structures through morphological links. The compositional approach is also called the "bag-of-equivalents" approach (Vintar, 2010) when the bilingual dictionary is built from a parallel corpus and contains all words that occur in the corpus and their suggested translation equivalents, together with a probability score. The "bag-of-equivalents" approach has been used for SMT to build a word-level translation lexicon from parallel corpora (Munteanu and Marcu 2006) and a cognate lexicon from comparable corpora (Koehn and Knight 2002).

Much of the work involving general or LSP comparable corpora has focused on extracting SWT translations using only contextual information (Fung, 1998; Rapp, 1999; Chiao and Zweigenbaum, 2002; Gaussier et al., 2004; Morin et al., 2007; Laroche and Langlais, 2010, among others). The contextual information method as defined by Fung (1998) gives very low results for MWTs: from the 785 non translated French MWTs, 483 French MTWS with an accuracy of 15.6% (Top_{10}) were found.

CMCBP is a new method dedicated to MWT that combines both the compositional and the contextual information. CMCBP significantly improves both the compositional method commonly used for bilingual alignment of MWTs extracted from comparable corpora, and the contextual information method we obtained from English: 514 French translations of MWTS with a precision of 56.8% and 510 German translations of MWTS with a precision of 51.2% for Top_{10}.

8 Conclusion

In this study, we have investigated the compilation of bilingual terminologies from a specialized comparable corpus and show how to push back the limits of the compositional approach used in alignment programs to translate MWTs. We have proposed CMCBP: a compositional method enhanced with pre-processed context information. The experiments that we carried out have shown that we increase the results of the compositional approach by providing a significant number of additional correct lexical entries that could not be identified either by the dictionary look-up or by compositional methods.

In future work, we will generalise this method to obtain an homogeneous modular design for all languages by reconsidering the rough distinction between simple and complex terms and applying the CMCBP both at the morphological and the lexical levels. We will investigate how

to improve the solving of the fertility problem for MWTs which produces incomplete translations. Fertility was only partially solved thanks to the term variation treatment associated to the CMCBP. We aim to modify the evaluation protocol by accepting one-to-many translations in the case of synonym or semantically-related translation candidates that are not handled through term variation processing.

Acknowledgments

The research leading to these results has received funding from the European Community's Seventh Framework Programme (FP7/2007-2013) under Grant Agreement no 248005.

References

Baldwin, T. and Tanaka, T. (2004). Translation by Machine of Complex Nominals: Getting it Right. In *Proceedings of the ACL 2004 Workshop on Multiword Expressions: Integrating Processing*, pages 24–31, Barcelona, Spain.

Bowker, L. and Pearson, J. (2002). *Working with Specialized Language: A Practical Guide to Using Corpora*. Routledge, London/New York.

Brown, P., Della Pietra, S., Della Pietra, V., and Mercer, R. (1993). The Mathematics of Statistical Machine Translation: Parameter Estimation. *Computational Linguistics*, 19(2):263–311.

Chiao, Y.-C. and Zweigenbaum, P. (2002). Looking for Candidate Translational Equivalents in Specialized, Comparable Corpora. In *Proceedings of the 19th International Conference on Computational Linguistics (COLING'02)*, pages 1208–1212.

Fung, P. (1998). A Statistical View on Bilingual Lexicon Extraction: From Parallel Corpora to Non-parallel Corpora. In Farwell, D., Gerber, L., and Hovy, E., editors, *Proceedings of the 3rd Conference of the Association for Machine Translation in the Americas (AMTA'98)*, pages 1–16, Langhorne, PA, USA.

Gaussier, E., Renders, J.-M., Matveeva, I., Goutte, C., and Déjean, H. (2004). A Geometric View on Bilingual Lexicon Extraction from Comparable Corpora. In *Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics (ACL'04)*, pages 526–533, Barcelona, Spain.

Grefenstette, G. (1999). The World Wide Web as a Resource for Example-Based Machine Translation Tasks. In *ASLIB'99 Translating and the Computer 21*, London, UK.

Keenan, E. L. and Faltz, L. M. (1985). *Boolean Semantics for Natural Language*. D. Reidel, Dordrecht, Holland.

Koehn, P. and Knight, K. (2002). Learning a translation lexicon from monolingual corpora. In *Proceedings of the ACL-02 Workshop on Unsupervised Lexical Acquisition*, pages 9–16, Philadelphia, Pennsylvania, USA.

Laroche, A. and Langlais, P. (2010). Revisiting Context-based Projection Methods for Term-Translation Spotting in Comparable Corpora. In *Proceedings of the 23rd International Conference on Computational Linguistics (COLING'10)*, pages 617–625, Beijing, China.

McEnery, A.and Xiao, Z. (2007). Parallel and comparable corpora: What is happening? In Anderman, G. and Rogers, M., editors, *Incorporating Corpora: The Linguist and the Translator*, Multilingual Matters. Clevedon.

Melamed, I. D. (2001). *Empirical Methods for Exploiting Parallel Texts*. MIT Press, Cambridge, MA, USA.

Morin, E. and Daille, B. (2010). Compositionality and Lexical Alignment of Multi-word terms. In *Language Resources and Evaluation*, volume 44, pages 79–95. Springer.

Morin, E., Daille, B., Takeuchi, K., and Kageura, K. (2007). Bilingual Terminology Mining – Using Brain, not brawn comparable corpora. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics (ACL'07)*, pages 664–671, Prague, Czech Republic.

Munteanu, D. S. and Marcu, D. (2006). Extracting parallel sub-sentential fragments from non-parallel corpora. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics (COLING-ACL'06)*, Sydney, Australia.

Nakagawa, H. and Mori, T. (2003). Automatic term recognition based on statistics of compound nouns and their components. *Terminology*, 9(2):201–219.

Nomura, M. and M., I. (1989). *Gakujutu Yogo Goki-Hyo*. National Language Research Institute, Tokyo.

Pekar, V., Mitkov, R., Blagoev, D., and Mulloni, A. (2006). Finding translations for low-frequency words in comparable corpora. *Machine Translation*, 20(4):247–266.

Rapp, R. (1999). Automatic Identification of Word Translations from Unrelated English and German Corpora. In *Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics (ACL'99)*, pages 519–526, College Park, MD, USA.

Robitaille, X., Sasaki, X., Tonoike, M., Sato, S., and Utsuro, S. (2006). Compiling French-Japanese Terminologies from the Web. In *Proceedings of the 11th Conference of the European Chapter of the Association for Computational Linguistics (EACL'06)*, pages 225–232, Trento, Italy.

Rocheteau, J. and Daille, B. (2011). TTC TermSuite - A UIMA Application for Multilingual Terminology Extraction from Comparable Corpora. In *Proceedings of the IJCNLP 2011 System Demonstrations*, pages 9–12, Chiang Mai, Thailand. Asian Federation of Natural Language Processing.

Sag, I. A., Baldwin, T., Bond, F., Copestake, A. A., and Flickinger, D. (2002). Multiword expressions: A pain in the neck for nlp. In *Proceedings of the 3rd International Conference on Computational Linguistics and Intelligent Text Processing (CICLing'02)*, pages 1–15, Mexico City, Mexico.

Savary, A. and Jacquemin, C. (2003). Reducing Information Variation in Text. In Grefenstette, G., editor, *Text- and Speech-Triggered Information Access*, Lecture Notes in Computer Science, pages 141–181. Springer Verlag.

Schmid, H. (1995). Improvements In Part-of-Speech Tagging With an Application To German. In *In Proceedings of the ACL SIGDAT-Workshop*, pages 47–50.

Sharoff, S., Babych, B., and Hartley, A. (2009). 'irrefragable answers' using comparable corpora to retrieve translation equivalents. *Language Resources and Evaluation*, 43(1):15–25.

Tanaka, T. (2002). Measuring the Similarity between Compound Nouns in Different Languages Using Non-parallel Corpora. In *Proceedings of the 19th International Conference on Computational Linguistics (COLING'02)*, pages 1–7, Taipei, Taiwan.

Tanaka, T. and Baldwin, T. (2003). Noun-Noun Compound Machine Translation: A Feasibility Study on Shallow Processing. In *Proceedings of the ACL 2003 workshop on Multiword Expressions: Analysis, Acquisition and Treatment*, pages 17–24, Sapporo, Japan.

Vintar, S. (2010). Bilingual term recognition revisited: The bag-of-equivalents term alignment approach and its evaluation. *Terminology*, 16(2):141–158.

Weller, M. and Heid, U. (2012). Simple methods for dealing with term variation and term alignment. In *Proceedings of the 8th International Conference on Language Resources and Evaluation (LREC'12)*, Istanbul, Turkey. ELDA.

Is Bad Structure Better Than No Structure?: Unsupervised Parsing for Realisation Ranking

Yasaman MOTAZEDI Mark DRAS François LAREAU
Centre for Language Technology, Macquarie University
Yasaman.Motazedi@students.mq.edu.au, Mark.Dras@mq.edu.edu,
Francois.Lareau@mq.edu.au

Abstract
In natural language generation using symbolic grammars, state-of-the-art realisation rankers use statistical models incorporating both language model and structural features. The rankers depend on multiple structures produced by the particular large-scale symbolic grammars to rank the output; for languages with smaller resources and in-development grammars, we look at the feasibility of an alternative source of structural features, unsupervised parsers. We show that, in spite of their lower quality of structure, raw sets of unsupervised parse features can be helpful with smaller language models; and that the parses do contain particular elements that can be highly useful, improving performance on our classification task by up to 10% on 60% of the test set leading to an overall improvement under a back-off model.

Title and Abstract in French

Une mauvaise structure est-elle mieux que pas de structure du tout? L'analyse non supervisée pour la sélection des réalisations

Dans plusieurs systèmes récents de génération de texte basés sur des grammaires symboliques, les résultats sont ordonnés selon leur acceptabilité par des modèles statistiques qui incorporent des modèles de Markov et des traits structurels. Ces modules d'ordonnancement dépendent de diverses structures produites par la grammaire, ce qui présuppose une grammaire suffisamment développée. Pour les langues à faibles ressources ou pour les grammaires en cours de développement, nous étudions ici la viabilité d'une source alternative de traits structurels: les analyseurs non supervisés. Nous démontrons que, en dépit de la faible qualité des structures produites, elles contiennent des éléments qui peuvent être très utiles pour les langues peu dotées, permettant d'améliorer de 10% la performance de notre classificateur pour 60% des phrases de notre corpus de test.

Keywords: natural language generation, realization ranking, unsupervised parsing.

Keywords in French: génération de langue naturelle, analyseur statistique non supervisé.

Proceedings of COLING 2012: Technical Papers, pages 1811–1830,
COLING 2012, Mumbai, December 2012.

1 Condensed 2-page version in French

Les systèmes de génération de textes peuvent produire plusieurs textes de qualité variable pour les mêmes données. Cela a mené à la création d'algorithmes de sélection pour choisir le meilleur texte (Langkilde-Geary, 2000; Velldal and Oepen, 2005; Cahill et al., 2007, par exemple). Charniak (2001) a proposé d'utiliser les analyseurs statistiques comme source de structures sur lesquelles baser les algorithmes de sélection. À notre connaissance, seuls des analyseurs supervisés ont été utilisés à cet effet. Cependant, pour les systèmes basés sur des grammaires symboliques de modeste envergure ou en développement, trop peu de données sont disponibles, et les modèles existants s'appliquent mal. Dans cet article, nous étudions donc la possibilité d'utiliser un analyseur statistique non supervisé (Naseem et al., 2010) comme source alternative d'information pour la sélection automatique des textes générés, malgré la faible qualité des structures qu'ils produisent.

Nos travaux sont basés notamment sur ceux de Cahill et al. (2007), qui à la suite de Velldal and Oepen (2006), ont développé un modèle log-linéaire de réalisation pour une grammaire Lexical Functional Grammar (LFG). Comme eux, nous utilisons la plateforme Xerox Linguistics Environment (XLE) et construisons une banque d'arbres symétrique en analysant puis en regénérant les phrases. Ceci nous fournit à la fois des exemples positifs et négatifs. Nous utilisons les sections 2 à 21 du Penn Treebank (38 008 phrases) pour l'entraînement et la section 23 (2245 phrases) pour l'évaluation. Nous les avons analysées en utilisant la grammaire ParGram de l'anglais sur XLE (Butt et al., 2002), puis nous les avons regénérées en renversant la même grammaire pour produire de multiples paraphrases candidates. Nous avons rejeté les cas où la grammaire ne regénérait pas plus d'une phrase, ou plus de 1000, pour obtenir un corpus d'entraînement de 20 613 ensembles de paraphrases et un corpus de test de 1168 ensembles de paraphrases. Souvent, la grammaire n'arrivait pas à reproduire la phrase originale. Nous avons donc calculé la distance d'édition pour chaque paraphrase. Nous utilisons comme cas positif pour l'entraînement et pour le test la phrase regénérée ayant la plus petite distance par rapport à la phrase originale. Comme exemple négatif, nous avons comparé deux alternatives : la phrase ayant la plus grande distance (greatest, dans nos tableaux), et une phrase choisie au hasard parmi celles qui n'avaient pas la plus petite distance (random, dans nos tableaux).

Pour la classification, nous avons utilisé le système à entropie maximale MegaM (cinquième version) de Hal Daumé III. Pour fins de comparaison, nous avons reproduit la même méthode avec deux analyseurs supervisés : celui de Stanford (Klein and Manning, 2003) et celui de Charniak et Johnson (2005) (ci-après, C&J). Nous couplons les analyseurs à des modèles de Markov : un grand, construit avec SRILM (Stolcke, 2002) sur le corpus Gigaword, et un petit, afin d'imiter les conditions de développement pour des petites grammaires. Nous utilisons ces modèles de Markov à la fois comme modèles de base et en combinaison avec les modèles structurels. En outre, parce que le principal obstacle à l'utilisation d'un analyseur non supervisé est la faible qualité des analyses qu'il produit (voir Figure 1 pour une analyse non supervisée, par opposition à l'analyse supervisée de la Figure 2). Nous avons comparé deux approches pour identifier les dépendances les plus fiables à utiliser comme traits : le calcul de précision des dépendances individuelles, et la méthode de gain d'information.

Les résultats de notre expérience montrent que, sans surprise, les analyseurs non supervisés donnent des résultats nettement moins bons que les analyseurs supervisés. Cependant, ils sont utiles s'ils sont couplés à un modèle de Markov, même de taille modeste. L'utilisation de traits d'analyse seuls donne quand même des résultats meilleurs qu'un simple modèle de Markov. En conclusion, donc, les analyseurs non supervisés peuvent être utiles. En général, les traits qu'ils fournissent contribuent à améliorer les résultats obtenus avec un modèle de Markov de taille modeste, du type qu'on trouve normalement pour les langues peu dotées. Ils contribuent aussi très fortement à améliorer

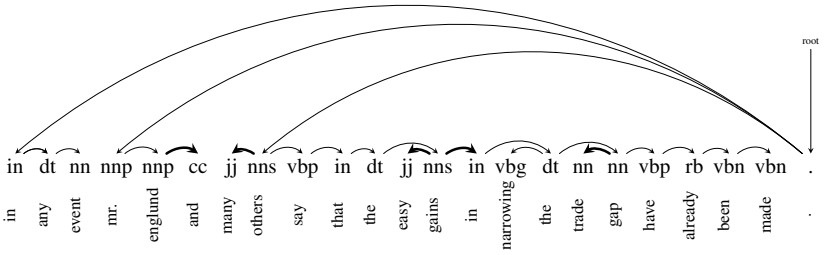


Figure 1 – Unsupervised dependency tree for a sample sentence.//Analyse non supervisée pour une phrase.

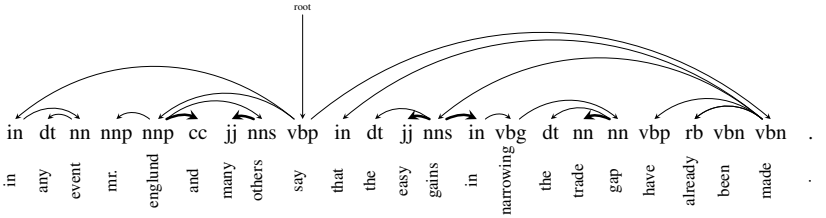


Figure 2 – Supervised dependency tree for sample sentence of Fig 1.//Analyse supervisée pour la même phrase qu'à la Figure 1.

les résultats pour un échantillon respectable du corpus de test, une fois que les traits utiles ont été identifiés par la méthode de gain d'information : nous avons observés des gains de précision de 10% dans 60% des phrases du corpus de test, ce qui rend possible l'utilisation d'un modèle de repli.

2 Introduction

Natural Language Generation (NLG) systems take some machine-oriented input such as databases, knowledge bases or logical forms and produce human-like text. The realization involves making many decisions on the surface representation of a sentence, including lexical selection, use of referring expressions, and word order. In general, such systems can thus generate multiple outputs, some of which are genuine good paraphrases, some of which may sound somewhat marked or odd out of context, and some of which may be (due to system limitations) incorrect.

This observation led to the idea of taking the possible outputs and ranking them, in the earliest work by use of a language model (LM) (Langkilde-Geary, 2000) and later by injecting new features like headwords, constituent category and Part of Speech (POS) tags (Langkilde-Geary, 2002). This idea was developed further for handcrafted symbolic grammars based on linguistic formalisms that have been used for generation. In the Head-driven Phrase Structure Grammar (HPSG) system LOGON, Velldal and Oepen (2005) improve on this by using a maximum entropy model incorporating both a LM and features derived from HPSG structures for the candidate outputs; this drew on earlier developments in the field of statistical parsing that used such features for reranking (Johnson et al., 1999; Riezler et al., 2002, for example). Cahill et al. (2007) and White and Rajkumar (2009) similarly show that ranking for Lexical Functional Grammar (LFG)- and Combinatory Categorial Grammar (CCG)-based systems respectively can be improved by models incorporating a LM and structural features.

All of these rankers depend on multiple structures produced by the respective large-scale symbolic grammars to rank the output. For much smaller symbolic grammars, and those in the process of development — e.g. LFG grammars for Indonesian (Arka et al., 2009) or Arrernte (Dras et al., 2012), or HPSG grammars for Persian (Müller, 2010) or Wambaya (Bender, 2008) — these multiple structures will not be available or will be quite impoverished. The goal of this paper then is to investigate using other sources of structural information that will be available for such languages: specifically — since such languages are also often unlikely to have treebanks — unsupervised statistical parsers. Unsupervised parsers have been motivated by possible application in contexts where large treebanks for training supervised parsers are absent (Klein and Manning, 2004, for example), but given their very modest performance with respect to supervised alternatives it is an open question as to whether the structures they produce have any extrinsic application. We aim to give an answer to this question in the context of realisation ranking.

In Section 3 we look at some previous work on reranking in realization in more depth, and more briefly note work on parser-based models for ranking and on unsupervised parsing. In Section 4 we describe our models based on statistical parsers, and the experimental set-up for investigating them. In Section 5 we discuss our results, and conclude in Section 6.

3 Related Works

Reranking in Realization Using statistical models that can combine a wide range of features to rank candidate outputs is standard across many applications, such as parse selection and Machine Translation (MT). They have also been applied to realisation ranking in NLG, to select the best of a set of candidate generated sentences. Here we review some of these, with a focus on the work of Cahill et al. (2007), as this is the setup closest to the one we use in this paper.

Velldal and Oepen (2006) implemented realisation ranking in the context of an MT system that uses a hand-crafted HPSG grammar to generate sentences from semantic specifications. Their system produced multiple candidate sentences which were ranked under three different models: an n-gram

LM; a discriminative maximum entropy model using (HPSG) structural features; and a combination of the two, which produced the best results.

To provide training data for the model, they constructed a 'symmetric treebank' (Velldal et al., 2004) composed of:

1. a set of pairings of surface forms and associated semantics;
2. a set of alternative analyses for each surface form; and
3. a set of alternative realisations of each semantic form.

The preferred realisations are automatically specified by comparing the yields of the generated sentences with the original strings in the treebank (for them, the Redwoods English treebank); this gives the sets of positive and negative (i.e. all non-original) examples for the maxent learner.

Cahill et al. (2007) drew on this to develop a log-linear model for realisation in an LFG context. They worked with a large-scale hand-crafted grammar of German, as they were investigating applicability of the approach to a language with freer word order than English, and generated from f-structures using the XLE system (Maxwell and Kaplan, 1993). They similarly constructed a symmetric treebank, and also had as their goal to re-generate the strings in the original treebank.

For structural features in their model, the authors defined 186,731 instantiated templates based on Riezler et al. (2002), Rohrer and Forst (2006) and Riezler and Vasserman (2004); 1,471 of these actually occurred in their training data. The feature templates included information from both c-structure (e.g. simple features such as number of times a particular category label occurs, or compound features such as number of times it dominates another) and f-structure (e.g. frequency of relative order of subject and object), sentence length and LM scores. Structural features here contributed more strongly than for English.

Which structural features are effective in realisation ranking is still an open question. For example, in the context of CCG, Rajkumar and White (2010) found that features reflecting animacy agreement between nouns and relative clauses, and number agreement between subject and verb, were helpful. White and Rajkumar (2012) found that explicitly representing dependency lengths led to shorter average dependencies, in line with psycholinguistic evidence for human-produced sentences, and to better generated text. And Filippova and Strube (2009) found that while trigram LMs are appropriate for phrase linearisation in German, at a clause level (longer) dependency features produced better results.

Parser-based Ranking Charniak (2001) proposed the idea of using statistical parsers as a kind of structural language model. The idea has been applied a number of times in MT, for example by Charniak et al. (2003) or Post and Gildea (2008); it has also been applied in NLG, where Mutton et al. (2007) showed that a combination of parser-based metrics correlated with human judgements of the quality of generated text. To our knowledge, all work applying parser-based ranking has used supervised parsers. There is also an interesting piece of work by Cherry and Quirk (2008) where implicit discriminative syntactic LMs are constructed, using a latent Support Vector Machine (SVM) to train an unlexicalised parser to judge sentences produced by an MT system. The authors discuss some similarities to unsupervised parsing, in that both sorts of parsers are trained on sentences without the benefit of annotated parse trees. Using unsupervised parse trees as we do in this paper, however, can benefit from lexicalisation and, potentially, linguistic knowledge embodied in the parser (see below).

Unsupervised Parsers The first unsupervised parser that convincingly beat fairly simple baselines, such as constructing a right-branching tree, was the DMV model of Klein and Manning (2004), which combined constituency and dependency parsing to produce a better unsupervised parser than either separately. There have been various developments in unsupervised parsing since then: Headden III et al. (2009) introduced basic valence frames and lexical information, along with a smoothing technique to handle resulting data sparsity; Berg-Kirkpatrick and Klein (2010) used a phylogeny-structured model of parameter drift; and Naseem et al. (2010) used a single set of manually specified language-independent rules characterising syntactic dependencies across languages, as a soft constraint during the inference of the probabilistic model. We use this last one in our work, as it gives state-of-the-art results, and embodies potentially useful hard-coded universal tendencies. Notwithstanding that state-of-the-art performance, directed dependency results for that system range from only 50.9% (Slovene) to 71.9% (English), and importantly, as in previous work, these cross-linguistic evaluations are only carried out on sentences of 10 or fewer words.

4 Experimental Setup

Our goal is to evaluate the usefulness of the structural information proposed by an unsupervised parser in assessing the quality of sentences generated by a symbolic grammar, given its noticeably lower quality than supervised parsers. Like Cahill et al. (2007), we use the XLE system and construct a symmetric treebank by parsing and re-generating sentences: this gives both positive examples (those that match the original sentence) and negative examples (those that don't). Details of the various aspects of this process follow.

4.1 Data and Evaluation

We took the Penn Treebank sections 2–21 for training, consisting of 38008 sentences, and section 23 for testing, consisting of 2245 sentences. We parsed them using XLE and the large-scale ParGram grammar of English (Butt et al., 2002), and then used XLE's re-generate facility to produce the multiple candidate realisations. As Cahill et al. (2007) did, we found that XLE could not parse and re-generate all sentences; and for the purposes of constructing a training set, only instances that re-generated more than one sentence were useful.[1] In addition, we excluded the few sentences that contained more than 1000 re-generated sentences. This resulted in a training set of 20613 (sets of) sentences and a test set of 1168 (sets of) sentences.

Also as Cahill et al. (2007) did, we found that not all sets of re-generated sentences contained the original sentence in its exact form: this was often because of small differences such as use of abbreviations (e.g. *Nov* for *November*) or use of punctuation. In contrast to their approach, where they discarded these cases, we used minimum edit distance (MED) to determine the re-generated candidate closest to the original.

Following this, we constructed training and test sets of pairs of re-generated sentences by taking as the positive example the one with smallest MED from the original sentence, and as the negative example one of two alternatives: either the one with largest MED (greatest) or a random selection from those candidates that did not have the smallest MED (random). greatest always gave the higher result in the classification experiments below, often by several percentage points (which is not surprising, as the differences are larger and the classification hence easier), so we mostly only report results for random.

[1] There is also a technical issue in re-generating numerals. To get around this problem, we preprocessed the text to change numerals to those that could be re-generated.

We used the maximum entropy learner MegaM (fifth release) by Hal Daumé III.[2] Our evaluation metric is the classification accuracy of predicting the positive example from each pair.

4.2 Parsers and Language Models

Before answering any questions about the usefulness of unsupervised parsers, we address the question, Do *supervised* parsers produce structural information that is useful for realisation ranking? If the answer is yes (and based on the work cited in Section 3 and others, we would think it likely), the supervised parsers and a large LM would constitute an upper bound on the efficacy of using parsers to rank candidate sentences generated by XLE. The supervised parsers we use are the Stanford parser (Klein and Manning, 2003) and the Charniak and Johnson (henceforth C&J) parser (Charniak and Johnson, 2005). These parsers are both quite accurate: the Stanford parser gets a labelled f-score of 85.61 on the WSJ, and the C&J 91.09.

From the Stanford parser we examined both horizontal slices of parse trees, in effect treating them as sets of CFG production rules, and dependencies; the production rules and dependencies were either lexicalised or unlexicalised, and the dependency relations either named or unnamed. This gives two constituency and four dependency feature representations from the Stanford parser: prod-rule-lex and prod-rule-unlex; and dep-lex-named, dep-lex-unnamed, dep-unlex-named and dep-unlex-unnamed.

C&J is a reranking parser; the reranker uses 13 feature schemas such as tuples covering head-to-head dependencies, preterminals together with their closest maximal projection ancestors, and subtrees rooted in the least common ancestor. We took two types of features from C&J: production rules; and the instantiated feature schemas from the parse reranking process, making them do 'double duty' as realisation ranking features. C&J is used as a complement to the Stanford parser here, with respect to production rules, as an easy way of getting something approximating the compound features used in realisation ranking discussed in Section 3.

The large LM was constructed using SRILM (Stolcke, 2002) on the Gigaword corpus.[3] We use the 64K 4-gram and the 64K 2-gram (where the size nK represents the use of the top n words occurring in the training text as vocabularies), which are at the two extremes of size. We refer to these as l-lm4 and l-lm2.

For the unsupervised structure that is the main interest of the paper, we used the parser of Naseem et al. (2010),[4] which constructs dependency trees. Relation names are not inferred, so the two alternative representations are lexicalised (unsuper-lex) and unlexicalised (unsuper-unlex).

For a small LM that would be of a realistic size for the scenario we are interested in — i.e. the development of a symbolic grammar for a language with few resources — we considered the size of corpora produced by *An Crúbadán*[5] (Scannell, 2007). This is a web crawler whose specific goal is the "automatic development of large text corpora for minority languages". As examples, it has produced corpora of ~2M words for Akan (Ghana), ~5M words for Tamil, and ~7M words for Turkmen. Consequently, we use the Penn Treebank (~4M words) as our realistically-sized LM. On this sized corpus, we derive both 4-gram and 2-gram LMs from SRILM using the default settings; we refer to these as s-lm4 and s-lm2.[6]

[2] MegaM software is available on `http://www.cs.utah.edu/~hal/megam/`.

[3] `http://www.keithv.com/software/giga/` was the source for these models. They used interpolated, modified Kneser-Ney smoothing, bigram cutoff 3, trigram cutoff 5.

[4] `http://groups.csail.mit.edu/rbg/code/dependency/`

[5] `http://borel.slu.edu/crubadan/`

[6] We note that these are in-domain LMs, in contrast to the Gigaword-derived ones, and so will have a bit of an advantage.

4.3 Models

Base models, supervised and unsupervised The basic models are then the structural models described above. We used the LMs both as baselines and in combination with the structural models. (Note that the C&J parser already includes a LM in its reranking feature templates that we use, so we do not combine in this case.) In the base cases, we use all structures (production rules, templates or dependencies as appropriate) returned by the relevant parser.[7]

For the unsupervised parser, the major issue is that the parses are generally of lower quality. As an illustration, Figures 1 and 2 represent unsupervised and supervised parses of the same sentence (common edges are indicated in bold). The supervised parse looks relatively reasonable, while the unsupervised parse makes some odd choices (*mr.* being a dependent of the root, the verb having no special status, etc), and contains many adjacent dependencies. Figures 3 and 4 represent another pair of parses, this one with a few longer dependencies than the previous pair of figures.

We take two approaches to finding higher quality individual dependencies for use as features: one is the calculation of fine-grained accuracy rates for dependencies, and the other is Information Gain (IG). In the case of fine-grained accuracy rates, we are assessing which dependencies are likely to be reliable, by comparison with a gold standard,[8] and in the case of IG, we identify which dependencies are particularly strongly associated with positive or negative examples.

Unsupervised with reliability-based selection To measure unsupervised dependency reliability, raw precision / recall scores would capture some of that notion — e.g. `VBD TO`, with 6593 gold-standard instances and 1520 correctly identified (recall = 0.231), is almost certainly more reliable than `NNP IN` with 7031 gold standard instances and only 547 correctly identified (recall = 0.078) — but they would obviously overrepresent low frequency dependency pairs, which are likely to be particularly unreliable. Taking the precision and recall scores as probabilities of correctness, we therefore adopted a prior α to smooth the scores. Given the relevant denominators N_p for precision or N_r for recall for each dependency pair type, our modified scores use $N_p + \alpha$ and $N_r + \alpha$ respectively; this leaves a probability mass $\frac{\alpha}{\alpha+N_p}$ (resp. $\frac{\alpha}{\alpha+N_r}$) for unseen instances of each dependency. By inspection of the actual raw scores on the training set, we chose $\alpha = 20$. We then select features with modified scores above various thresholds.

For lexicalised dependencies, with their much greater data sparsity issues, we based the choice on the corresponding unlexicalised dependency: if the parts of speech of the lexicalised dependencies matched an unlexicalised dependency above a particular threshold, we selected that lexicalised dependency. (So `the man` would deemed reliable if `DT NN` were deemed reliable.)

Unsupervised with IG selection We calculate IG over the training set, using a standard formulation of Yang and Pedersen (1997):

$$\begin{aligned} IG(r) = & -\sum_{i=1}^{m} \Pr(c_i) \log \Pr(c_i) \\ & + \Pr(r) \sum_{i=1}^{m} \Pr(c_i|r) \log \Pr(c_i|r) \\ & + \Pr(\bar{r}) \sum_{i=1}^{m} \Pr(c_i|\bar{r}) \log \Pr(c_i|\bar{r}) \end{aligned}$$

[7] In cases where the feature representation is the same for both positive and negative instances, we make a random choice.

[8] The Penn Treebank dependency gold standard was derived using `http://nlp.cs.lth.se/software/treebank_converter/`.

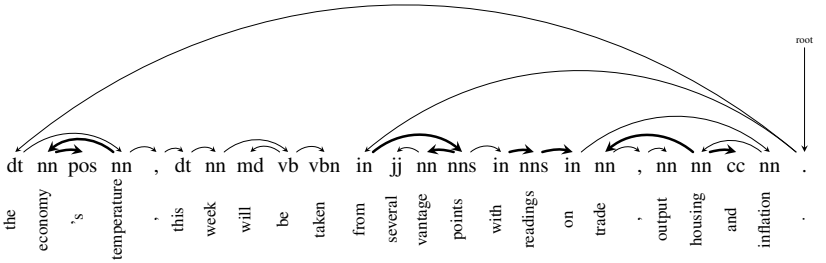


Figure 3: Unsupervised dependency tree for another sample sentence.

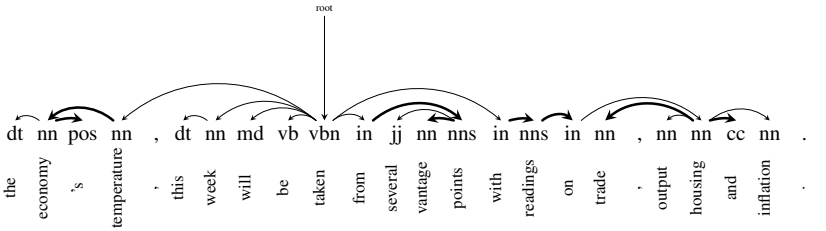


Figure 4: Supervised dependency tree for sample sentence of Fig 3.

with r representing a dependency, c a binary class, and $m = 2$. We then select features with IG scores above various thresholds. The application to unlexicalised features is straightforward: IG is applied to the unlexicalised dependencies, and some subset of these chosen based on the resulting ranking and a particular threshold. For lexicalised dependencies, there are two alternatives. One is to apply IG to the lexicalised dependencies directly (direct); the second is the same as the approach for unsupervised dependencies with reliability selection above, where we extract all the lexicalised dependencies that have corresponding selected unlexicalised dependencies (indirect).

Of course, both of these refinements of unsupervised parse features require some kind of annotation, and we discuss the implications for our scenario later in Section 5; but here we are just interested in the question of the extent to which unsupervied parsers can produce dependencies that are at all useful for realisation ranking.

5 Results

Base models (supervised) Table 1 gives classification accuracy results for the large LMs, and for the supervised parsers, both separately and in combination with the LM, all on the random test set. The key results here are:

1. Using parse features alone almost always outperforms the LM (except in the case of dep-unlex-

model	acc.%	+l-lm4	+l-lm2
l-lm4	68.36	-	-
l-lm2	64.16	-	-
C&J	90.50	-	-
prod-rule-lex	79.48	80.76	80.33
prod-rule-unlex	71.49	72.24	71.64
dep-lex-named	72.45	76.18	74.68
dep-lex-unnamed	71.51	75.54	74.72
dep-unlex-named	69.88	72.92	71.55
dep-unlex-unnamed	63.41	66.75	65.04

Table 1: Classification scores for supervised parse features and large LM on random: accuracy on parse features; parse features plus large l-lm4 LM over these sentences; parse features plus large l-lm2 LM over these sentences

unnamed), and the combination with the LM always improves over just parse features alone. This is broadly in line with the findings discussed in Sec 3 on using structural features from symbolic grammars, and suggests that using external statistical parsers is a valid alternative approach to carrying out realisation ranking.

2. While the l-lm4 LM forms quite a strong baseline, the l-lm2 version is somewhat weaker. It does still, however, contribute to an improvement in performance when added to the dependency feature models, of around 2%. This is not unexpected, as it would be contributing information that is complementary to the dependencies, which (in the cases where the dependencies are not adjacent) give longer-distance information. For the same reason, adding this bigram data to the production rule models produces a much smaller improvement.

3. Production rule features are better than dependency features; presumably one reason for this is that there are more production rule features (i.e. the ones consisting of only non-terminal nodes in the tree).

4. The C&J parser's templates — one particular set of choices for representing compound structural features — outperform just production rules combined with a LM, quite substantially. This also fits with previous work on using compound features.

5. In comparison with the results for the greatest test set (Table 2), as mentioned above, random is always consistently lower, for each comparable cell in the table. This is expected on the assumption that our MED method for choosing positive and negative examples is an accurate reflection of their goodness with respect to the original sentence. There is generally the same pattern for subsequent results as well, so henceforth we only report random, as the more conservative of the two measures and as a more realistic scenario (comparing a reference sentence against some arbitrary one, not one that we know is the 'worst' of a set of candidates).

Base models (unsupervised) Table 3 gives results when unsupervised parse structures are combined with a large LM. Not surprisingly, these results are quite a lot worse than for the supervised: a drop of around 13% for lexicalised dependencies and 8% for unlexicalised. These are also lower than the LMs in Table 1. For a more detailed look, we calculated the classification accuracy over those sentences where the feature representation differed (which we refer to as the *effective test*

model	acc.%	+l-lm4	+l-lm2
l-lm4	71.83	-	-
l-lm2	65.19	-	-
C&J	91.61	-	-
prod-rule-lex	84.25	84.45	84.19
prod-rule-unlex	75.81	76.56	76.01
dep-lex-named	76.78	81.11	78.58
dep-lex-unnamed	75.75	80.42	77.98
dep-unlex-named	73.25	75.75	74.38
dep-unlex-unnamed	66.80	69.84	66.97

Table 2: Classification scores for supervised parse features and large LM on greatest: accuracy on parse features; parse features plus large l-lm4 LM over these sentences; parse features plus large l-lm2 LM over these sentences

model	overall acc.%	#sent	acc.%
unsuper-lex	62.67	941	65.73
unsuper-unlex	58.26	791	62.20

Table 3: Classification scores for unsupervised parse features and large LM on random: accuracy on parse features; number of sentences where feature vectors differ for positive and negative examples; accuracy over effective test set.

model	#sent	l-lm4	+l-lm4	l-lm2	+l-lm2
unsuper-lex	941	75.29	70.03	69.65	68.97
unsuper-unlex	791	75.53	67.95	71.49	67.95

Table 4: Classification scores for unsupervised parse features and large LM on random: number of sentences where feature vectors differ for positive and negative examples; accuracy of language models (individually, or combined with unsupervised model) over effective test set.

model	#sent	s-lm4	+s-lm4	s-lm2	+s-lm2
unsuper-lex	941	67.80	68.76	67.48	68.76
unsuper-unlex	791	67.95	64.48	67.48	63.84

Table 5: Classification scores for unsupervised parse features and small LM on random: number of sentences where feature vectors differ for positive and negative examples; accuracy of language models (individually, or combined with unsupervised model) over effective test set.

set) and hence the model makes a genuine prediction,[9] to see whether a back-off model would be appropriate: if the accuracy for the unsupervised models is higher than for the LM over the effective test set, the decision for that subset could be based on the unsupervised model decision, with a back-off to the LM. Table 3 includes the effective test set, and the unsupervised classification accuracy scores over that set, for each model.

The LMs may also differ over the effective test sets: we present the classification accuracies for these in Table 4 (for the large LMs) and Table 5 (for the small LMs). The tables include both the accuracy of the LMs alone (e.g. the column l-lm4), as well as the accuracy of the LM in combination with the unsupervised model (e.g. +l-lm4). It is apparent that the effective test sets for the unsupervised models are also in fact easier for the large LMs: their scores on these subsets are all higher than the overall LM accuracies. They are also higher than the unsupervised models, and the combinations are lower than for the LMs alone. The story is different for the small LMs: while the LMs are higher than the unsupervised models, the combinations are better than both, by 3% in the lexicalised case. The conclusion here would be that if a very large LM is available, raw unsupervised parse features would not be helpful; but if only a small LM is available, they would still contribute.

Unsupervised with reliability-based selection Table 6 presents the results for our recall-based reliability measure, along with the effective test set sizes for four thresholds across dependencies with a positive reliability score.[10] These are uniformly poor, and in fact marginally worse than the raw unsupervised features in Table 3, so we do not present combinations with the LMs. We looked at the 10 highest- and 10 lowest-ranked features under this measure (Table 7), along with their raw counts in the training corpus. The highest-ranked ones were believable as reliable instances of dependencies: infinitival *to* in VB TO seems likely to be often correct, as does existential *there* in the first three cases. However, it is quite possible that many of the reliable ones such as PRP VBZ are actually poor at distinguishing between positive and negative examples by virtue of their frequency — they may occur equally often with both, in the same way that words like *the* are useless in general text classification.

Another possibility is that our reliability metric is not capturing the right phenomenon: that it is flagging as errors systematic intentional choices that the unsupervised parser is making, just because they happen to disagree with the systematic choices of the gold standard. For example, in Figures 3 and 4, the unsupervised parse contains the dependency MD VB, choosing serial attachment of auxiliaries and modals to the main verb, while the supervised parser contains MD VBN, choosing attachment of all to the main verb instead. Here the unsupervised parser does not necessarily seem incorrect. Indeed, the very smallness of the proportion of correct instances in the lowest-ranking dependency pairs in Table 7 suggests that these are not just random (and almost always incorrect) choices by the unsupervised parser; four of them in fact appear to relate to the sort of verb attachment choices mentioned. The issue of the difficulty of comparing dependency treebanks in general, perhaps because of fairly arbitrary decisions about headedness, is raised in Zeman et al. (2012); this would appear to be relevant to the task here too.

[9] Recall that in cases where the model cannot make a prediction, it chooses at random.

[10] This consequently does not include dependencies with recall-based reliability zero. We do not present the precision-based ones here.

model	cut-off%	acc.%	#sent
unsuper-lex	100	65.19	915
unsuper-lex	75	64.33	897
unsuper-lex	50	64.37	856
unsuper-lex	25	64.55	708
unsuper-unlex	100	61.44	721
unsuper-unlex	75	62.39	686
unsuper-unlex	50	62.00	629
unsuper-unlex	25	62.44	442

Table 6: Classification scores for unsupervised parse features selected by reliability on random: threshold cut-off accuracy on parse features; number of sentences where feature vectors differ for positive and negative examples (effective test set)

Highest			Lowest		
feature	correct	# in gold	feature	correct	# in gold
EX VBZ	72	179	VBZ VBP	1	201
EX VBP	45	110	VBP VBP	1	218
EX VBD	27	72	VBD NNP	1	292
DT VBZ	108	364	VBP VBD	1	335
$ TO	151	541	VBZ VBD	2	696
PRP VBZ	509	1792	IN CD	2	860
RP VBN	60	222	NN $	1	481
PRP VBD	600	2423	CD RB	1	485
VB TO	1520	6593	CD TO	1	487
NNS JJ	28	103	IN MD	1	694

Table 7: Highest- (left) and lowest- (right) ranking dependency features based on reliability measure. For each, columns represent the dependency; the number of times it was correct in an unsupervised parse; and the number of times it occurred in the gold standard.

model	cut-off%	acc.%	#sent	l-lm4
unsuper-lex(direct)	100	71.97	710	80.28
unsuper-lex(direct)	75	82.17	300	81.17
unsuper-lex(direct)	50	83.26	242	80.58
unsuper-lex(direct)	25	90.00	185	82.70
unsuper-lex(indirect)	100	74.57	920	78.80
unsuper-lex(indirect)	75	82.56	894	78.91
unsuper-lex(indirect)	50	85.19	849	79.62
unsuper-lex(indirect)	25	89.53	717	79.50
unsuper-unlex	100	60.84	738	79.20
unsuper-unlex	75	60.59	708	79.45
unsuper-unlex	50	61.08	657	80.29
unsuper-unlex	25	62.06	506	80.90

Table 8: Classification scores for unsupervised parse features selected by IG: threshold cut-off; accuracy on parse features; number of sentences where feature vectors differ for positive and negative examples (effective test set); large LM score over these sentences

rank	features				
1–5	TO VB	VBN VB	VB VBN	VB TO	VBG VB
6–10	VB VBG	NNS TO	IN RBR	VBN TO	DT RP

Table 9: Highest ranking features based on IG.

Unsupervised with IG selection Table 8 shows the results for selecting the top $k\%$ unsupervised parse features ranked by positive IG scores.[11] We see a dramatic improvement in the lexicalised case (direct) over the raw features (Table 3). The effective test set also falls a lot more for the direct lexicalised case; this is not surprising, as the top (say) 25% lexicalised features on the training set are much less likely to occur in the test set than the top 25% unlexicalised features. On the other hand, the indirect lexicalised case — where lexicalised features are instantiated on the basis of IG-ranked *unlexicalised dependencies* — have a sentence coverage that is much greater, as would be expected, but perhaps surprisingly an accuracy that is comparable to the direct method, and in fact even higher for three of the four thresholds presented in Table 8. For the 25% threshold on the indirect method, the model performs around 10% higher on around 60% of the total test set, making a back-off model a very suitable option.

Table 9 presents the top 10 unlexicalised features. It is not immediately apparent why there is a preponderance of paired verbs, such as VB VBG or VB TO (with the infinitival *to* probably indicating another verb following). Perhaps verb sequences indicate poor sentences, although this would require further inspection of the data.

Illustration of Generated Content To see how unsupervised parses, though bad, might contribute useful structure, we present an example where the unsupervised parse model predicted the better candidate, while the other models did not. The original sentence is given in (1), the preferred candidate in (2), and the dispreferred candidate in (3). The preferred candidate is basically the same as the original, slightly odd punctuation notwithstanding, while the dispreferred candidate has obvious problems in terms of ordering.

(1) For example, their selling caused trading halts to be declared in USAir Group, which closed down 3 7/8 to 41 1/2, Delta Air Lines, which fell 7 3/4 to 69 1/4, and Philips Industries, which sank 3 to 21 1/2.

(2) For example their selling, caused trading halts to be declared in USAir Group which closed, down 3 7/8 to 41 1/2 Delta Air Lines which fell 7 3/4 to 69 1/4 and Philips Industries which sank 3 to 21 1/2.

(3) For example to be declared in USAir Group, their selling, caused trading halts which closed, down 3 7/8 to 41 1/2 Delta Air Lines which fell 7 3/4 to 69 1/4 and Philips Industries which sank 3 to 21 1/2.

Figure 5 shows the unsupervised (upper) and supervised (lower) parses for the preferred candidate (2). For the most part, the supervised parser makes sensible linguistic choices: it identifies as the head of e.g. *Delta Air Lines which fell 7 3/4 to 69 1/4* the final noun of the named entity, which in turn is a dependant of the head of the previous parallel clause; the unsupervised parser, by contrast,

[11] That is, those cases with an IG of zero — i.e. entirely non-distinguishing between positive and negative examples — are excluded, which is why the 100% scores are higher than for the raw features of Table 3.

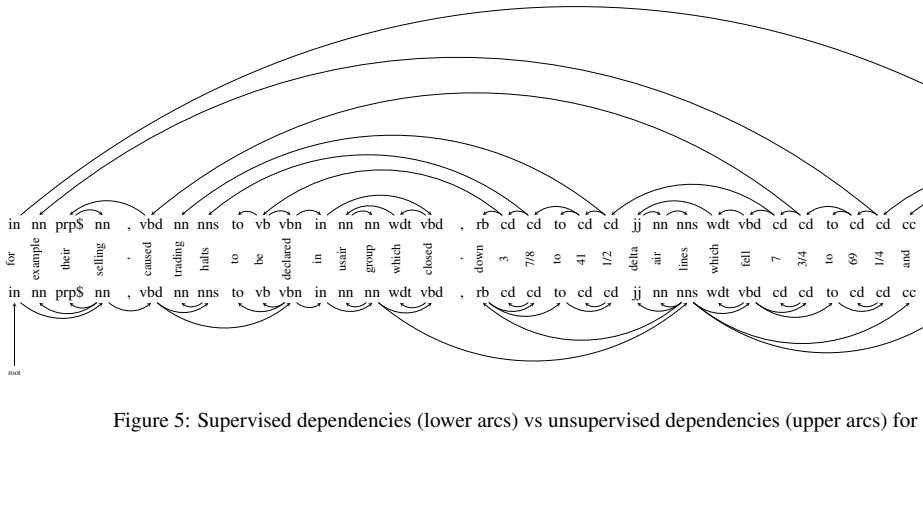


Figure 5: Supervised dependencies (lower arcs) vs unsupervised dependencies (upper arcs) for

has as the ultimate head the final numeral, with odd choices in the intermediate edges as well such as *which* being the head of the named entity in each case. However, the supervised parser makes a fundamental error in grouping two chunks of numerical quantities (i.e. *down 3 7/8 to 41 1/2* and *fell 7 3/4 to 69 1/4*) being associated with the same named entity. The unsupervised parse does not do this: while it makes odd choices, they are consistent odd choices. This consistency of choices may in fact not be accidental, and is further reason to reconsider our approach to assessing the reliability of unsupervised parse edges.

Implications If the reliability-based selection for unsupervised parse features had proved effective, it would have been necessary to find some mechanism to approximate a check against a gold standard, such as constructing a committee of unsupervised parsers and using only those dependencies that received a sufficient number of votes. However, this is not warranted by the results.

For the IG-based selection, what is necessary is a binary annotation of sentence pairs as preferred or dispreferred. This is much less intensive than treebank annotation, and so a reasonable alternative to constructing supervised parsers. As noted, a large LM outperforms raw unsupervised features; but even constructing large corpora (semi-)automatically for many of the world's languages, as per Scannell (2007), is challenging, and the IG-selected features in any case do better than the large LM alone and better still in conjunction with it.

6 Conclusion

For symbolic grammars that are small and/or in development, there may well not be many resources to draw on for building good realisation ranking models for natural language generation. In this paper we have examined unsupervised parsers as a possible source of structural features for such models, notwithstanding their generally much poorer quality of parses than supervised parsers.

We found that they can indeed be useful. In the general case, unsupervised parse features contribute in the case of smaller language models, of the sort that might be available for many less resourced languages. They also contribute very strongly over reasonable sized subsets of the test set once useful features have been identified by Information Gain: there are improvements of up to 10% in classification accuracy over 60% of the test set, making a model that uses unsupervised features and then backs off to a language model an attractive option. This would be feasible in scenarios where it is possible to annotate pairs of sentences as preferred and dispreferred.

Choosing features by a reliability-based measure did not prove useful. However, this may be related to systematic choices made by the unsupervised parser that were different from the gold standard's choices, rather than bad parsing; an option for aligning systematic choices is the HamleDT approach and software of Zeman et al. (2012). Another option for improving performance is the use of compound features, as is used in much of the work discussed in Section 3. The results in this paper for the supervised parsers showed that the model with compound (or 'higher order') features dramatically outperformed the ones with simple features; it could be promising to extend these to the dependency models, as for example in their use in parse reranking, such as by Hall (2007) and Wang and Zong (2011). In addition to evaluating the approach by measures other than binary classification accuracy (e.g. bleu or the ranking score of Cahill et al. (2007)), and examining other potential comparators (e.g. the treebank-trained generator of Belz (2005)), the next major step would be applying it to one of these smaller languages that has motivated the work in this paper.

Acknowledgements

The authors acknowledge the support of ARC Discovery grant DP1095443.

References

Arka, I. W., Andrews, A., Dalrymple, M., Mistica, M., and Simpson, J. (2009). A linguistic and computational morphosyntactic analysis for the applicative -i in Indonesian. In *Proceedings of LFG 2009*, pages 85–105, Cambridge, UK.

Belz, A. (2005). Statistical Generation: Three Methods Compared and Evaluated. In *Proceedings of the 10th European Workshop on Natural Language Generation (ENLG 2005)*, pages 15–23, Edinburgh, UK.

Bender, E. (2008). Radical Non-Configurationality without Shuffle Operators: An Analysis of Wambaya. In *Proceedings of the Fifteenth Annual Conference on Head-Driven Phrase Structure Grammar (HPSG08)*.

Berg-Kirkpatrick, T. and Klein, D. (2010). Phylogenetic Grammar Induction. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 1288–1297, Uppsala, Sweden.

Butt, M., Dyvik, H., King, T. H., Masuichi, H., and Rohrer, C. (2002). The Parallel Grammar Project. In *Proceedings of COLING-2002 Workshop on Grammar Engineering and Evaluation*, pages 1–7.

Cahill, A., Forst, M., and Rohrer, C. (2007). Stochastic realisation ranking for a free word order language. In *Proceedings of the Eleventh European Workshop on Natural Language Generation*, ENLG '07, pages 17—-24, Morristown, NJ, USA. Association for Computational Linguistics.

Charniak, E. (2001). Immediate-Head Parsing for Language Models. In *Proceedings of 39th Annual Meeting of the Association for Computational Linguistics (ACL'01)*, pages 124–131, Toulouse, France.

Charniak, E. and Johnson, M. (2005). Coarse-to-fine n-best parsing and MaxEnt discriminative reranking. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 173–180, Ann Arbor, Michigan.

Charniak, E., Knight, K., and Yamada, K. (2003). Syntax-based Language Models for Machine Translation. In *Proceedings of MT Summit IX*.

Cherry, C. and Quirk, C. (2008). Discriminative, Syntactic Language Modeling through Latent SVMs. In *Proceedings of the Eighth Conference of the Association for Machine Translation in the Americas (AMTA'08)*.

Dras, M., Lareau, F., Börschinger, B., Dale, R., Motazedi, Y., Rambow, O., Turpin, M., and Ulinski, M. (2012). Complex Predicates in Arrernte. In *Proceedings of LFG 2012*, Bali, Indonesia.

Filippova, K. and Strube, M. (2009). Tree Linearization in English: Improving Language Model Based Approaches. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL'09)*, pages 225–228, Boulder, Colorado.

Hall, K. (2007). K-best spanning tree parsing. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics (ACL'07)*, pages 392–399, Prague, Czech Republic. Association for Computational Linguistics.

Headden III, W. P., Johnson, M., and McClosky, D. (2009). Improving Unsupervised Dependency Parsing with Richer Contexts and Smoothing. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL'09)*, pages 101–109, Boulder, Colorado.

Johnson, M., Geman, S., Canon, S., Chi, Z., and Riezler, S. (1999). Estimators for Stochastic "Unification-Based" Grammars. In *Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics (ACL'99)*, pages 535–541, College Park, MD, US.

Klein, D. and Manning, C. (2004). Corpus-Based Induction of Syntactic Structure: Models of Dependency and Constituency. In *Proceedings of the 42nd Meeting of the Association for Computational Linguistics (ACL'04)*, pages 478–485, Barcelona, Spain.

Klein, D. and Manning, C. D. (2003). Accurate unlexicalized parsing. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, pages 423–430, Sapporo, Japan.

Langkilde-Geary, I. (2000). Forest-based statistical sentence generation. In *Proceedings of the 1st North American chapter of the Association for Computational Linguistics conference*, pages 170–177. Morgan Kaufmann Publishers Inc. San Francisco, CA, USA.

Langkilde-Geary, I. (2002). An empirical verification of coverage and correctness for a general-purpose sentence generator. In *Proceedings of the 12th International Natural Language Generation Workshop*, pages 17–24. Citeseer.

Maxwell, J. T. and Kaplan, R. M. (1993). The Interface between Phrasal and Functional Constraints. *Computational Linguistics*, 19(4):571–590.

Müller, S. (2010). Persian Complex Predicates and the Limits of Inheritance-Based Analyses. *Journal of Linguistics*, 46(3):601–655.

Mutton, A., Dras, M., Wan, S., and Dale, R. (2007). GLEU: Automatic Evaluation of Sentence-Level Fluency. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics (ACL'07)*, pages 344–351, Prague, Czech Republic.

Naseem, T., Chen, H., Barzilay, R., and Johnson, M. (2010). Using Universal Linguistic Knowledge to Guide Grammar Induction. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing (EMNLP'10)*, pages 1234–1244, Cambridge, MA.

Post, M. and Gildea, D. (2008). Parsers as language models for statistical machine translation. In *Eighth Conference of the Association for Machine Translation in the Americas (AMTA 2008)*, Honolulu, HI.

Rajkumar, R. and White, M. (2010). Designing Agreement Features for Realization Ranking. In *Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010)*, pages 1032–1040, Beijing, China.

Riezler, S., King, T. H., Kaplan, R. M., Crouch, R., Maxwell III, J. T., and Johnson, M. (2002). Parsing the Wall Street Journal using a Lexical-Functional Grammar and Discriminative Estimation Techniques. In *Proceedings of the 40th Annual Meeting of the ACL*, number July, pages 271–278. Association for Computational Linguistics.

Riezler, S. and Vasserman, A. (2004). Gradient feature testing and l1 regularization for maximum entropy parsing. In *Proceedings of EMNLP'04*, Barcelona, Spain.

Rohrer, C. and Forst, M. (2006). Improving coverage and parsing quality of a large-scale LFG for German. In *LREC-2006*, Genoa, Italy.

Scannell, K. (2007). The Crúbadán Project: Corpus building for under-resourced languages. In Fairon, C., Naets, H., Kilgarriff, A., and de Schryver, G.-M., editors, *Building and Exploring Web Corpora: Proceedings of the 3rd Web as Corpus Workshop*, volume 4 of *Cahiers du Cental*, pages 5–15. Louvain-la-Neuve, Belgium.

Stolcke, A. (2002). SRILM – An Extensible Language Modeling Toolkit. In *Proc. Intl. Conf. on Spoken Language Processing*, pages 901–904, Denver, CO, US.

Velldal, E. and Oepen, S. (2005). Maximum entropy models for realization ranking. In *Proceedings of the 10th MTSummit X Phuket Thailand*. Citeseer.

Velldal, E. and Oepen, S. (2006). Statistical Ranking in Tactical Generation. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing (EMNLP'06)*, pages 517–525, Sydney, Australia.

Velldal, E., Oepen, S., and Flickinger, D. (2004). Paraphrasing treebanks for stochastic realization ranking. In *Proceedings of the 3rd Workshop on Treebanks and Linguistic Theories*, Tübingen, Germany.

Wang, Z. and Zong, C. (2011). Parse reranking based on higher-order lexical dependencies. In *Proceedings of 5th International Joint Conference on Natural Language Processing (IJCNLP'11)*, pages 1251–1259, Chiang Mai, Thailand.

White, M. and Rajkumar, R. (2009). Perceptron Reranking for CCG Realization. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, number August, pages 410–419. Association for Computational Linguistics.

White, M. and Rajkumar, R. (2012). Minimal Dependency Length in Realization Ranking. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP'12)*, pages 244–255, Jeju, Korea.

Yang, Y. and Pedersen, J. O. (1997). A comparative study on feature selection in text categorization. In *Proceedings of the Fourteenth International Conference on Machine Learning (ICML'97)*, pages 412–420.

Zeman, D., Mareček, D., Popel, M., Ramasamy, L., Štěpánek, J., Žabokrtský, Z., and Hajič, J. (2012). Hamledt: To parse or not to parse? In *Proceedings of the Eight International Conference on Language Resources and Evaluation (LREC'12)*, Istanbul, Turkey.

Analysis of Linguistic Style Accommodation in Online Debates

Arjun Mukherjee Bing Liu
Department of Computer Science
University of Illinois at Chicago
arjun4787@gmail.com liub@cs.uic.edu

ABSTRACT

Psycholinguistic phenomenon of communication accommodation (Giles et al., 1991) is probably one of the most important contributions in the interdisciplinary field of linguistics, psychology, information, and communication theory. Existing works have applied this theory to various domains like gesture, linguistics, backchannels, and even social media like tweets. In this work, we analyze the psycholinguistic phenomenon of linguistic style accommodation in online debates. First, we present a Joint Topic Expression (JTE) model for modeling debate posts and use it to generate our unique dataset for studying accommodation in debates. Specifically, we analyze the phenomenon across agreeing/disagreeing debating pairs generated using our JTE model. Second, we propose a formal framework for analyzing the linguistic phenomena of accommodation in online debates. Experiments on a large collection of real-life debate posts reveal very interesting insights about the complex phenomenon of psycholinguistic accommodation in online debates.

KEYWORDS : Linguistic style accommodation, linguistic convergence, accommodation in debates, online debate conversations.

Proceedings of COLING 2012: Technical Papers, pages 1831–1846,
COLING 2012, Mumbai, December 2012.

1 Introduction

The psycholinguistic theory of communication accommodation was developed by Howard Giles (Giles et al., 1991). It argues that "when people interact, they adjust their speech, their vocal patterns and their gestures, to accommodate to others". This adjustment or accommodation tends to occur unconsciously, i.e., people tend to instinctively converge to one another's communicative behavior. Over the past five decades, this phenomenon has received a great deal of attention across a myriad of domains: posture (Condon and Ogston, 1967), speech pause length (Jaffe and Feldstein, 1970), head nodding (Hale and Burgoon, 1984), generic linguistic style (Niederhoffer and Pennebaker, 2002), tweets (Danescu-Niculescu-Mizil et al., 2011), etc. This work presents a formal framework to model communication accommodation in online debates. Online debate forums are perhaps the most popular form of debates where people participate in discussions of various issues like politics, religions, society, human rights, etc. It is naturally very interesting to analyze the phenomenon of accommodation in debates.

In this work, we focus on linguistic style accommodation in debates. In detail, we will perform the following types of analysis: stylistic cohesion, stylistic accommodation, influence, and accommodation across both agreeing and disagreeing debate posts in online debates. We use the linguistic style markers in LIWC (Pennebaker et al., 2007) to measure the amount of linguistic accommodation exhibited. The underlying hypothesis behind the "measurement" of linguistic accommodation using linguistic style markers is based on the prior works in (Gonzales et al., 2010; Niederhoffer and Pennebaker, 2002; Taylor and Thomas, 2008), which have shown that linguistic accommodation being most pronounced in style dimensions is a good metric for measurement. Linguistic "style" here denotes content independent language constructs, i.e., *how* things are said as opposed *what* is said. Linguistic style has also been shown (Levelt and Kelter, 1982) to be exhibited somewhat unconsciously and hence it is an interesting target for analysis, especially in the domain of online debates. We will explain the meaning of these concepts in detail in the subsequent sections.

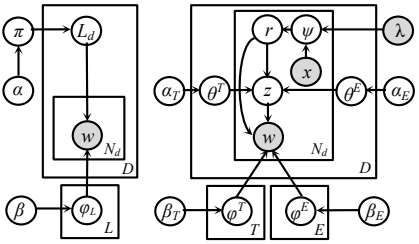
To perform these analyses, we need the right data. That is, we need to classify debate posts into those showing agreement and those showing disagreement. Given a large set of debate posts, this problem can be solved using supervised learning. Manually labeling of posts is also possible, but it is too time consuming because we will need to label a huge number of posts in order to ensure that we have enough data to produce statistically reliable results. We take a learning approach. However, the issue is the effective features that should be used for learning. An important characteristic of the debate posts is that they almost always use some specific expressions to express agreement or disagreement, e.g., "I agree," "you're correct," etc., for agreement and "I disagree," "you speak nonsense," etc., for disagreement. Discovering such expressions clearly help improve classification. Accurate classification is essential for our subsequent analysis.

We propose to use generative models for the discovery of such expressions and use them for classification. In fact, such models themselves can be used for classification directly too. In the next section, we propose the models for modeling debate posts, which include the Naïve Bayes model (both supervised and unsupervised) and the Joint Topic Expression (JTE) model. We also report classification results. Section 3 introduces the LIWC framework (Pennebaker et al., 2007). Section 4 presents our probabilistic framework where we analyze linguistic phenomenon like stylistic cohesion, accommodation, influence, and their effect across arguing nature of debating user pairs. Section 5 concludes our work.

2 Modeling debate posts for linguistic style analysis

We employ two generative models (Sections 2.1, 2.2) to accomplish the first task of debate post

classification and then generate the data for linguistic style experiments in Section 2.3. However, before proceeding, we briefly review related work on debates. Existing works have two major threads of research. The first thread puts debaters into support and oppose camps. Agrawal et al. (2003) used a graph method to place discussion participants into camps. Murakami and Raymond (2010) used a rule-based method to perform the same task. In (Somasundaran and Wiebe, 2009), opinions/polarities which were correlated with a debate-side were used to classify a post as for or against. However, this thread of research does not model agreements and disagreements in debates.



(a) Naïve Bayes (b) JTE

FIGURE 1: Graphical models in plate notations.

Another thread of research (Galley et al., 2004; Hillard et al., 2003; Thomas et al., 2006, Bansal et al., 2008; Burfoot et al., 2011) studies speaker interaction in the context of discourse and speech act classification of conversational speeches (e.g., U.S. Congress meeting transcripts). The above works mostly use three types of features: *durational* (e.g., time taken by a speaker, time separating two speakers, duration of speaker overlap, speech rate, etc.); *structural* (e.g., no. of speakers per side, no. of spurts with and without time overlap, no. of votes cast by a speaker on a bill, vote labels for and against the bill under discussion); and lexical (e.g., first word, last word, unigrams, *n*-grams, etc.) features to perform classification. While this is related to our approach of modeling agreeing and disagreeing debate posts, online debate forums (e.g., Volconco.com) are textual as opposed to conversational speeches. Thus, *durational* and *structural* features used in the prior works (e.g., time taken by a speaker, speech rate, speaker overlap, votes, etc.) are not directly applicable for our task.

Our approach relies on strong lexical features which we call AD-expressions. AD-expressions refer to Agreement (e.g., "I agree", "you're correct") and Disagreement (e.g., "I disagree", "you speak nonsense") expressions. As AD-expressions are an integral part of debates (because while arguing people invariably emit AD-expressions), our approach aims to first mine AD-expressions which serve as strong lexical features and further exploit them to classify debate discussions into agreeing and disagreeing posts. To model debate posts and lexical AD-expressions, we use hierarchical Bayesian generative models. Generative models like LDA (Blei et al., 2003) and PLSA (Hofmann, 1999) have been proved to be very successful in modeling topics and other textual information in an unsupervised manner. For our task of modeling and classifying debate posts we compare performance using two models. The first is the Naïve Bayes model (which serves as a baseline model) and the second is our Joint Topic Expression (JTE) model.

2.1 Naïve Bayes graphical model

This section introduces the well-known Naïve Bayes model in the light of unsupervised Bayesian graphical models. In generative models for text, words and phrases (*n*-grams) are viewed as random variables, and a document is viewed as a bag of *n*-grams and each *n*-gram takes a value from a predefined vocabulary. In this work, we use up to 4-grams, i.e., n = 1, 2, 3, 4. For simplicity, we use *terms* to denote both *words* (unigrams or 1-grams) and *phrases* (*n*-grams). We denote the entries in our vocabulary by $v_{1\ldots V}$ where V is the number of unique terms in the vocabulary. The entire corpus contains $d_{1\ldots D}$ documents. A document (e.g., debate post) d is represented as a vector of terms W_d with N_d entries. W is the set of all observed terms with

cardinality, $|W| = \sum_d N_d$. Also, let L_d denote the document class variable ($a$greeing or $d$isagreeing) we are trying to predict, i.e., $L_d = a$ or $L_d = d$. Lastly, let π denote the prior over document labels and φ_L the label specific distribution over vocabulary terms. Following Bayesian inference, our goal is precisely to choose L_d for W_d that maximizes $P(L_d|W_d)$. Applying Bayes rule, we get $L_d = \text{argmax}_L\, P(L|W_d) = \text{argmax}_L\, P(W_d|L)P(L)$. This lays the foundation for the generative process of the model (Figure 1a) which we detail as follows:

A. Draw $\pi \sim Beta(\alpha)$
B. For each label $L = \{a, d\}$, draw $\varphi_L \sim Dir(\beta)$
C. For each debate post $d \in \{1 \dots D\}$:
   i. Draw $L_d \sim Bernoulli(\pi)$
   ii. For each term $w_{d,j}, j \in \{1 \dots N_d\}$:
      a. Emit $w_{d,j} \sim Mult(\varphi_{L_d})$

To learn the model, we employ posterior inference using Monte Carlo Gibbs sampling. The samplers for L and φ_L are given as follows:

$$P(L_d = L|L_{\neg d}, W_{\neg d}, \varphi_L) \propto \frac{n_L + \alpha - 1}{D + 2\alpha - 1} \prod_{v=1}^{V} \left(\varphi_{L,v}\right)^{n_v^d} \quad (1)$$

$$\varphi_L \sim Dirichlet(n_v^L + \beta) \quad (2)$$

where n_L is the number of documents with label L, n_v^d is the number of times term v appears in document d, and n_v^L is the number of times term v appears in all documents with label L. Learning the model according to the Gibbs sampler in (1) and (2) results in a fully unsupervised Naïve Bayes model for document label (agreeing or disagreeing) prediction. However, if we have some labeled data (more details in Section 2.3), we can add supervision into the model using a simple trick. Given a set of labeled documents, D_{Train}, where each post has a document label (i.e., agreeing or disagreeing), we can employ a supervised Naïve Bayes model keeping the label variable, L_d of the training documents fixed to the supplied labels (i.e., we do not samples $L_d, d \in D_{Train}$). Fixing the labels will effectively serve the purpose of "ground truth" evidence for the distributions that created them.

2.2 JTE: A Graphical Model for Debates

We now present the Joint Topic Expression (JTE) model, which was proposed for analyzing debates in (Mukherjee and Liu, 2012). JTE is a hierarchical generative model motivated by the joint occurrence of various topics and AD-expressions in debate posts. A typical debate post mentions a few topics (using semantically related topical terms) and expresses some viewpoints with one or more AD-expression types (using semantically related expressions). This observation motivates the generative process of our model where documents (posts) are represented as random mixtures of latent topics and AD-expression types (Agreement and Disagreement).

Assume we have $t_{1\dots T}$ topics and $e_{1\dots E}$ expression types in our corpus. Note that in our case of Volconvo.com debate posts, based on reading various posts, we hypothesize that $E = 2$ as in such debates, we mostly find 2 expression types: Agreement and Disagreement[1]. Let $\psi_{d,j}$ denote the distribution over topics and AD-expressions with $r_{d,j} \in \{\hat{t}, \hat{e}\}$ denoting the binary indicator/switch variable (topic or AD-expression) for the j^{th} term of d, $w_{d,j}$. In this work, a document is viewed as a bag of n-grams and we use terms to denote both words (unigrams) and

[1] The hypothesis has been statistically validated using the perplexity metric in (Mukherjee and Liu, 2012). The model is however very general and can be used with any number of expression types, e.g., for modeling review comments in (Mukherjee and Liu, 2012a) with E = 6 expression types: Agreement, Disagreement, Thumbs-up, Thumbs-down, Question, and Answer-acknowledgement.

phrases (n-grams). $z_{d,j} \sim Mult(\theta_d)$ denotes the appropriate topic ($\theta^T_{d,t}$) or AD-expression type ($\theta^E_{d,e}$) index to which $w_{d,j}$belongs. Also let $\varphi^T_{t,v}$ and $\varphi^E_{e,v}$denote the topic and expression type specific multinomials over the vocabulary respectively. JTE is a switching graphical model performing a switch between expressions and topics similar to that in (Zhao et al., 2010). The switch is done using a maximum entropy (Max-Ent) model. The idea is due to the observation that topical and AD-expression terms usually play different syntactic roles in a sentence. Topical terms (e.g., "U.S. senate", "marriage", "income tax") tend to be noun and noun phrases while expression terms ("I refute", "how can you say", "probably agree") usually contain pronouns, verbs, wh-determiners, and modals. In order to utilize the part-of-speech (POS) tag information, we place the topic/AD-expression distribution $\psi_{d,j}$ (the prior over the indicator variable $r_{d,j}$) in the term plate (Figure 1) and set it from a Max-Ent model conditioned on the observed feature vector $\overrightarrow{x_{d,j}}$ associated with $w_{d,j}$ and the learned Max-Ent parameters λ. In this work, we encode both lexical and POS features of the previous, current and next POS tags/lexemes of the term $w_{d,j}$. More specifically, the feature vector is $\overrightarrow{x_{d,j}} = [POS_{w_{d,j}-1}, POS_{w_{d,j}}, POS_{w_{d,j}+1}, w_{d,j} - 1, w_{d,j}, w_{d,j} + 1]$. For phrasal terms (*n*-grams), all POS tags and lexemes of $w_{d,j}$ are considered as features. The generative process of JTE (Figure 1b) is given by:

A. For each C-expression type e, draw $\varphi^E_e \sim Dir(\beta_E)$
B. For each topic t, draw $\varphi^T_t \sim Dir(\beta_T)$
C. For each comment post $d \in \{1 \dots D\}$:
   i. Draw $\psi_d \sim Beta(\gamma \boldsymbol{u})$
   ii. Draw $\theta^E_d \sim Dir(\alpha_E)$
   iii. Draw $\theta^T_d \sim Dir(\alpha_T)$
   iv. For each term $w_{d,j}, j \in \{1 \dots N_d\}$:
      b. Draw $r_{d,j} \sim Bernoulli(\psi_d)$
      c. if $(r_{d,j} = \hat{e})$ // $w_{d,j}$is a C-expression term
            Draw $z_{d,j} \sim Mult(\theta^E_d)$
         else // $r_{d,j} = \hat{t}$, $w_{d,j}$is a topical term
            Draw $z_{d,j} \sim Mult(\theta^T_d)$
      d. Emit $w_{d,j} \sim Mult(\varphi^{r_{d,j}}_{z_{d,j}})$

We employ posterior inference using Monte Carlo Gibbs sampling. Denoting the random variables $\{w, z, r\}$ by singular subscripts$\{w_k, z_k, r_k\}$, $k_{1\dots K}$, where $K = \sum_d N_d$, a single iteration consists of performing the following sampling:

$$p(z_k = t, r_k = \hat{t} | W_{\neg k}, Z_{\neg k}, R_{\neg k}, w_k = v) \propto \frac{\exp(\sum_{i=1}^{n} \lambda_i f_i(x_{d,j}, \hat{t}))}{\sum_{y \in \{\hat{t}, \hat{e}\}} \exp(\sum_{i=1}^{n} \lambda_i f_i(x_{d,j}, y))} \times \frac{n^{DT}_{d,t\,\neg k} + \alpha_T}{n^{DT}_{d,(\cdot)\,\neg k} + T\alpha_T} \times \frac{n^{CT}_{t,v\,\neg k} + \beta_T}{n^{CT}_{t,(\cdot)\,\neg k} + V\beta_T} \quad (3)$$

$$p(z_k = e, r_k = \hat{e} | W_{\neg k}, Z_{\neg k}, R_{\neg k}, w_k = v) \propto \frac{\exp(\sum_{i=1}^{n} \lambda_i f_i(x_{d,j}, \hat{e}))}{\sum_{y \in \{\hat{t}, \hat{e}\}} \exp(\sum_{i=1}^{n} \lambda_i f_i(x_{d,j}, y))} \times \frac{n^{DE}_{d,e\,\neg k} + \alpha_E}{n^{DE}_{d,(\cdot)\,\neg k} + E\alpha_E} \times \frac{n^{CE}_{e,v\,\neg k} + \beta_E}{n^{CE}_{e,(\cdot)\,\neg k} + V\beta_E} \quad (4)$$

where $k = (d, j)$ denotes the j^{th} term of document d and the subscript $\neg k$ denotes assignments excluding the term at (d, j). Counts $n^{CT}_{t,v}$ and $n^{CE}_{e,v}$ denote the number of times term v was assigned to topic t and expression type e respectively. $n^{DT}_{d,t}$ and $n^{DE}_{d,e}$ denote the number of terms in document d that were assigned to topic t and AD-expression type e respectively. $\lambda_{1\dots n}$ are the parameters of the learned Max-Ent model corresponding to the n binary feature functions $f_{1\dots n}$ from Max-Ent. Omission of the latter index denoted by $(\cdot)$ represents the marginalized sum over the latter index. We employ a blocked sampler jointly sampling r and z as this improves convergence and reduces autocorrelation of the Gibbs sampler (Rosen-Zvi et al., 2004).

JTE (agreement expressions)	JTE (disagreement expressions)
agree, **I**, correct, yes, true, accept, I agree, indeed correct, **your**, **point**, I concede, is valid, **your claim**, **not really**, *would agree*, **might**, *agree completely*, yes indeed, absolutely, you're correct, *valid point*, **argument**, proves, *do accept*, support, agree with you, *rightly said*, **personally**, well put, *I do support*, *personally agree*, **doesn't necessarily**, exactly, *very well put*, absolutely correct, *kudos*, *point taken*...	**I**, disagree, I don't, I disagree, **argument**, reject, **claim**, I reject, I refute, I refuse, nonsense, *I contest*, dispute, **I think**, completely disagree, don't accept, don't agree, incorrect, *hogwash*, *I don't buy your*, *I really doubt*, your nonsense, **true**, *can you prove*, argument fails, *you fail to*, your assertions, *bullshit*, *sheer nonsense*, *doesn't make sense*, *you have no clue*, *how can you say*, *do you even*, *contradict yourself*, …

TABLE 1: Top terms (comma delimited) of two expression types. **Red (bold)** terms denote possible errors[2]. *Blue (italics)* terms are newly discovered; rest (black) were used in Max-Ent training.

2.3 Dataset Generation using Models

This section uses the models to classify agreeing and disagreeing debate posts which is a pre-requisite for this work. The hyper-parameters for the models were set to the heuristic values $\alpha = 1$, $\beta = 0.1$ for NB and $\alpha_T = 50/T$, $\alpha_E = 50/E$, $\beta_T = \beta_E = 0.1$ for JTE as suggested in (Griffiths and Steyvers, 2004). For both NB and JTE, we estimate model parameters using 5000 Gibbs iterations with a burn-in of 1000. To learn the Max-Ent parameters λ, we randomly sampled 500 terms from our corpus appearing at least 10 times[3] and labeled them as topical (361) or AD-expressions (139) and used the corresponding feature vector of each term (in the context of posts where it occurs) to train the Max-Ent model. Please note that this is term-level labeling which is very different from document labels or "tags" used in LabeledLDA (Ramage et al., 2009). LabeledLDA uses tagged data from del.icio.us setting the number of topics to the number of unique labels in the corpus. It restricts document-topic distributions to be defined only over the topics that correspond to the observed document-labels. For JTE, we induce $T = 100$ topics and $E = 2$ (agreement and disagreement) AD-expression types as in debate forums, there are usually two expression types. Values for $E > 2$ were also tried, but they did not produce any new dominant expression type. Instead, the expression types: *disagreement* and *agreement* became somewhat less specific as the expression-term ($\Phi^E_{E\times V}$) space became sparser. There was also slight increase in the model perplexity showing that values of $E > 2$ do not fit the data well.

Table 1 lists some top AD-expressions discovered by JTE. We see that JTE can cluster many correct AD-expressions, e.g., "I agree", "you're correct", "agree with you", etc. in agreement and "I disagree", "I refute", "don't accept", etc. in disagreement. In addition, it also discovers and clusters highly specific and more "distinctive" expressions beyond those used in Max-Ent training (marked *blue* in italics), e.g., "valid point", "rightly said", "I do support", and "very well put" in agreement; and phrases like "I don't buy your", "can you prove," "you fail to", and "you have no clue" in disagreement. We will later see that these AD-expressions serve as high quality lexical features for debate post classification. Note that we don't quantitatively evaluate topics, perplexity of the JTE model here as our focus is to classify agreeing and disagreeing posts using discovered AD-expression for our linguistic accommodation experiments on debates.

We now turn our attention to debate post classification. In this work, we use debate forum posts from Volconvo.com. We extracted 309376 debate posts from various domains like Politics, Religion, Society, Science, etc. To evaluate model performance, we construct a validation set. We randomly sampled 2000 posts from the corpus and asked two judges (CS grad students) to

[2] Clustering errors is a known issue with unsupervised generative models for text because the objective function of the model does not always correlate well with human judgments (Chang et al., 2009).
[3] A minimum frequency count of 10 ensures that the training data is representative of the corpus.

Feature Setting	Agreement			Disagreement		
	P	R	F_1	P	R	F_1
NB-unsupervised	0.69	0.65	0.67	0.71	0.69	0.70
NB-supervised	0.72	0.73	0.72	0.75	0.76	0.75
JTE-unsupervised	0.70	0.71	0.70	0.73	0.73	0.73
W+POS 1-4 grams + SVM (all terms)	0.75	0.76	0.75	0.80	0.81	0.80
W+POS 1-4 grams + SVM + χ^2 (top 1%)	0.79	0.77	0.78	0.84	0.84	0.84
W+POS 1-4 grams + SVM + χ^2 (top 2%)	0.80	0.78	0.79	0.85	0.85	0.85
AD-Expressions, Φ^E (top 1000) + SVM	0.84	0.81	0.82	0.88	0.86	0.87
AD-Expressions, Φ^E (top 2000) + SVM	0.86	0.83	0.84	0.88	0.87	0.87

TABLE 2: Precision (P), Recall (R) and F_1 scores of different models. Improvements in F_1 using AD-expression as features (Φ^E) are statistically significant (p<0.001) using paired t-test across 5-fold cross validation.

label the overall arguing nature of each post as agreeing or disagreeing or none[4]. We obtained strong agreement using κ_{Cohen} = 0.87. This is not surprising, as debate post classification is a fairly easy task and one can almost certainly make out whether a post expresses agreement or disagreement. Finally, we deemed a post as agreeing or disagreeing if both judges deemed it so. This yielded 1268 disagreement, 621 agreement posts. Out of the rest 111 posts, 39 are labeled "none" while 72 had no consensus among judges. We evaluate our models on the validation set, D_V, of 1268 disagreement and 621 agreement (1889) posts.

We consider the following classifiers:

i) NB-unsupervised, i.e., estimating the model (document labels) directly from D_V.
ii) NB-supervised, which performs 5-fold cross validation (CV) on D_V.
iii) JTE-unsupervised, which estimates the posterior on θ_d^E over D_V and classifies a post as agreeing if $\theta_{d,e=Agreement}^E > \theta_{d,e=Disagreement}^E$ else disagreeing. We call this unsupervised because although JTE uses Max-Ent term-level supervision for switching between topics and AD-expressions, it does not use the document-labels produced by judges.
iv) SVM + W+POS n-gram. We train a SVM classifier with the linear kernel[5] using standard word and POS n-gram features and 5-fold CV.
v) SVM + W+POS n-gram + χ^2. We extend (iv) by employing feature selection using Chi-Squared test[6].
vi) SVM+AD-expressions. We induce a SVM classifier using AD-expressions as features over 5-fold CV.

For unsupervised learners (no learning), we compute precision and recall on the corresponding bin of testing for 5-fold CV. For feature selection using χ^2, and AD-expressions (as they are basically rankings from φ_e^E), we try two settings: top 1% and 2% features. Results across agreement and disagreement posts are summarized in Table 2. For SVM, we used SVM^{light} (Joachims, 1999). We see that AD-expressions+SVM performs the best. This shows that AD-expressions discovered by JTE are of high quality. Next in order is SVM + χ^2. This shows that feature selection (FS) is useful. AD-expressions can be thought of as an FS scheme where a set

[4] First posts of thread who start a topic, ambiguous, vague, partly agreeing/disagreeing posts, etc. belong to the "none" category.
[5] Polynomial, RBF, and sigmoid kernels were tired but yielded poorer results hence not reported. Linear kernel has been shown very effective for text classification problems by many researchers, e.g., (Joachims, 1998).
[6] We also tried other feature selection schemes like Information Gain, Mutual information. However, they yielded poorer results than Chi-Squared test and hence not reported.

Dimension	Examples	Size
Article	a, an , the	3
Certainty	always, never	83
Conjunction	and, but, whereas	28
Discrepancy	should, could, would	76
Exclusive	but, without, exclude	17
Inclusive	and, with, include	18
Indefinite Pronoun (Indef-Pron.)	it, those, it's	46
Negation	no, not, never	57
Preposition	to, with, above	60
Quantifier	few, many, several	89
Tentative	maybe, guess, perhaps	155
1st Person Singular Pronoun (1st-S-Pron.)	I, me, mine	12
1st Person Plural Pronoun (1st-P-Pron.)	we, our, us	12
2nd Person Pronoun (2nd-Pron.)	you, your, thou	20

TABLE 3: LIWC Style Dimensions.

of highly discriminative lexical features are selected using JTE. It is understandable that the unsupervised methods are inferior to the supervised baselines. But JTE does attain a respectable F_1 of 0.70 for agreement and 0.73 for disagreement and is better than NB-unsupervised.

We now turn to our task of generating the debate dataset (agreeing and disagreeing posts) for linguistic accommodation study. While the ideal situation would involve manually labeling all 309376 debate posts under study, it is impractical. Hence we resort to SVM+AD-expression as our classifier. Since the labeled data contains three categories, we train a multiclass SVM using our labeled data: agreement (621), disagreement (1268), and none (39) with AD-expressions. Classification on our debate corpus resulted in 123751 agreement, 177087 disagreement, and 8538 none (e.g., first posts of thread that start a topic, ambiguous, vague, partly agreeing/disagreeing posts, etc.) posts. While this classification is not perfect and may have some noise, labels on our unlabeled debate posts are sufficiently reliable as the confidence of the classifier (SVM+AD-expression) is reasonably high on the validation set. Our database consists of 7973 authors and 4387 author pairs who have debated/interacted with each other [7] and 6828 discussion threads. We now proceed to linguistic accommodation experiments.

3 LIWC: A metric for Linguistic Style

To study the general phenomenon of linguistic accommodation in debates, we need a metric for linguistic style. Following prior work on linguistic accommodation in stylometry and psycholinguistics (Niederhoffer and Pennebaker, 2002; Taylor and Thomas; 2008), we use the psycholinguistic framework Linguistic Inquiry and Word Count (LIWC) (Pennebaker et al., 2007). LIWC measures word usage in psychologically meaningful style dimensions (e.g., articles, pronouns, emotion words, etc.) and has been proven useful in the analysis of personality (Yee et al., 2010); gender, age (Mukherjee and Liu, 2010; Argamon et al., 2007); deceptive opinions (Ott et al., 2011); social relations (Scholand et al., 2010), etc. In this work, we focus on 14 strictly non-topical style dimensions detailed in Table 3 [8], i.e., we study the linguistic phenomenon of accommodation in debates over those 14 style dimensions. Please refer to (Pennebaker et al., 2007) for full list of terms. A debate post is said to exhibit a style dimension if it contains at least one word form that respective LIWC category.

[7] As it may not be interesting to study linguistic accommodation across pairs who interacted only a few times, we only consider pairs who interacted at least 20 times.
[8] Other dimensions like Family, Sexuality, Religion, etc. do not convey any style information.

4 Probabilistic Framework

This section introduces a probabilistic framework to model the linguistic phenomenon of accommodation in a principled manner.

4.1 Stylistic Cohesion

Stylistic cohesion is the general phenomenon which is grounded on the following hypothesis: Related conversations tend to be stylistically closer (hence the nomenclature, cohesion) than unrelated conversations. In the context of online debates, this transforms as follows: Related debate posts (i.e., post pairs comprising of the original post, say d and another post, say r which quotes or replies to d. Related debate posts are denoted by $d \leftrightarrow r$ from now on) exhibit significantly higher stylistic cohesion than unrelated posts. Formally, for a given style dimension, s, we can measure stylistic cohesion on s using the following probabilistic expression:

$$Coh(s) \triangleq P(d^s \wedge r^s | d \leftrightarrow r) - P(d^s \wedge r^s | d \nleftrightarrow r) \quad (5)$$

where d^s , r^s denote the event that debate posts d, r respectively exhibit style dimension s. Thus, statistically, if the former probability expression in Eq. (5) tends to be greater than the latter, we say that related debate posts $d \leftrightarrow r$ tend to "agree" on the style dimension s. $d \nleftrightarrow r$ denotes that d and r do not form a conversation pair. Before proceeding, it is worthwhile to test the hypothesis on our debate domain. Establishing that stylistic cohesion is exhibited in online debates corresponds to rejecting the null hypothesis that the two probabilities in Eq. (5) are equal. A two tailed *t*-test rejects the null hypothesis with p-value < 0.001 for all 14 style dimensions in Table 4. Table 4 (a) shows the differences of expected probabilities[9] across each style dimension over all posts in our debate database.

Having established that stylistic cohesion is exhibited in online debates, we now turn our

s	$P(d^s \wedge r^s \mid d \leftrightarrow r)$	$P(d^s \wedge r^s \mid d \nleftrightarrow r)$	$Coh(s)$
Article	0.295	0.271	0.024*
Certainty	0.042	0.034	0.008**
Conjunction	0.212	0.176	0.036*
Discrepancy	0.069	0.062	0.007**
Exclusive	0.074	0.068	0.006**
Inclusive	0.238	0.223	0.015*
Indef-Pron.	0.278	0.261	0.017*
Negation	0.157	0.134	0.023*
Preposition	0.342	0.315	0.027*
Quantifier	0.076	0.067	0.009**
Tentative	0.097	0.091	0.006**
1st-S-Pron.	0.221	0.201	0.02*
1st-P-Pron.	0.019	0.009	0.01*
2nd-Pron.	0.124	0.120	0.004**

TABLE 4 (a)

$Coh_{Agree}(s)$	$Coh_{Diagree}(s)$
0.021	0.016
0.007	0.002
0.034	0.028
0.005	0.002
0.005	0.003
0.012	0.007
0.014	0.010
0.017	**0.019**
0.026	0.022
0.005	0.003
0.003	0.002
0.018	0.015
0.007	0.003
0.003	0.002

TABLE 4 (b)

Table 4: (a)**:** Effect of stylistic cohesion across each style dimension. The differences are statistically significant (*: p<0.0001 **: p<0.001) over two-tailed *t*-test. (b): Cohesion across agreeing and disagreeing debate discussions. Differences are significant p<0.001.

[9] The expectation was taken over discussion threads, i.e., the probabilities in Eq. (5) were computed for each thread and averaged over all threads in our database. We do so because $|d \nleftrightarrow r|$ is very large $\approx \binom{309376}{2}$, for our database.

attention to the analysis of stylistic cohesion across agreeing and disagreeing debate posts. Let $d \xleftarrow{Agree} r$ denote the post pair where the post r is an agreement post and it quotes/replies to d. Analogously, we have $d \xleftarrow{Disagree} r$ when post r is a disagreement post. Extending the definition of (5), stylistic cohesion in related posts expressing agreement, $Coh_{Agree}$is given by:

$$Coh_{Agree}(s) \triangleq P\left(d^s \wedge r^s \middle| d \xleftarrow{Agree} r\right) - P(d^s \wedge r^s | d \leftrightarrow r) \quad (6)$$

and $Coh_{Diagree}$ is given by:

$$Coh_{Disagree}(s) \triangleq P\left(d^s \wedge r^s \middle| d \xleftarrow{Disagree} r\right) - P(d^s \wedge r^s | d \leftrightarrow r) \quad (7)$$

Table 4 (b) compares Coh_{Agree} and $Coh_{Diagree}$ across 14 style dimensions. We find that cohesion across agreeing conversations is significantly ($p<0.001$) more than cohesion in disagreeing conversations except for the style dimension Negation. This gives a very interesting insight to the phenomenon. While it is intuitive that agreeing discussions tend to have more cohesion for most style dimensions, the situation becomes reversed for negation. We believe this is so because owing to the very nature of debates, when people disagree over conversations (i.e., chain of post interactions debating via arguing and disagreeing), they try to negate the views of the other partner resulting in more cohesion.

4.2 Stylistic Accommodation

We now throw light on our key objective: linguistic style accommodation in debates. The theory of communication accommodation in linguistics (Giles et al., 1991) hinges on the general observation that during conversations/communications both textual and spoken, people try to unconsciously converge to one another's communicative behavior. In other words, there exists some coordination among conversers across a variety of dimensions like words, syntax, style, etc. Extending our probabilistic framework, we measure accommodation of a user b to another user a (where a and b are conversing pairs, i.e., they interact via reply/quote relations) as whether the stylistic dimension s in the initial post (of user a) increases the probability of s in the reply (of user b) beyond what is normally expected from user b. Formally:

$$Acc_{(a \leftarrow b)}(s) \triangleq P(d_b^s | d_a^s, d_a \curvearrowright d_b) - P(d_b^s | \, d_a \curvearrowright d_b) \quad (8)$$

where d_a^s and d_b^s denote the event that posts d_a and $d_b$exhibit style dimension s respectively by users a and b. $d_a \curvearrowright d_b$ denotes the reply/quote relation from b to a. We want to emphasize the temporal aspect ($\curvearrowright$ relation) of our modeling which offers us two crucial advantages. First, accounting the temporal aspect of accommodation (i.e., a user can accommodate to his/her conversation partner only after receiving his/her input), we minimize the effects of background style similarity like homophily. Secondly, encoding the temporal aspect gives a richer formulation than prior works (Niederhoffer and Pennebaker, 2002; Taylor and Thomas; 2008) which used correlation based measures[10]. Eq. (8) defines pairwise accommodation and can be extended to compute global accommodation for a given style dimension, s by taking the expectation over all conversing pairs, i.e.,$Acc(s) = E[Acc_{(a \leftarrow b)}(s)]$.

Establishing that linguistic phenomenon of stylistic accommodation is observable for a given dimension s is reduced to showing that $Acc(s) > 0$. Denoting $\bar{P}(d_b^s | d_a^s, d_a \curvearrowright d_b)$ as the expected value of the subtrahend and $\bar{P}(d_b^s | \, d_a \curvearrowright d_b)$ as the expected value of the minuend in Eq. (8),

[10] This is so because, correlation based measures do not distinguish between the case when the initial post exhibits a style dimension *s* but the reply/quote does not, and the reverse case when the initial post does not exhibit *s* but the reply does.

Table 6 (a) shows the differences in these means. The expectation is taken over all ordered pairs in our database. Differences between these means are statistically significant using a two-tailed *t*-test for all style dimensions (except 2nd person pronoun and 1st person singular pronoun. See caption of Table 6 (a)). It is clear from Table 6 (a) that accommodation exhibits significantly across major style dimensions. In fact, we find the highest accommodation in the *negation* style dimension. This is intuitive as debates usually involve a lot of negation (especially while disagreeing, more details follow in the next section).

Validating the existence of accommodation in online debates paves the way for comparative analysis of accommodation across agreeing and disagreeing pairs in debates. We define the notion of mutual accommodation $Acc_{(a,b)}(s)$ of a pair (a, b)as the expected accommodation of each with the other:

$$Acc_{(a,b)}(s) = \frac{1}{2}\left[Acc_{(b \leftarrow a)}(s) + Acc_{(a \leftarrow b)}(s)\right] \quad (9)$$

With mutual accommodation of a pair defined, we now analyze mutual accommodation across agreeing and disagreeing pairs. To perform this experiment, it is required to classify the 4387 pairs in our database into agreeing and disagreeing pairs. Before proceeding, we note the following important aspect of agreement and disagreement in debates. It reflects the intuition that when a user (say a) mostly agrees with the views of his conversing partner b (i.e., (a, b) form a conversing pair), b also agrees (or at least does not completely disagree) with a. Similarly, if a mostly disagrees with the views of b, it is highly unlikely that b completely agrees with a, i.e., b also inherently disagrees or at least does not completely agree with a. This hypothesis is grounded on the very human psychological nature of debating and arguing with others. Building on this intuition of human arguing/debating nature, it is reasonable to deem a pair of users as:

1. agreeing, if more than *k*% of their interactions (posts) exhibit agreement.
2. disagreeing, if more than *k*% of their interactions (posts) exhibit disagreement.
3. mixed, (i.e., partly agreeing and partly disagreeing), otherwise.

Choosing the threshold k is somewhat subjective as the definition of agreeing and disagreeing pairs may have different degrees of strictness according to end users. In this work, we experiment with two thresholds k = 65% and k = 75%. The thresholds are reasonable because a threshold of 50% says that the pair is mixed while if any one of the arguing nature (agreeing or disagreeing) is more pronounced then that nature is dominant. As we have the labels (agreeing or disagreeing) for each post (using our classifier in Section 2), each pair in our database was classified as agreeing, disagreeing or mixed according to the above scheme. It resulted in the following split:

k	Agreeing	Disagreeing	Mixed
0.65	1360 (31%)	2588 (59%)	439 (10%)
0.75	1141 (26%)	2676 (61%)	570 (13%)

TABLE 5: Distribution of split according to two thresholds.

As it may not be interesting to study accommodation over pairs with mixed nature (they also comprise a relatively small percentage), we focus on agreeing and disagreeing pairs. Table 6(b) shows the difference in average mutual accommodation over agreeing ($\overline{Acc}_{(a,b)}^{Agree}(s)$) and disagreeing ($\overline{Acc}_{(a,b)}^{Disagree}(s)$) pairs across each style dimension using the threshold k = 0.75. Table 6 (c) reports the corresponding results using the threshold k = 0.65. From Tables 6 (b), we note the following. For most style dimensions, mutual accommodation across agreeing pairs is more than that for disagreeing pairs. However, for 4 style dimensions, *negation*, *exclusive*, *discrepancy*, and *2nd person pronoun*, we find the trend reversed (values marked in bold).

s	$\bar{P}(d_b^s \mid d_a^s, d_a \leftrightarrow d_b)$	$\bar{P}(d_b^s \mid d_a \leftrightarrow d_b)$	$Acc(s)$
Article	0.376	0.343	0.033*
Certainty	0.145	0.113	0.032*
Conjunction	0.283	0.247	0.036*
Discrepancy	0.261	0.202	0.059*
Exclusive	0.243	0.198	0.045*
Inclusive	0.309	0.301	0.008**
Indef-Pron.	0.278	0.261	0.017*
Negation	0.257	0.178	**0.079***
Preposition	0.365	0.332	0.033*
Quantifier	0.213	0.201	0.012**
Tentative	0.176	0.169	0.007**
1st-S-Pron.	0.322	0.320	0.002
1st-P-Pron.	0.336	0.318	0.018**
2nd-Pron.	0.251	0.247	0.004

(a)

$\overline{Acc}_{(a,b)}^{Agree}(s)$	$\overline{Acc}_{(a,b)}^{Disagree}(s)$
0.019	0.015
0.021	0.019
0.027	0.025
0.017	**0.021**
0.021	**0.026**
0.005	0.003
0.011	0.007
0.035	**0.042**
0.021	0.018
0.007	0.004
0.004	0.003
0.0009	0.0006
0.011	0.008
0.001	**0.003**

(b)

$\overline{Acc}_{(a,b)}^{Agree}(s)$	$\overline{Acc}_{(a,b)}^{Disagree}(s)$
0.021	0.018
0.024	0.023
0.026	0.025
0.013	**0.016**
0.018	**0.021**
0.004	0.003
0.015	0.012
0.017	**0.023**
0.023	0.021
0.006	0.005
0.0059	0.0050
0.0008	0.0006
0.013	0.011
0.0013	**0.00018**

(c)

TABLE 6: (a): Effect of accommodation across each style dimension. The differences are statistically significant (*: p<0.0001 **: p<0.001) over two-tailed *t*-test. Table 6 (b, c): Average mutual accommodation over agreeing and disagreeing pairs using two different thresholds, *k*: Table 6 (b): k = 0.75 Table 6 (c): k = 0.65.

Disagreeing pairs happen to accommodate more. The effect is most pronounced for *negation* style dimension. We believe this is because disagreeing pairs in debates invariably emit the above 4 style dimensions to other partners who in turn also emit the same style dimensions to counter/debate. To get an intuitive feeling, we list some of the frequent expressions among disagreeing posts as follows: "**your** claim **should**", "I **would** disagree", "**you cannot exclude**", "**without** knowing", "**you** do **not**", "**I don't** accept **your**", etc. We mark the words in **red** (**bold**) which appear in the above mentioned 4 style dimensions. Lastly, we note that the results follow a similar trend for the threshold k = 0.65 used to split agree/disagree pairs (Table 6(c)). This renders confidence in our results. Also interesting to note is that the mutual accommodation differences between agreeing and disagreeing pairs have reduced when k = 0.65 (Table 6(c)) than the results using k = 0.75 (Table 6 (b)). This is not surprising as the agree/disagree pair split using k = 0.75 creates a better demarcation of pairs based on their arguing nature.

4.3 Stylistic Influence

Linguistic accommodation has a unique characteristic of asymmetry, i.e., the accommodation of a user b to another user a over a style dimension s, $Acc_{(a \leftarrow b)}(s)$ is potentially different from the accommodation of a to b over the dimension s, $Acc_{(b \leftarrow a)}(s)$. This gives rise to the notion of stylistic influence. Theoretically, considering the following probabilistic expression:

$$I_{a,b}(s) = |Acc_{(a \leftarrow b)}(s) - Acc_{(b \leftarrow a)}(s)| \quad (10)$$

when $I_{a,b}(s) > 0$, we have the following two *exclusive* cases:

Case 1: $Acc_{(a \leftarrow b)}(s) > Acc_{(b \leftarrow a)}(s)$
Case 2: $Acc_{(b \leftarrow a)}(s) > Acc_{(a \leftarrow b)}(s)$.

However, both cannot happen simultaneously. In either case, it implies that there is a difference in the amount one user accommodates to the other. Put in other words, one of the users has an "influence" over the other. If case 1 holds, then b accommodates to a more than a does to b on

style dimension s, i.e., a is influencing b to emit style dimension s. If case 2 holds, then b is influencing a to emit style dimension s.

Before proceeding to analyze this interesting and fine-grained linguistic phenomenon of stylistic influence in debates, we need to statistically validate the existence of stylistic influence in online debates. Precisely, we are interested in answering the following question: In general, across various conversing pairs, is one of the users (former) forming the pair stylistically influencing (or causing the other user (latter) to accommodate) more than the extent to which he/she (the former user) is accommodating to him/her (the latter user)?

s	SA (%)	AS (%)	DA (%)	NA (%)
Article	32	32	35	1
Certainty	40	24	34	2
Conjunction	47	22	29	2
Discrepancy	52	25	22	1
Exclusive	46	21	31	2
Inclusive	38	32	29	1
Indef-Pron.	57	13	28	2
Negation	**42**	30	26	2
Preposition	40	33	26	1
Quantifier	27	44	28	1
Tentative	41	27	30	2
1st-S-Pron.	35	26	38	1
1st-P-Pron.	62	14	22	2
2nd-Pron.	**33**	40	26	1

(a): Agreeing

SA (%)	AS (%)	DA (%)	NA (%)
29	21	48	2
33	22	41	4
42	23	33	2
49	19	29	3
38	23	35	4
32	32	33	3
54	15	29	2
53	7	38	2
35	32	30	3
25	39	34	2
39	27	32	2
33	25	41	1
58	8	31	3
39	41	18	2

(b) Disagreeing

TABLE 7: Percentage of Agreeing (TABLE 7 (a)) and Disagreeing (TABLE 7(b)) pairs exhibiting different types of accommodation.

Answering the above question is reduced to the following statistical test: can we reject the null hypothesis that $E\left[I_{a,b}(s)\right] = 0$? Put in simple language, whether in expectation there is an imbalance (in either way[11]) between the amounts of accommodation among users in a conversing pair (this is the alternate hypothesis) or there is balance and each user accommodates to the other in almost the same extent in expectation (this is the null hypothesis). The key term here is "in expectation". Establishing that stylistic influence is exhibited in debates corresponds to rejecting the null hypothesis, H_1: $E\left[I_{a,b}(s)\right] = 0$. As $I_{a,b}(s)$ is an absolute value as defined in Eq. (10), we also need to test the hypothesis, H_2:

$$E\left[\max\left(Acc_{(a\leftarrow b)}(s), Acc_{(b\leftarrow a)}(s)\right)\right] = E\left[\min\left(Acc_{(a\leftarrow b)}(s), Acc_{(b\leftarrow a)}(s)\right)\right]$$

We subject H_1 and H_2 to a paired t-test over all pairs in our database. Paired t-test rejects both H_1 (with p-value < 0.0001) and H_2 (with p-value < 0.002) for all style dimensions. This empirically validates that stylistic influence is exhibited in online debates.

4.4 Accommodation across Arguing Nature

Having established that stylistic influence is exhibited in online debates, we now further investigate this intriguing phenomenon across arguing nature.

In psycholinguistic literature (Giles et al., 1991), it has been observed that when accommodation is exhibited, it can occur symmetrically (when both partners accommodate to each other) or asymmetrically (when only one of the conversers accommodates). Using our probabilistic framework, given a pair of interacting/debating user pair (a, b), the following cases arise for a style dimension s:

Case 1: Symmetry: When both $Acc_{(a\leftarrow b)}(s) > 0$ and $Acc_{(b\leftarrow a)}(s) > 0$, i.e., both of the conversers accommodate to each other.

Case 2: Asymmetry: When only one of $Acc_{(a\leftarrow b)}(s)$ or $Acc_{(b\leftarrow a)}(s)$ is > 0, i.e., only one

[11] Hence we need to employ a two-tailed t-test for testing the hypothesis.

accommodates. This further gives rise to the following two subcases. Say $Acc_{(a \leftarrow b)}(s) > 0$, i.e., b accommodates to a, then we can have:

Case 2 (a): Default asymmetry: The other non-accommodating converser maintains his "default" behavior, i.e., $Acc_{(b \leftarrow a)}(s) = 0$.

Case 2 (b): Divergent asymmetry: The non-accommodating converser accentuates his communication behavior in the opposite direction, i.e., *diverges* and $Acc_{(b \leftarrow a)}(s) < 0$

Case 3 No accommodation: None of the conversers accommodates, i.e., both $Acc_{(a \leftarrow b)}(s)$ and $Acc_{(b \leftarrow a)}(s)$ are ≤ 0.

To investigate the above cases, we compute the percentage of various forms of accommodation mentioned above across agreeing and disagreeing debating pairs in Table 7. For nomenclature, we use the following acronyms: Symmetric accommodation (SA), Default asymmetry (AS), Divergent asymmetry (DA), No accommodation (NA). However, we report results for agree/disagree pair split using threshold $k = 0.75$ (see Table 5) only as split using a higher threshold ensures better demarcation of agreeing/disagreeing pairs. We note the following interesting observations from Table 7 (a, b):

i) From column SA, we find that among agreeing pairs, percentage of pairs exhibiting symmetric accommodation (i.e., both members of a pair accommodating to each other) is more than that for disagreeing pairs. However, for style dimensions *negation* and *2nd person pronoun*, percentage of symmetric accommodating pairs among disagreeing pairs is more than that in agreeing pairs (shown in bold in SA column). The reason can be linked to the similar phenomenon in Section 4.2, i.e., disagreeing pairs in debates invariably emit style dimensions like *negation* and *2nd person pronoun* to other partners who in turn also emit the same style dimensions in order to counter/debate eventually resulting in somewhat symmetric accommodation.

iii) From column DA, we find that percentage of pairs exhibiting divergent asymmetry is more in disagreeing posts than agreeing posts. This is intuitive as divergent asymmetry calls for the non-accommodating converser to accentuate his communication behavior in the opposite direction so as to signal a stylistic "disagreement" along with a disagreement of views.

iv) Percentage of non-accommodating pairs among disagreeing pairs is in general more than that in agreeing pairs (See column NA in Table 7 (a, b)). This is reasonable and a plausible reason for such phenomenon is that pairs express "disagreement" in linguistic style by not accommodating at all. However, it is important here to note the following point. Earlier in Section 4.2, we showed that $Acc(s) > 0$ and accommodation is expressed in online debates. But in Table 7 we find that there are some pairs with $Acc(s) \leq 0$. It should not be considered as a contradiction to our results in Section 4.2. The key point is that we are interested in the "expected" accommodation over pairs and $E[Acc_{(a \leftarrow b)}(s)] > 0$ for all style dimensions.

Lastly, we note that the above experiments reveal no specific trend for percentage of pair exhibiting default asymmetry (DA) among agreeing and disagreeing posts based on our dataset of debate posts from Volconvo.com.

5 Conclusion

This paper studied the sociolinguistic phenomenon of accommodation in online debates. It first discussed a graphical model to perform debate post analysis to generate the required data for linguistic experiments. It then carried out a comprehensive analysis of various complex linguistic phenomena like stylistic cohesion, stylistic accommodation, influence, and accommodation across both agreeing and disagreeing debate posts. Several interesting results were obtained which dovetail with the intuitive psychology of online debaters, i.e., agreement and disagreement are also exhibited in the "style" dimension (beyond mere content) using symmetric and divergent asymmetric accommodation respectively. To our knowledge, this is the first study to report such fine-grained analysis of the linguistic phenomenon of accommodation in online debates. All experimental results were empirically validated using a large number of real-life debate posts.

References

Agrawal, R.; Rajagopalan, S.; Srikant, R.; and Xu. Y. (2003). Mining newsgroups using networks arising from social behavior. Proceedings of the International World-Wide Web Conference (WWW-2003).

Argamon, S., Koppel, M., Pennebaker, J. W., Schler, J. (2007). Mining the Blogosphere: Age, Gender and the varieties of self-expression, First Monday, 2007 - firstmonday.org

Bansal, M., Cardie, C., and Lee, L. (2008). The power of negative thinking: Exploiting label disagreement in the min-cut classification framework. In Proceedings of the International Conference on Computational Linguistics (Coling-2008): Companion volume: Posters.

Blei, D.; Ng, A.; and Jordan, M.; (2003). Latent Dirichlet Allocation. Journal of Machine Learning Research (JMLR).

Burfoot, C.; Bird, S.; and Baldwin, T. (2011). Collective Classification of Congressional Floor-Debate Transcripts. Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics (ACL-2011).

Chang, J., Boyd-Graber, J., Wang, C. Gerrish, S. Blei, D. (2009). Reading tea leaves: How humans interpret topic models. Proceedings of the Neural Information Processing Systems (NIPS-2009).

Cristian Danescu-Niculescu-Mizil, Michael Gamon, and Susan Dumais. (2011). Mark my words! Linguistic style accommodation in social media. Proceedings of the International World-Wide Web Conference (WWW-2011).

Condon and Ogston. (1973). A segmentation of behavior. Journal of psychiatric research.

Galley, M.; McKeown, K.; Hirschberg, J.; and Shriberg, E. (2004). Identifying agreement and disagreement in conversational speech: Use of Bayesian networks to model pragmatic dependencies. Proceedings of Annual Meeting of the Association for Computational Linguistics (ACL-2004).

Gonzales, A. L., J. T. Hancock, and J. W. Pennebaker. (2010). Language style matching as a predictor of social dynamics in small groups. Communication Research, 37(1):3.

Giles, H. J. Coupland, and N. Coupland. (1991). Accommodation theory: Communication, context, and consequences. In Contexts of accommodation: developments in applied sociolinguistics. Cambridge University Press, 1991.

Hale, J. and J. Burgoon. (1984). Models of reactions to changes in nonverbal immediacy. Journal of Nonverbal Behavior, 8(4):287.

Hillard, D., Ostendorf, M., and Shriberg, E. (2003). Detection of agreement vs. disagreement in meetings: Training with unlabeled data. Proceedings of Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT-2003).

Hofmann, T. (1999). Probabilistic latent semantic analysis. Proceedings of Conference on Uncertainty in Artificial Intelligence (UAI-1999).

Jaffe, J. and S. Feldstein. (1970). Rhythms of dialogue. Academic Press.

Joachims, T. (1998). Text Categorization with Support Vector Machines: Learning with Many Relevant Features. European Conference on Machine Learning (ECML-1998).

Joachims, T. Making large-Scale SVM Learning Practical. (1999). Advances in Kernel Methods - Support Vector Learning, B. Schölkopf and C. Burges and A. Smola (ed.), MIT-Press, 1999.

Levelt, W. and S. Kelter. (1982). Surface form and memory in question answering. Cognitive Psychology, 14(1):78.

Murakami A.; and Raymond, R. (2010). Support or Oppose? Classifying Positions in Online Debates from Reply Activities and Opinion Expressions. In Proceedings of the International Conference on Computational Linguistics (Coling-2010).

Mukherjee, A. and Liu, B. (2010). Improving gender classification of blog authors. Empirical Methods in Natural Language Processing (EMNLP-2010).

Mukherjee, A. and Liu, B. (2012). Mining Contentions from Discussions and Debates. KDD-2012.

Mukherjee, A. and Liu, B. (2012a). Modeling Review Comments. ACL-2012.

Niederhofer, K. and J. Pennebaker. (2002). Linguistic style matching in social interaction. Journal of Language and Social Psychology.

Ott M., Choi Y., Cardie C., Hancock J. T. (2011). Finding deceptive opinion spam by any stretch of the imagination, Proc. of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL-HLT-2011).

Pennebaker, J. W., R. J. Booth, and M. E. Francis. (2007). Linguistic Inquiry and Word Count (LIWC): A computerized text analysis program. LIWC.net, 2007.

Ramage, D.; Hall, D.; Nallapati, R; Manning, C. (2009). Labeled LDA: A supervised topic model for credit attribution in multi-labeled corpora. Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing (EMNLP-2009).

Rosen-Zvi, M.; Griffiths, T.; Steyvers, M.; and Smith, P. (2004). The author-topic model for authors and documents. Proceedings of Conference on Uncertainty in Artificial Intelligence (UAI-2004).

Somasundaran, S. and Wiebe, J. (2009). Recognizing stances in online debates. Proceedings of Annual Meeting of the Association for Computational Linguistics (ACL-2009).

Scholand, J. A., Tausczik, Y. R. and Pennebaker, J. W. (2010). Social language network analysis. In Proceedings of CSCW, pages 23–26, 2010.

Taylor, P. and S. Thomas. (2008). Linguistic style matching and negotiation outcome. Negotiation and Conict Management Research, 1(3):263.

Yee, N., Harris, H., Jabon, M. and Bailenson, J. (2010). The Expression of Personality in Virtual Worlds. Social Psychological & Personality Science (in press).

Zhao, X., Jiang, J. Yan, H., Li, X. (2010). Jointly modeling aspects and opinions with a MaxEnt-LDA hybrid. EMNLP. 2010.

Sentiment Analysis in Twitter with Lightweight Discourse Analysis

Subhabrata Mukherjee[†], *Pushpak Bhattacharyya*[‡]
[†]IBM India Research Lab
[‡]Dept. of Computer Science and Engineering, IIT Bombay
subhabmu@in.ibm.com, pb@cse.iitb.ac.in

ABSTRACT

We propose a lightweight method for using discourse relations for polarity detection of *tweets*. This method is targeted towards the web-based applications that deal with *noisy, unstructured* text, like the *tweets*, and cannot afford to use heavy linguistic resources like *parsing* due to frequent failure of the parsers to handle noisy data. Most of the works in micro-blogs, like *Twitter*, use a bag-of-words model that ignores the discourse particles like *but, since, although etc*. In this work, we show how the discourse relations like the *connectives* and *conditionals* can be used to incorporate discourse information in any bag-of-words model, to improve sentiment classification accuracy. We also probe the influence of the semantic operators like *modals* and *negations* on the discourse relations that affect the sentiment of a sentence. Discourse relations and corresponding rules are identified with minimal processing - just a list look up. We first give a linguistic description of the various discourse relations which leads to conditions in rules and features in SVM. We show that our discourse-based bag-of-words model performs well in a noisy medium (*Twitter*), where it performs better than an existing Twitter-based application. Furthermore, we show that our approach is beneficial to structured reviews as well, where we achieve a better accuracy than a state-of-the-art system in the *travel review* domain. Our system compares favorably with the state-of-the-art systems and has the additional attractiveness of being less resource intensive.

KEYWORDS : Sentiment Analysis, Discourse, Twitter, Connectives, Micro-blogs

Proceedings of COLING 2012: Technical Papers, pages 1847–1864,
COLING 2012, Mumbai, December 2012.

1 INTRODUCTION

An essential phenomenon in natural language processing is the use of discourse relations to establish a coherent relation, linking phrases and clauses in a text. The presence of linguistic constructs like *connectives, modals, conditionals* and *negation* can alter sentiment at the sentence level as well as the clausal or phrasal level. Consider the example, "*@user share 'em! i'm* **quite excited** *about Tintin, despite* **not really liking** *original comics. Probably because Joe Cornish had a hand in."* The overall sentiment of this example is *positive*, although there is equal number of positive and negative words. This is due to the connective *despite* which gives more weight to the previous discourse segment. Any bag-of-words model would be unable to classify this sentence without considering the discourse marker. Consider another example, "*Think i'll stay with the whole 'sci-fi'* **shit***. but this time...a* **classic** *movie."* The overall sentiment is again *positive* due to the connective *but*, which gives more weight to the following discourse segment. Thus it is of utmost importance to capture all these phenomena in a computational model.

Traditional works in *discourse analysis* use a discourse parser (Marcu 2000; Zirn *et al.*, 2011, Wellner *et al.*; 2007; Pitler *et al.*, 2009; Elwell *et al.*, 2008) or a dependency parser (Vincent *et al.*, 2006). Many of these works and some other works in discourse (Taboada *et al.*, 2008; Zhou et al., 2011) build on the Rhetorical Structure Theory (RTS) proposed by Mann *et al.* (1988) which tries to identify the relations between the nucleus and satellite in the sentence.

Most of these theories are well-founded for *structured text*, and *structured* discourse annotated corpora are available to train the models. However, using these methods for micro-blog discourse analysis pose some fundamental difficulties:

1. Micro-blogs, like *Twitter*, do not have any restriction on the form and content of the user posts. Users do not use formal language to communicate in the micro-blogs. As a result, there are abundant *spelling mistakes, abbreviations, slangs, discontinuities* and *grammatical errors*. This can be observed in the given examples from real-life *tweets*. The errors cause natural language processing tools like *parsers* and *taggers* to fail frequently (Dey *et al.*, 2009). As the tools are generally trained on structured text, they are unable to handle the noisy and unstructured text in this medium. Hence most of the discourse-based methods, based on RST or parsing of some form, will be unable to perform very well in micro-blog data.

2. The web-based applications require a fast response time. Using a heavy linguistic resource like *parsing* increases the processing time and slows down the application.

Most of the works in micro-blogs, like *Twitter*, (Alec *et al.*, 2009; Read *et al.*, 2005; Pak *et al.*, 2010; Gonzalez *et al.*, 2011) use a bag-of-words model with features like part-of-speech information, unigrams, bigrams *etc.* along with other domain-specific, specialized features like *emoticons, hashtags etc.* In *most* of these works, the *connectives, modals* and *conditionals* are simply ignored as stop words during feature vector creation. Hence, the discourse information that can be harnessed from these elements is completely discarded. In this work, we show how

the *connectives, modals, conditionals* and *negation* based discourse information can be incorporated in a bag-of-words model to give better sentiment classification accuracy.

The roadmap for the rest of the paper is as follows: Related work is presented in Section 2. Section 3 presents a comprehensive view of the different discourse relations. Section 4 studies the effect of these relations on sentiment analysis and identifies the critical ones. Section 5 discusses the influence of some semantic operators on discourse relations for sentiment analysis. We develop techniques for using the discourse relations to create feature vectors in Section 6. Lexicon based classification and supervised classification systems are presented in Section 7 to classify the feature vectors. Experimental results are presented in Section 8, where we use *three* different datasets, from the *Twitter* and *Travel review* domain, to validate our claim. The results are discussed in Section 9, followed by conclusions.

2 RELATED WORK

2.1 Discourse Based Works

Maru (2000) discussed probabilistic models for identifying elementary discourse units at clausal level and generating trees at the sentence level, using lexical and syntactic information from discourse-annotated corpus of RST. Wellner *et al.* (2007) considers the problem of automatically identifying arguments of discourse connectives in the PDTB. They model the problem as a predicate-argument identification where the predicates are discourse connectives and arguments serve as anchors for discourse segments. Wolf *et al.* (2005) presents a set of discourse structure relations and ways to code or represent them. The relations were based on Hobbs (1985). They report a method for annotating discourse coherent structures and found different kinds of crossed dependencies.

In the work, *Contextual Valence Shifters* (Polanyi *et al.,* 2004), the authors investigate the effect of *intensifiers, negatives, modals* and *connectors* that changes the prior polarity or valence of the words and brings out a new meaning or perspective. They also talk about pre-suppositional items and irony and present a simple weighting scheme to deal with them.

Somasundaran *et al*. (2009) and Asher *et al.* (2008) discuss some discourse-based supervised and unsupervised approaches to opinion analysis. Zhou *et al.* (2011) present an approach to identify discourse relations as identified by RST. Instead of depending on cue-phrase based methods to identify discourse relations, they leverage it to adopt an unsupervised approach that would generate semantic sequential representations (SSRs) without cue phrases.

Taboada *et al*. (2008) leverage discourse to identify relevant sentences in the text for sentiment analysis. However, they narrow their focus to adjectives alone in the relevant portions of the text while ignoring the remaining parts of speech of the text.

Most of these discourse based works make use of a *discourse parser* or a *dependency parser* to identify the scope of the discourse relations and the opinion frames. As said before, the parsers fare poorly in the presence of noisy text like *ungrammatical sentences* and *spelling mistakes* (Dey *et al.,* 2009). In addition, the use of parsing slows down any real-time interactive system due to

increased processing time. For this reason, the micro-blog applications mostly build on a bag-of-words model.

2.2 Twitter Based Works

Twitter is a micro-blogging website and ranks second amongst the present social media websites (Prelovac, 2010). A micro-blog allows users to exchange small elements of content such as short sentences, individual pages, or video links. Alec *et al.* (2009) provide one of the first studies on sentiment analysis on micro-blogging websites. Barbosa *et al.* (2010) and Bermingham *et al.* (2010) both cite noisy data as one of the biggest hurdles in analyzing text in such media.

Alec *et al.* (2009) describe a distant supervision-based approach for sentiment classification. They use *hashtags* in tweets to create training data and implement a multi-class classifier with topic-dependent clusters. "*The # symbol, called a hashtag, is used to mark keywords or topics in a Tweet. It was created organically by Twitter users as a way to categorize messages*"[1].

Barbosa *et al.* (2010) propose an approach to sentiment analysis in Twitter using POS-tagged n-gram features and some Twitter specific features like *hashtags.* Joshi *et al.* (2011) propose a rule-based system, *C-Feel-It*, which classifies a tweet as positive or negative based on the opinion words present in it. It uses sentiment lexicons for classification and twitter-specific features like *emoticons, slangs, hashtags etc.* Use of *emoticons* is common in social media and micro-blogging sites, where the users express their sentiment in the form of accepted symbols. Example: ☺ *(happy)*, ☹ *(sad).*

Read *et al.*, (2005) and Pak *et al.* (2010) propose a method to automatically create a training corpus using micro-blog specific features like *emoticons*, which is subsequently used to train a classifier. Gonzalez *et al.* (2011) discuss an approach to identify sarcasm in tweets. To create a corpus of *sarcastic, positive* and *negative* tweets, they rely on the user provided information in the form of *hashtags.* They claim that the author is the best judge for determining whether the tweet is sarcastic or not, which is indicated by the *hashtags* used by the author in the post.

Our work builds on the discourse-related works of Polanyi *et al.* (2004), Wolf *et al.* (2005) and Taboada *et al.* (2008) and carries the idea further in the sentiment analysis of micro-blogs. We exploit the various features discussed in the Twitter specific works to develop a bag-of-words model, in which the discourse features are incorporated to give better sentiment classification accuracy.

We evaluate our system on three datasets using lexicon-based classification as well as a supervised classifier (SVM). We use a manually labeled tweet set of 8,507 tweets and an automatically annotated set of 15,204 tweets using *hashtags*, to establish our claim. We, further, use a dataset from the *travel review* domain by Balamurali *et al.* (2011) to show that our method is beneficial to structured reviews as well, which is indicated by the improved classification accuracy over the compared work.

[1] https://support.twitter.com/articles/49309

3 CATEGORIZATION OF DISCOURSE RELATIONS

"An important component of language comprehension in most natural language contexts involves connecting clauses and phrases together in order to establish a coherent discourse" (Wolf *et al.,* 2004). A coherently structured discourse is a collection of sentences having some relation with each other. A coherent relation reflects how different discourse segments interact (Hobbs 1985; Marcu 2000; Webber *et al.,* 1999). Discourse segments are non-overlapping spans of text. The interaction relations between discourse segments may be of various forms as listed in *Table 1*.

Coherence Relations	Conjunctions
Cause-effect	because; and so
Violated Expectations	although; but; while
Condition	if…(then); as long as; while
Similarity	and; (and) similarly
Contrast	by contrast; but
Temporal Sequence	(and) then; first, second, … before; after; while
Attribution	according to …; …said; claim that …; maintain that …; stated
Example	for example; for instance
Elaboration	also; furthermore; in addition; note (furthermore) that; (for , in, with) which; who; (for, in, on, against, with) whom
Generalization	in general

Table 1: Contentful Conjunctions used to illustrate Coherence Relations (Wolf *et al.* 2005)

Our work, almost entirely, rests on this platform, where we identify the relations from *Table 1*, which can affect the analysis of opinions most in a discourse segment. *Table 2* provides some examples, taken from *Twitter*, to illustrate the effect of conjunctions in discourse analysis. These examples are similar to the ones in Polanyi *et al.* (2004) and Taboada *et al.* (2008). The words in *bold* connect the discourse segment in brackets. The relations are broadly classified in ten categories in *Table 2*.

4 DISCOURSE RELATIONS CRITICAL FOR SENTIMENT ANALYSIS

Not all of the discourse relations are significant from the point of view of sentiment analysis. This section examines the role of the critical ones in SA.

1. Violated Expectations and Contrast - A simple bag-of-words model will classify *Example 2* (*Table 2*) as neutral. This is because it has one positive term *excited* and one negative phrase *not*

really liking. However, it represents a positive emotion of the opinion holder, due to the segment after the connective *despite*. In *Example 5*, *brightened* is positive and *poorly* is negative. Hence the overall polarity is un-decided. But it should have been *positive*, since the segment following *but* gives the overall impression of the opinion-holder which is positive.

Violating expectation conjunctions oppose or refute the neighboring discourse segment. We further categorize them into the following *two* sub-categories: *Conj_Fol* and *Conj_Prev.*

Conj_Fol is the set of conjunctions that give more importance to the discourse segment that follows them. *Conj_Prev* is the set of conjunctions that give more importance to the previous discourse segment.

Thus, in *Example 5* of *Table 2*, the discourse segment following *but* should be given more weight. In *Example 2*, the discourse segment preceding *despite* should be given more weight.

1. Cause-effect: *(YES! I hope she goes with Chris)* **so** *(I can freak out like I did with Emmy Awards.)*

2. Violated Expectations: *(i'm quite excited about Tintin),* **despite** *(not really liking original comics.)*

3. Condition: **If** ***(****MicroMax improved its battery life), (it wud hv been a gr8 product).*

4. Similarity: *(I lyk Nokia)* **and** *(Samsung as well).*

5. Contrast: *(my daughter is off school very poorly),* **but** *(brightened up when we saw you on gmtv today).*

6. Temporal Sequence: *(The film got boring)* **after a while**.

7. Attribution: ***(****Parliament is a sausage-machine: the world)* **according to** *(Kenneth Clarke).*

8. Example: *(Dhoni made so many mistakes…)* **for instance**, *(he shud've let Ishant bowl wn he was peaking).*

9. Elaboration: **In addition** *(to the worthless direction), (the story lacked depth too).*

10. Generalization**: In general**,*(movies made under the RGV banner) (are not worth a penny).*

Table 2: Examples of Discourse Coherent Relations

2. Conclusive or Inferential Conjunctions - These are the set of conjunctions, *Conj_infer*, that tend to draw a conclusion or inference. Hence, the discourse segment following them (*subsequently* in *Example 11*) should be given more weight.

Example 11: @User *I was nt much satisfied with ur so-called gud phone and* **subsequently** *decided to reject it.*

3. Conditionals - In *Example 3 (Table 2),* both *improve* and *gr8* represent a *high degree* of positive sentiment. But the presence of *if* tones down the final polarity as it introduces a hypothetical situation in the context. The *if…then…else* constructs depict situations which may or may not happen subject to certain conditions.

In our work, the polarity of the discourse segment in a conditional statement is toned down, in *lexicon-based classification*. In *supervised classifiers*, the conditionals are marked as features. Such statements are not completely ignored as objective, as they bear some sentiment polarity.

4. Other Discourse Relations - Sentences under *Cause-Effect, Similarity, Temporal Sequence, Attribution, Example, Generalization and Elaboration,* provide no contrasting, conflicting or hypothetical information. They can be handled by taking a simple majority valence of the individual terms, as in a bag-of-words model.

Table 3 lists all the essential discourse relations discussed in this section. The relations have been compiled from Hobbs (1985), Polanyi *et al.* (2004) and Taboada *et al.* (2008).

5 SEMANTIC OPERATORS INFLUENCING DISCOURSE RELATIONS

There are some semantic operators that influence the discourse relations connecting the phrases. In the sentence *You may like the movie despite the bad reviews,* the connective *despite* gives more weightage to the discourse segment preceding it and hence, *like* is weighed up. But the uncertainty resulting from *may* that pulls down the weightage is completely ignored. Similarly, in the sentence *He put a lot of effort for the finals, but still it was not good enough to win the match,* the connective *but* upweights *good* and *win* ignoring the negation operator *not*. In this section, we consider the semantic operators like the *modals* and *negation*, ignoring which results in an incorrect interpretation of the sentiment.

Relations	Attributes
Conj_Fol	*but, however, nevertheless, otherwise, yet, still, nonetheless*
Conj_Prev	*till, until, despite, in spite, though, although*
Conj_Infer	*therefore, furthermore, consequently, thus, as a result, subsequently, eventually, hence*
Conditionals	*If*
Strong_Mod	*might, could, can, would, may*
Weak_Mod	*should, ought to, need not, shall, will, must*
Neg	*not, neither, never, no, nor*

Table 3: Discourse Relations and Semantic Operators Essential for Sentiment Analysis

1. Modals - Events that have happened, events that are happening or events that are certain to occur are called *realis events.* Events that have possibly occurred or have some probability to occur in the distant future are called *irrealis events.* Thus, it is important to distinguish between

real situations and *hypothetical* ones. The modals (*might, may, could, should, would etc.*) depict *irrealis events. Example 3 (Table 2)* does not necessarily talk of MicroMax being *great*, but talks of its *possibility* of being *great* subject to certain conditions (its *battery life*). These constructs cannot be handled by taking a simple majority valence of terms.

We further divide the modals into *two* sub-categories: *Strong_Mod* and *Weak_Mod*.

Strong_Mod is the set of modals that express a higher degree of uncertainty in any situation.

Weak_Mod is the set of modals that express lesser degree of uncertainty and more emphasis on certain events or situations.

In our work, the polarity of the discourse segment neighboring a *strong modal* is toned down in *lexicon-based classification*, similar to the *conditionals*, as it expresses a higher degree of uncertainty. In *supervised classifiers*, the modals are marked as features.

Example 12 (Strong Modals): *Unless I missed the announcement their God is now featured on postage stamps, it* **might** *be a hard sell.*

He **may** *be a rising star.*

Example 13 (Weak Modals): *G.E 12* **must** *be the* most *deadly General Election for politicians ever.*

Our civil service **should** *work without TD interference.*

2. Negation - The negation operator (Example: *not, neither, nor, nothing etc.*) inverts the sentiment of the word following it. The usual way of handling negation in SA is to consider a window of size *n* (typically 3-5) and reverse the polarity of all the words in the window. In *Example 14*, negating all the words in a window of size 5 reverses the polarity of "*like*" for Samsung as well; this is undesirable. We consider a negation window of size 5 and reverse all the words in the window, till either the window size exceeds or a *violating expectation* (*or a contrast*) conjunction is encountered. Hence, the scope of reversing polarity is limited to the appearance of *but* in the given example.

Example 14 (Negation): *I do not like Nokia but I like Samsung.*

6 ALGORITHM TO HARNESS DISCOURSE INFORMATION

The discourse relations and the semantic operators (identified in *Section 4* and *Section 5*) are used to create a feature vector, according to the following algorithm.

Let a user post R consist of 'm' sentences s_i $(i=1…m)$, where each s_i consist of n_i words w_{ij} $(i=1…m,\ j=1…n_i)$. Let f_{ij} be the weight of the word w_{ij} in sentence s_i, initialized to 1. Let A be the set of all discourse relations in *Table 3*. Let $flip_{ij}$ be a variable which indicates whether the polarity of w_{ij} should be flipped or not. Let hyp_{ij} be a variable which indicates the presence of a *conditional or a strong modal* in s_i.

for $i=1…m$
  for $j=1…n_i$
    f_{ij}=1;
    hyp_{ij}=0;
1. **if** $w_{ij} \in$ Conditionals *or* $w_{ij} \in$ Strong_Mod
      hyp_{ij}=1;
    end if
  end for

  for $j=1…n_i$
    $flip_{ij}$=1;
2. **if** $w_{ij} \in$Conj_Fol *or* $w_{ij} \in$ Conj_Infer
      for $k=j+1…n_i$ *and* $w_{ij} \notin A$
        f_{ik}+=1;
      end for
    end if
3. **else if** $w_{ij} \in$ Conj_Prev
      for $k=1…j-1$ && $w_{ij} \notin A$
        f_{ik}+=1;
      end for
    end if
4. **else if** $w_{ij} \in$ *Neg*
      for $k=1…$*Neg_Window and* $w_{ik} \notin$ *Conj_Prev and* $w_{ik} \notin$ *Conj_Fol*
        $flip_{i,j+k}$=-1;
      end for
    end if
  end for
end for
Input: Review *R*
Output: w_{ij}, f_{ij}, $flip_{ij}$, hyp_{ij}

Algorithm 1: Using the Discourse Relations to Create the Feature Vector

In *Step 1*, we mark all the *conditionals* and *strong modals* which are handled separately by the lexicon-based classifier and the supervised classifier.

In *Step 2* and *Step 3*, the weight of any word appearing before *Conj_Prev* and after *Conj_Fol* or *Conj_Infer* is incremented by 1.

In *Step 4*, the polarity of all the words appearing within a window (*Neg_Window* is taken as 5), from the occurrence of a negation operator and before the occurrence of a *violating expectation* conjunction, are reversed.

Finally, we get the feature vector $\{w_{ij}, f_{ij}, flip_{ij}$ and $hyp_{ij}\}$ for all the words in the review.

Here, the assumption is that the effect of any conjunction is restricted to continuous spans of text till another conjunction or the sentence boundary.

7 FEATURE VECTOR CLASSIFICATION

We devised a lexicon based system as well as a supervised system for feature vector classification.

7.1 Lexicon Based Classification

The Bing Liu sentiment lexicon (Hu *et al.*, 2004) is used to find the polarity $pol(w_{ij})$ of a word w_{ij}. It contains around 6800 words which are manually polarity labeled. The polarity of the review (*pos* or *neg*) is given by,

$$sign(\sum_{i=1}^{m}\sum_{j=1}^{n_i} f_{ij} \times flip_{ij} \times p(w_{ij}))$$

$$where\ p(w_{ij}) = pol(w_{ij})\ \ if\ hyp_{ij} = 0$$
$$= \frac{pol(w_{ij})}{2} \quad if\ hyp_{ij} = 1 \qquad \ldots\ Equation\ 1$$

Equation 1 finds the weighted, signed polarity of a review. The polarity of each word, $pol(w_{ij})$ *being +1 or -1*, is multiplied with its discourse-weight f_{ij} (assigned by *Algorithm 1*), and all the weighted polarities are added. $Flip_{ij}$ indicates if the polarity of w_{ij} is to be negated.

In case there is any *conditional* or *strong modal* in the sentence (indicated by $hyp_{ij} = 1$), then the polarity of every word in the sentence is toned down, by considering half of its assigned polarity ($\frac{+1}{2}$ *or* $\frac{-1}{2}$).

Thus, if *good* occurs in the user post twice, it will contribute a polarity of $+1 \times 2 = +2$ to the overall review polarity, *if* $hyp_{ij} = 0$. In the presence of a *strong modal* or *conditional*, it will contribute a polarity of $\frac{+1}{2} \times 2 = +1$.

All the *stop words*, *discourse connectives* and *modals* are ignored during the classification phase, as they have a zero polarity in the lexicon.

7.2 Supervised Classification

The Support Vector Machines have been found to outperform other classifiers, like *Naïve Bayes* and *Maximum Entropy*, in sentiment classification (Pang *et al.*, 2002). Hence, in our work, SVM's are used to classify the set of feature vectors $\{flip_{ij}, w_{ij}, f_{ij}$ and $hyp_{ij}\}$.

Features used in the Support Vector Machines:

N-grams – *Unigrams* along with *Bigrams* are used.

Stop Words – All the stop words (like *a, an, the, is etc.*) and discourse connectives are discarded.

Feature Weight – In the *baseline bag-of-words* model, the feature weight has been taken as the feature frequency *i.e.* the number of times the unigram or bigram appears in the text. In the *discourse-based bag-of-words* model, the *discourse-weighted frequency* of a word is considered. *Algorithm 1* assigns a weight f_{ij} to every occurrence of a word w_{ij} in the post. If the same word occurs multiple times, the weights from its multiple occurrences will be added and used as a feature weight for the word.

Modal and Conditional Indicator – This is a Boolean variable which indicates the presence of a strong modal or conditional in the sentence (i.e. $hyp_{ij}=1$).

Stemming – All the words are stemmed in the text so that "*acting*" and "*action*" have a single entry corresponding to "*act*".

Negation – A Boolean variable ($flip_{ij}$) is appended to each word (w_{ij}) to indicate whether it is to be negated or not (i.e. $flip_{ij}=1$ or $flip_{ij}=0$).

Emoticons – An emoticon dictionary is used to map each emoticon to a *positive* or *negative* class. Subsequently, the emoticon class information is used in place of the emoticon.

Part-of-Speech Information – The part-of-speech information is also used with a word.

Feature Space - In the *lexeme feature space* individual words are used as features; whereas in the *sense space*, the sense of the word (synset-id) is used in place of the word. A synset is a set of synonyms that collectively disambiguate each other to give a unique sense to the set, identifiable by the *synset-id*. This is beneficial in distinguishing between the various senses of a word.

For example, the word *bank* has 18 senses (10 Noun senses and 8 Verb Senses). Consider the two senses of a *bank* – 1) *Bank* in the sense of "*a financial institution*", identifiable by the synset "*depository financial institution, bank, banking concern, banking company"*, and 2) *Bank* in the sense of *relying*, identifiable by the synset "*trust, swear, rely, bank"*. Now, the first sense has an objective polarity whereas the second sense has a positive polarity. This distinction cannot be made in the lexeme feature space, where we consider only the *first* sense of the word.

8 EVALUATION

8.1 Dataset

We performed experiments on three different datasets to validate our approach:

Dataset 1: Twitter is crawled using a Twitter API and 8507 tweets are collected based on a total of around 2000 different entities from 20 different domains. The following domains are used for crawling data: *Movie, Restaurant, Television, Politics, Sports, Education, Philosophy, Travel, Books Technology, Banking & Finance, Business, Music, Environment, Computers, Automobiles,*

Cosmetics brands, Amusement parks and *Eatables and History.* These are manually annotated by 4 annotators into four classes - *positive, negative, objective-not-spam* and *objective-spam.* The objective-not-spam category contains tweets which are objective in nature but are not spams. The objective-spam category contains spam tweets which are subsequently ignored during evaluation.

Dataset 2: Following the works of Read *et al.* (2005), Alec *et al.* (2009), Pak *et al.* (2010) and Gonzalez *et al.* (2011) we create an artificial dataset using *hashtags*. The Twitter API is used to collect another set of 15,214 tweets based on *hashtags*. Hashtags *#positive, #joy, #excited, #happy etc.* are used to collect tweets bearing positive sentiment, whereas hashtags like *#negative, #sad, #depressed, #gloomy, #disappointed etc.* are used to collect negative tweets. *Hashtag keywords are subsequently removed from the tweets.*

Dataset 3: *Travel Review* Dataset in Balamurali *et al.* (2011) contains 595 polarity-tagged documents for each of the positive and negative classes. All the words in the travel review documents are automatically tagged with their corresponding synset-id's using *Iterative Word Sense Disambiguation* algorithm (Khapra *et al.*, 2010).

8.2 Evaluation on the Twitter Dataset 1 and 2

The crawled tweets are pre-processed before evaluation. All the *links* (urls) in the tweets are replaced by *#link*. All the *user id's* in the tweets are replaced by *#user*.

Manually Annotated Dataset 1				
#Positive	#Negative	#Objective Not Spam	#Objective Spam	Total
2548	1209	2757	1993	8507
Auto Annotated Dataset 2				
#Positive		#Negative		Total
7348		7866		15214

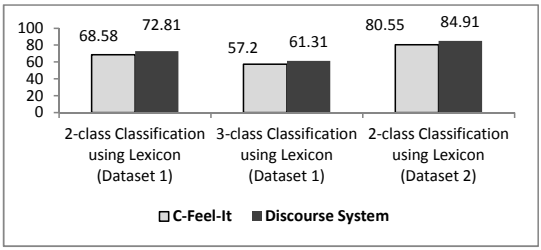
Table 4: Twitter Datasets 1 and 2 Statistics

Evaluations are performed in *Dataset 1* and *2* under a *2-class* and a *3-class* classification setting. In the 2-class setting, only *positive* and *negative* tweets are considered; whereas in the 3-class setting *positive, negative* and *objective-not-spam* tweets are considered. All the experiments in these two datasets are performed in the *lexeme feature space* using *lexicon-based classification* as well as *supervised classification*.

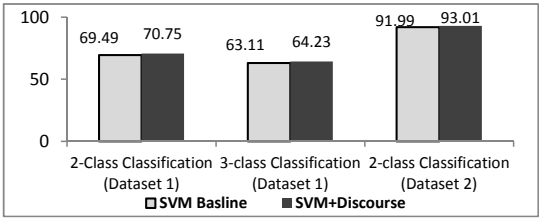
The baseline system, for this part of the evaluation, is taken as *C-Feel-It* (Joshi *et al.*, 2011). It is a rule-based system which implements a bag-of-words model using lexicon-based classification. The accuracy comparisons between C-Feel-It and the discourse system are performed under identical settings. The only difference between the two systems is the handling of *connectives, modals, conditionals* and *negation*, as indicated by *Algorithm 1*.

Graph 1 shows the accuracy comparison between C-Feel-It and the discourse system, in Datasets 1 and 2, using lexicon-based classification under a 2-class and a 3-class setting. *Graph 2* shows

the accuracy comparison between baseline SVM and SVM integrated with discourse features, in Datasets 1 and 2, under a 2-class and a 3-class setting. All the SVM features discussed in *Section 7.2*, except the discourse features arising out of the incorporation of *discourse weighting*, *modal and conditional indicator* and *negation*, are used in the baseline SVM model. A linear kernel, with default parameters, is used in the SVM (Chang *et al.*, 2011) with 10-fold cross-validation.



Graph 1: Accuracy Comparison between C-Feel-It and Discourse System using Lexicon in Datasets 1 and 2



Graph 2: Accuracy Comparison between Baseline SVM and SVM with Discourse in Datasets 1 and 2

8.3 Evaluation on the Travel Review Dataset 3

The *travel review* dataset (Balamurali *et al.*, 2011) is used to determine whether our discourse-based approach performs well for structured text as well. This evaluation is done under a 2-class classification setting in the *lexeme space* as well as the *sense space*. *Table 5* shows the accuracy comparison between the baseline bag-of-words model and bag-of-words model integrated with discourse features using lexicon-based classification, in dataset 3, under a 2-class setting.

Sentiment Evaluation Criterion	**Accuracy**
Baseline Bag-of-Words Model	69.62
Bag-of-Words Model + Discourse	**71.78**

Table 5: Accuracy Comparison between Bag-of-Words and Discourse System using Lexicon in **Dataset 3**

An automatic word sense disambiguation algorithm, *Iterative Word Sense Disambiguation* (Khapra *et al.*, 2010), has been used to auto-annotate the words in the review with their corresponding synset-id's. The same dataset is used in this work. A linear kernel, with default parameters, is used in the SVM with 5-fold cross-validation, similar to the compared system (Balamurali *et al.*, 2011). *Table 6* shows the performance of the discourse system along with the compared system using different features, on Dataset 3, using supervised classification. The features used in the SVM, for this part of the evaluation, include *stop word removal*, *no stemming*, *part-of-speech information* and *unigrams*. These features are used in all the systems in *Table 6*, including the discourse one. *Table 6* shows the system accuracy under four scenarios:

1. When only unigrams are used
2. When only sense of unigrams are used
3. When unigrams are used along with their senses (synset-id's)
4. When unigrams are used with senses and discourse features

9 DISCUSSIONS

Accuracy improvements over the baseline and the compared systems in all the datasets clearly signify the effectiveness of incorporating discourse information for sentiment classification. The bag-of-words model integrated *with discourse information* outperforms the bag-of-words model, *without this information*, under all the settings; although, the performance improvements vary in different settings. Statistical tests have been performed and all the accuracy improvements have been found to be statistically significant with *99%* level of confidence.

Systems	Accuracy (%)
Only Unigrams	84.90
Only IWSD Sense of Unigrams [26]	85.48
Unigrams + IWSD Sense of Unigrams [26]	86.08
Unigrams + IWSD Sense of Unigrams + Discourse Features	**88.13**

Table 6: Accuracy Comparisons in Travel Review Dataset 3

9.1 Accuracy Comparison between C-Feel-It and Discourse System

These comparisons are performed under a 2-class and a 3-class classification setting, using lexicon-based classification, in the lexeme space under identical settings - the only difference being the incorporation of discourse features. In *Dataset 1*, there is an accuracy improvement of around *4%* over C-Feel-It for both 2-class and 3-class classification. The discourse system accuracy at *72.81%* for 2-class classification is higher than that of the 3-class classification accuracy of *61.31%*. This shows that 3-class classification of tweets is much more difficult than 2-class classification.

9.2 Accuracy Comparison between Baseline SVM and Discourse System

These comparisons are performed under a 2-class and a 3-class classification setting, using supervised classification, in the lexeme space. A similar feature set, except the discourse features, is used for both the systems. In *Dataset 1*, there is an accuracy improvement of *1%* in both the 2-class and 3-class classification, which has been found to be statistically significant. In *Dataset 2*, there is an accuracy improvement of *2%* over baseline SVM for 2-class classification. It is observed that in the 2-class setting, the discourse system performs better in the lexicon-based classification with an accuracy of *72.81%* compared to the supervised classification accuracy of *70.75%*. This is contrary to the common scenario in text classification, where the supervised classification system always performs much better than the lexicon-based classification. This may be due to the very sparse feature space, owing to the length limit of tweets (140 characters).

9.3 Accuracy Comparison in Dataset 3

In the *Travel review* dataset, lexicon-based classification yielded an accuracy improvement of *2%* for the discourse model over simple bag-of-words model, in the *lexeme space*. In the SVM classification, in the *sense space,* under a 2-class setting, the discourse system achieved an accuracy of *88%* compared to *86%* accuracy of Balamurali *et al.* (2011). A similar feature set has been used in both the models, which indicates that the performance improvement is due to the incorporation of discourse features in SVM.

9.4 Drawbacks

The lexicon-based classification suffers from the usage of a generic lexicon in the *lexeme space*, where it cannot distinguish between the various senses of a word. The lexicons do not have entries for the interjections like *wow*, *duh etc.* which are strong indicators of sentiment. The frequent spelling mistakes, abbreviations and slangs used in the tweets do not have entry in the lexicons. For example, *love* and *great* are frequently written as *luv* and *gr8* respectively, which will not be detected. A spell-checker may help the system in this regard.

The supervised system suffers from a sparse feature space due to very short contexts. A concept expansion approach, to expand the feature vectors, may prove to be useful. This is due to the extensive world knowledge embedded in the tweets. For example, the tweet "*He is a Frankenstein*" is tagged as objective. The knowledge that *Frankenstein* is a negative concept is not present in the lexicon. The IWSD algorithm for automatic sense annotation has an F-Score of *70%*, which means many of the word-senses were wrongly tagged. In case a better WSD algorithm is used, higher system accuracy can be achieved in the travel review dataset.

In the absence of *parsing* and *tagging* information, due to the noisy nature of the tweets, the scope of the discourse marker has been heuristically taken till the sentence boundary or till the next discourse marker. Consider the sentence, "*I wanted to follow my dreams and ambitions despite all the obstacles, but I did not succeed.*" Here *want* and *ambition* will get the polarity *+2 each*, as they appear before *despite*; *obstacle* will get a polarity *-1* and *not succeed* will get a polarity *-2*. Thus the overall polarity is *+1*, whereas the overall sentiment should be *negative*.

This is because we do not consider the *positional importance* of a discourse marker in the sentence and consider all the discourse markers to be equally important. A better method is to give a ranking to the discourse markers based on their *positional* and *pragmatic* importance.

10 FUTURE WORKS AND CONCLUSIONS

In this work, we showed that the incorporation of discourse markers in a bag-of-words model improves the sentiment classification accuracy by *2 - 4%*. This approach is particularly beneficial for - 1) applications dealing with noisy text where *parsing* and *tagging* do not perform very well, and 2) applications, requiring a fast response time, where employing a heavy linguistic tool like *parsing* will be detrimental to its performance due to the increased processing time.

Most of the works in micro-blogs, like *Twitter*, build on a bag-of-words model that ignores the discourse markers. We demonstrated an approach to incorporate discourse information to improve their performance, retaining the simplicity of the bag-of-words model. We validated this claim on two different datasets (manually and automatically annotated) from *Twitter*, where we achieved an accuracy improvement of *4%* for lexicon-based classification over an existing application (Joshi *et al.*, 2011), and 2% for supervised classification over the baseline SVM with advanced features. We also showed that our method fares well for structured reviews as well, where we achieved similar accuracy improvements over Balamurali *et al.*, 2011.

References

Alec, G.; Lei, H.; and Richa, B. Twitter sentiment classification using distant supervision. Technical report, Standford University. 2009

AR, Balamurali and Joshi, Aditya and Bhattacharyya, Pushpak. Harnessing WordNet Senses for Supervised Sentiment Classification. In Proceedings of Empirical Methods in Natural Language Processing (EMNLP). 2011

Asher, Nicholas and Benamara, Farah and Mathieu, Yvette Yannick. Distilling opinion in discourse: A preliminary study. In Proceedings of Computational Linguistics (CoLing). 2008

Barbosa, L., and Feng, J. Robust sentiment detection on twitter from biased and noisy data. In 23rd International Conference on Computational Linguistics: Posters, 36–44. 2010

Bermingham, A., and Smeaton, A. Classifying sentiment in microblogs: is brevity an advantage ACM 1833–1836. 2010

Chih-Chung Chang and Chih-Jen Lin, LIBSVM : a library for support vector machines. ACM Transactions on Intelligent Systems and Technology, 2:27:1--27:27. 2011

Dey, Lipika and Haque, Sk. Opinion Mining from Noisy Text Data. International Journal on Document Analysis and Recognition 12(3). pp 205-226. 2009

Elwell, Robert and Baldridge, Jason. Discourse Connective Argument Identification with Connective Specific Rankers. In Proceedings of IEEE International Conference on Semantic Computing. 2008

Gonzalez-Ibanez, Roberto and Muresan, Smaranda and Wacholder, Nina. Identifying sarcasm in Twitter: a closer look, In Proceedings of ACL: short paper. 2011

Hobbs, Jerry R., and Michael Agar.The Coherence of Incoherent Discourse, Journal of Language and Social Psychology, vol. 4, nos. 3-4, pp. 213-232. 1985

Joshi, A.; Balamurali, A. R.; Bhattacharyya, P.; and Mohanty, R. C-feel-it: a sentiment analyzer for microblogs. In Proceedings of ACL: Systems Demonstrations, HLT '11, 127–132. 2011

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

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

Minqing Hu and Bing Liu. Mining and summarizing customer reviews. In Proceedings of ACM SIGKDD. 2004

Mitesh Khapra, Sapan Shah, Piyush Kedia, and Pushpak Bhattacharyya. Domain-specific word sense disambiguation combining corpus basedand wordnet based parameters. In Proceedings of GWC'10, Mumbai, India. 2010

Ng, Vincent and Dasgupta, Sajib and Arifin, S. M. Niaz. Examining the role of linguistic knowledge sources in the automatic identification and classification of reviews. In Proceedings of the COLING/ACL on Main conference poster sessions. 2006

Pak, Alexander and Paroubek, Patrick. Twitter as a Corpus for Sentiment Analysis and Opinion Mining. In Proceedings of LREC. 2010

Pang, Bo and Lee, Lillian and Vaithyanathan, Shivakumar. Thumbs up?: sentiment classification using machine learning techniques. In Proceedings of the ACL-02 conference on Empirical Methods in Natural Language. 2002

Pitler, Emily and Louis, Annie and Nenkova, Ani. Automatic Sense Prediction for Implicit Discourse Relations in Text. In Proceedings of ACL and IJCNLP. 2009

Polanyi, Livia and Zaenen, Annie. Contextual valence shifters. In Exploring Attitude and Affect in Text: Theories and Applications. AAAI Spring Symposium Series. 2004

Prelovac, V. 2010. Top social media sites. Web.

Read, Jonathon. Using emoticons to reduce dependency in machine learning techniques for sentiment classification. In Proceedings of Association for Computer Linguistics (ACL). 2005

Somasundaran, Swapna. Discourse-level relations for Opinion Analysis. PhD Thesis, University of Pittsburgh. 2010

Taboada, Maite and Brooke, Julian and Tofiloski, Milan and Voll, Kimberly and Stede, Manfred. Lexicon-based methods for sentiment analysis. Computational Linguistics. 2011

Taboada, Maite and Brooke, Julian and Voll, Kimberly. Extracting Sentiment as a Function of

Discourse Structure and Topicality. Simon Fraser Univeristy School of Computing Science Technical Report. 2008

Webber, Bonnie and Knott, Alistair and Stone, Matthew and Joshi, Aravind. Discourse relations: A structural and presuppositional account using lexicalized tag. In Proceedings of ACL. 1999

Wellner, Ben and Pustejovsky, James. Automatically identifying the arguments of discourse connectives. In Proceedings of The Joint Conference on EMNLP-CoNLL, pp. 92-101. 2007

Wolf, Florian and Gibson, Edward and Desmet, Timothy. Discourse coherence and pronoun resolution, Language and Cognitive Processes, 19(6), pp. 665–675. 2004

Wolf, Florian and Gibson, Edward. Representing discourse coherence: A corpus-based study, Computational Linguistics, 31(2), pp. 249–287. 2005

Zhou, Lanjun and Li, Binyang and Gao, Wei and Wei, Zhongyu and Wong, Kam-Fai. Unsupervised discovery of discourse relations for eliminating intra-sentence polarity ambiguities. In Proceedings of EMNLP. 2011

Zirn, C\"{a}cilia and Niepert, Mathias and Stuckenschmidt, Heiner and Strube, Michael. Fine-Grained Sentiment Analysis with Structural Features. In Proceedings of IJCNLP. 2011

YouCat : Weakly Supervised Youtube Video Categorization System from Meta Data & User Comments using WordNet & Wikipedia

Subhabrata Mukherjee[†], *Pushpak Bhattacharyya*[‡]
[†]IBM India Research Lab
[‡]Dept. of Computer Science and Engineering, IIT Bombay
subhabmu@in.ibm.com, pb@cse.iitb.ac.in

ABSTRACT

In this paper, we propose a *weakly supervised* system, *YouCat*, for categorizing Youtube videos into different genres like *Comedy, Horror, Romance, Sports* and *Technology* The system takes a *Youtube video url* as input and gives it a belongingness score for each genre. The key aspects of this work can be summarized as: (1) Unlike other genre identification works, which are *mostly* supervised, this system is *mostly unsupervised,* requiring *no labeled data for training.* (2) The system can easily incorporate new genres without requiring labeled data for the genres. (3) YouCat extracts information from the *video title*, *meta description* and *user comments* (which together form the *video descriptor*). (4) It uses *Wikipedia* and *WordNet* for concept expansion. (5) The proposed algorithm with a time complexity of $O(|W|)$ (where (|W|) is the number of words in the video descriptor) is efficient to be deployed in web for real-time video categorization. Experimentations have been performed on real world Youtube videos where YouCat achieves an F-score of *80.9%*, without using any labeled training set, compared to the supervised, multiclass SVM F-score of *84.36%* for *single genre prediction*. YouCat performs better for multi-genre prediction with an F-Score of *90.48%*. Weak supervision in the system arises out of the usage of manually constructed WordNet and genre description by *a few root words.*

KEYWORDS : Youtube, Genre Prediction, Comments, Metadata, Wikipedia, WordNet

Proceedings of COLING 2012: Technical Papers, pages 1865–1882,
COLING 2012, Mumbai, December 2012.

1 INTRODUCTION

In recent times there has been an explosion in the number of online videos. With the gradually increasing multimedia content, the task of efficient query-based video retrieval has become important. The proper genre or category identification of the video is essential for this purpose. The automatic genre identification of videos has been traditionally posed as a supervised classification task of the features derived from the audio, visual content and textual features. Whereas some works focus on classifying the video based on the *meta data (text)* provided by the uploader (Cui *et al.*, 2010; Borth *et al.*, 2009, Filippova *et al.*, 2011), other works attempt to extract low-level features by analyzing the *frames, signals, audio etc.* along with textual features (Ekenel *et al.*, 2010; Yang *et al.*, 2007). There have been some recent advances in incorporating new features for classification like the social content comprising of the *user connectivity* (Zhang *et al.*, 2011; Yew *et al.*, 2011), *comments* (Filippova *et al.*, 2011), *interest etc.*

All the above approaches pose the genre prediction task as supervised classification requiring a large amount of training data. It has been argued that a *serious challenge* for supervised classification is the availability and requirement of manually labeled data (Filippova *et. al*, 2010; Wu *et. al*, 2010; Zanetti *et. al*, 2008). For example, consider a video with the descriptor "*It's the NBA's All-Mask Team!*". Unless there is a video in the training set with *NBA* in the video descriptor labeled with *Sport*, there is no way of associating *NBA* to *Sport*. It is also not possible to associate *NBA* to *Basketball* and then to *Sport*. As *new genre-related concepts* (like new sports, technologies, domain-dependent terms *etc.*) appear every day the training set should expand incorporating all these new concepts, which makes training very expensive. As the number of categories or genres is increased the data requirement goes up compounded. The problem is enhanced by the *noisy* and *ambiguous* text prevalent in the media due to the *slangs, acronyms etc.* The *very short text* provided by the user, for *title* and *video description*, provide little context for classification (Wu *et al.*, 2012). The focus of this paper is to propose a system that requires no labeled data for training and can be easily extended to identify new categories. The system can easily adapt to changing times, incorporating world knowledge, to overcome the labeled data shortage. It extracts all the features from the video uploader provided meta-data like the *video title, description of the video* as well as the *user comments.* The system incorporates social content by analyzing the user comments on the video, which is essential as the meta-data associated with a video is often absent or not adequate enough to predict its category. *WordNet* and *Wikipedia* are used as world knowledge sources for *expanding* the video descriptor since the uploader provided text is frequently very short, as are the user comments. WordNet is used for knowing the meaning of an unknown word whereas Wikipedia is used for recognizing the *named entities* (which are mostly absent in the WordNet) like "*NBA"* in the given example. In this work, we show how the textual features can be analyzed with the help of WordNet and Wikipedia to predict the video category without requiring any labeled training set.

The only weak supervision in the system arises out of the usage of a root words list (~ 1-3 words) used to describe the genre, WordNet which is manually annotated and a simple setting of the parameters of the model.

The paper is organized as follows: Section 2 gives the related work and compares them with our approach. Section 3 discusses the unsupervised feature extraction from various sources. Section 4 gives the algorithm for feature vector classification and genre identification. Section 5 discusses the parameter settings for the model. The experimental evaluations are presented in Section 6 followed by discussions of the results in Section 7. Section 8 concludes the paper with future works and conclusions.

2 RELATED WORK

The video categorization works can be broadly divided under 3 umbrellas: 1. Works that deal with low level features by extracting features from the video frames like the audio, video signals, colors, textures *etc.* 2. Works that deal with textual features like the title, tag, video description, user comments *etc.* 3. Works that combine low-level features like the video frame information with the high-level text features. In this section, we discuss only those works that include text as one of the features. Our work is similar to text classification but for a different application.

Filippova *et al.* (2011) showed that a text-based classifier, trained on imperfect predictions of weakly supervised video content-based classifiers, outperforms each of them taken independently. They use features from the video title, description, user comments, uploader assigned tags and use a maximum entropy model for classification.

Wang *et al.* (2010) considers features from the text as well as low-level video features, and proposes a fusion framework in which these data sources are combined with the small manually-labeled feature set independently. They use a Conditional Random Field (CRF) based fusion strategy and a Tree-DRF for classification.

The content features are extracted from training data in Cui *et al.* (2010) to enrich the text based semantic kernels to yield content-enriched semantic kernel which is used in the SVM classifier.

Borth *et al.* (2009) combines the results of different modalities like the uploader generated tags and visual features which are combined using a weighted sum fusion, where SVM's are used with bag of words as features. These categories are refined further by deep-level clustering using probabilistic latent semantic analysis.

Query expansion is performed in Wu *et al.* (2012) by using contextual information from the web like the related videos and user videos, in addition to the textual features and use SVM in the final phase for classification.

Some works have used user information like the browsing history along with other textual features. Zhang *et al.* (2006) develop a video categorization framework that combined multiple classifiers based on normal text features as well as users' querying history and clicking logs. They used Naïve Bayes with a mixture of multinomials, Maximum Entropy, and Support Vector Machines for video categorization.

Most of the works are similar to Huang *et al.* (2010) which use different text features and classifiers like the Naïve Bayes, Decision Trees and SVM's for classification.

Yang *et al*. (2007) propose a semantic modality that includes concept histogram, visual word vector model and visual word Latent Semantic Analysis (LSA); and a text modality that includes titles, descriptions and tags of web videos. They use various classifiers such as Support Vector Machine(SVM), Gaussian Mixture Model(GMM) and Manifold Ranking (MR) for classification.

Song *et al*. (2009) developed an effective semantic feature space to represent web videos consisting of concepts with small semantic gap and high distinguishing ability where Wikipedia is used to diffuse the concept correlations in this space. They use SVM's with fixed number of support vectors (n-ISVM) for classification.

All the above works build on supervised classification systems, requiring labeled data for training, mostly using the Support Vector Machines. In this paper, we propose a system that requires no labeled data for training, which is the primary difference of our work with those surveyed. Also, the usefulness of Wikipedia and WordNet for concept expansion has not been probed much in earlier video categorization tasks, save a few. We use many of the ideas from the above works and integrate them into YouCat.

3 FEATURE CONSTRUCTION

Given a *Youtube video url*, the objective is to assign scores to it which represent its belongingness to the different genres. The video genres are categories like *romance*, *comedy*, *horror*, *sports* and *technology*. The genre names are pre-defined in the system along with a small set of *root words* for each genre. The root words act like a description of the genre. For example, *funny* and *laugh* act as the key characteristics of the *comedy* genre. This allows *new* genres to be easily defined in the system in terms of the root words as well as to have a fine distinction between the genres.

A *seed list* of words is *automatically* created for each genre by searching a *thesaurus* using the roots words for that genre. A *concept list* is created for each genre with *relevant words* from the *WordNet* and *named entities* in *Wikipedia*, with the help of the seed list of the corresponding genre. Given a video descriptor consisting of the *video title*, the *meta-description of the video* and the *user comments*, the seed list and the concept list for each genre are used for finding appropriate matches in the *video descriptor* to predict appropriate tags or categories for the video using the scores.

3.1 Data Pre-Processing

3.1.1 Seed List Creation using Root Words

A set of tags is pre-defined in the system along with a set of 1-3 *root words* for each tag. A *seed list* of words is created for each genre (*defined in the system*) which captures the key characteristics of that category. For Example, *"love", "hug", "cuddle" etc.* are the characteristics of the *Romance* genre. *Root words of the genre* are taken and all their synonyms are retrieved from a thesaurus. *The root words list and the genre names are pre-defined in the system. Table 1* shows the root-words list for the *five* genres used in this work. An automatic breadth-first search is done on the thesaurus based on the root words to retrieve only the most relevant synonyms or

associated concepts. For example, the word *Laugh* is taken for its genre *Comedy* and all its first level synonyms are retrieved which are again recursively used to retrieve their level-one synonyms till a certain depth. A thesaurus is used for this purpose which gives every day words and slangs. In our work, the following thesaurus[1] retrieves the words *rofl, roflmao, lol etc*. when the word *Laugh* is looked up from the *Comedy* genre. A *snapshot* of the seed lists with number of words in the lists is shown in *Table 2*.

The set of root words can help in *fine genre distinction* as the seed list will have only *associated* concepts. For example if the *Transport* genre is sub-categorized into *Road* and *Railways*, the corresponding root words {car, road, highway, auto} and {train, rail, overhead wire, electricity, station} will distinguish between the two.

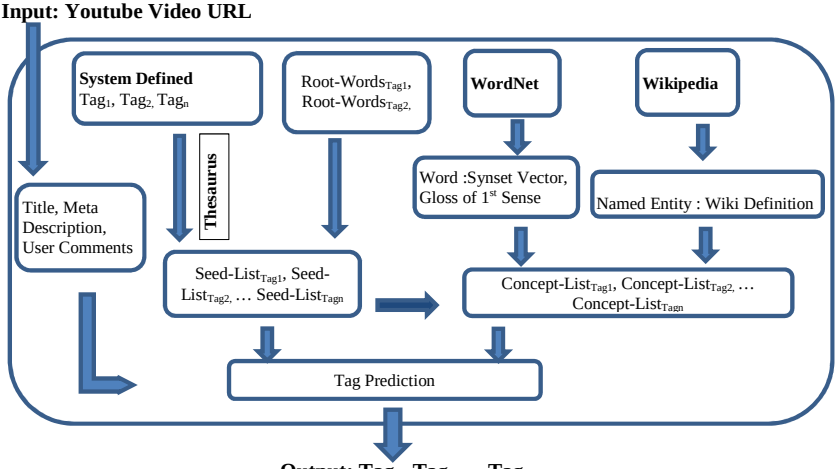


Fig. 1. System Block Diagram

Comedy	comedy, funny, laugh
Horror	horror, fear, scary
Romance	romance, romantic
Sport	sport, sports
Technology	tech, technology, science

Table 1. Root Words for Each Genre

[1] www.urbandictionary.com/thesaurus.php

Comedy (25)	funny, humor, hilarious, joke, comedy, roflmao, laugh, lol, rofl, roflmao, joke, giggle, haha, prank
Horror (37)	horror, curse, ghost, scary, zombie, terror, fear, shock, evil, devil, creepy, monster, hell, blood, dead, demon
Romance (21)	love, romantic, dating, kiss, relationships, heart, hug, sex, cuddle, snug, smooch, crush, making out
Sports (35)	football, game, soccer, basketball, cheerleading, sports, baseball, FIFA, swimming, chess, cricket, shot
Tech (42)	internet, computers, apple, iPhone, phone, pc, laptop, mac, iPad, online, google, mac, laptop, XBOX, Yahoo

Table 2. Snapshot of Seed List for Each Genre

3.1.2 Concept Hashing

Each word (used as a *key* for hashing) in the WordNet, that is not present in any seed list, is hashed with the set of all its synsets and the gloss of its first sense.

A synset is a set of synonyms that collectively disambiguate each other and give a unique sense to the set. For example, the word *dunk* has the synsets - {*dunk, dunk shot, stuff shot*; *dunk, dip, souse, plunge, douse*; *dunk*; *dunk, dip*}. Here the first synset {*dunk, dunk shot, stuff shot*} has the sense of a basketball shot. The meaning of a synset is clearer with its gloss. A gloss[2] is the definition or example sentences for a synset which portrays the context in which the synset or sense of the word can be used. For example, the gloss of the synset {*dunk, dunk shot, stuff shot*} is {*a basketball shot in which the basketball is propelled downward into the basket*}.

Technically, we should have taken only the words in the synset of its most appropriate sense. But we do not perform word sense disambiguation[3] to find out the proper synset of the word. Taking only the first sense provides fewer contexts while classifying the feature vector, and so the information from all the senses of a given word is used. The gloss of the first sense is frequently used, as in many cases the first sense is the best sense of a word (Macdonald *et al.*, 2007).

Wikipedia is necessary for *named entity recognition*, since the WordNet does not contain most of these entries. All the *named entities* in Wikipedia with the *top 2 line definition* in their corresponding Wiki articles are stored in a hashtable. For example, *NBA* is stored in the hashtable with its definition from the Wikipedia article as {*The National Basketball Association (NBA) is the pre-eminent men's professional basketball league in North America. It consists of thirty franchised member clubs, of which twenty-nine are located in the United States and one in Canada.*}.

Most of the named entities in practice are not unigrams like *Michael Jordon*. If the unigrams in this named entity are expanded separately, a different sense for each would be retrieved. This is not desirable. In this work, we use a simple heuristics method based on *capitalization* of the

[2] http://en.wikipedia.org/wiki/WordNet
[3] http://en.wikipedia.org/wiki/Word-sense_disambiguation

letters to identify the named entities. Any sequence of consecutive words such that each of them starts with a capital letter, and the sequence does not start or end with any *Stop Word* is considered a named entity. Stop Words are allowed within this sequence, provided the number of such Stop Words between any two consecutive words is less than or equal to two. Thus named entities like *United States of America, Lord of the Rings, Bay of Bengal etc.* are recognized. This method captures a lot of false positives. One such example can be the usage of capitalization in the social media in the form of pragmatics to express the intensity of emotions (Example: *I just LOVED that movie*). However, false positives are not a concern in our case as such entries, *if valid*, will only add to the concept lists. The named entity is considered as a single token and treated just like the unigrams.

3.2 Concept List Creation

Let *w* be any given word and its expanded form given by WordNet (*set of all its synsets* and the *gloss of its first sense*) or Wikipedia (*top 2 line definition*) be denoted by w'. Let w'_j be the j[th] word in the expanded word vector. Let $seed_k$ and $root_k$ be the seed list and root words list, respectively, corresponding to the k^{th} genre. The genre of *w* is given by

$$genre(w) = argmax_k \sum_j \mathbf{1}_{w'_j \in seed_k, w'_j \in root_k}$$

$$\ldots Equation\ 1$$

Here, **1** is an indicator function which returns 1 if a particular word is present in the seed list or root words list corresponding to a specific genre and 0 otherwise. In the given example, with the pre-defined 5 genres (*Table 1*), *dunk* and *basketball* both will be classified to the *Sports* genre as they have the maximum matches (*"shot", "basketball"*) from the seed list corresponding to the *Sports* genre in their expanded concept vector.

Finally, a *concept list* is created for each genre containing *associated* words in the WordNet (ignoring those in the seed lists) and named entities in the Wikipedia.

3.3 Video Descriptor Extraction

Given a video url, the *video title*, the *meta description* of the video and all the *user comments* on the video from Youtube are retrieved. A *stopwords* list is used to remove words like *is, are, been etc.* A lemmatizer is used to reduce each word to its base form or lemma. Thus "*play*", "*played*", "*plays*", "*playing*" are reduced to its lemma "*play*".

Consider the sentence in a video descriptor, "*It was an awesome slam dunk in the NBA finals by Michael Jordan*". None of the words here is present in any seed list. But *dunk* and *NBA* are present in the concept list corresponding to *Sports* genre and thus the given sentence is associated to *Sports*. The association (*Sports* via *Basketball*) can also be captured by considering the named entity *Michael Jordon* in Wikipedia.

4 FEATURE VECTOR CLASSIFICATION

Let the video descriptor *f* consist of *n* words, in which the j^{th} word is denoted by $word_j$. The *root word list*, *seed list* and the *concept list* for the k^{th} genre are denoted by $root_k$, $seed_k$ and $concept_k$ respectively. The score of f belonging to a particular $genre_k$ is given by,

$$score(f \in genre_k; w_1, w_2, w_3) = w_1 \times \sum_j \mathbf{1}_{word_j \in root_k} + w_2 \times \sum_j \mathbf{1}_{word_j \in seed_k} + w_3 \times \sum_j \mathbf{1}_{word_j \in concept_k}$$
$$where \quad w_3 < w_2 < w_1 \qquad \ldots Equation\ 2$$

Here, $\mathbf{1}$ is an indicator function that returns 1 if a word is present in the root words list, seed list or concept list corresponding to $genre_k$ and 0 otherwise. Weights w_1, w_2 *and* w_3 are assigned to words present in the root words list, seed list and the concept list respectively. The weight assigned to any root word is maximum as it has been specified as a part of the genre description manually. Lesser weightage is given to the words in the seed list as they are automatically extracted using a thesaurus. The weight assigned to concept list is the least to reduce the effect of *topic drift* during concept expansion (Manning *et al.*, 2008). The topic drift occurs due to the enlarged context window, during concept expansion, which may result in a match from the seed list of some other genre than the one it actually belongs to.

The score of a video belonging to a particular genre is,

$$score(video \in genre_k; p_1, p_2, p_3) = p_1 \times score(f^{Title} \in genre_k) + p_2 \times score(f^{Meta\ Data} \in genre_k) + p_3 \times score(f^{Comments} \in genre_k)$$
$$\ldots Equation\ 3$$

Here p_1, p_2, p_3 denote the weight of the feature belonging to the *title*, *meta data (meta description of the video)* and *user comments* respectively where $p_1 > p_2 > p_3$. This is to assign more importance to the title, then to the meta data and finally to the user comments. The genre to which the video belongs is given by,

$$video_{genre} = argmax_k\ score(video \in genre_k)$$
$$\ldots Equation\ 4$$

This assigns the highest scoring genre as the desired category for the video. However, most of the popular videos in Youtube can be attributed to more than one *genre*. Thus to allow multiple tags to be assigned to a video, a thesholding is done and the prediction is modified as:

$$video_{genre} = k, if\ score(video \in genre_k) \geq \theta$$
$$where\ \theta = \frac{1}{k} \sum_k score(video \in genre_k)$$
$$\ldots Equation\ 5$$

If the score of the video for any genre is greater than the average score of all the genres, then it is assigned as a possible tag for the video. In case the genre scores for the 5 categories are something like {*400, 200, 100, 50, 10*} with *avg=152*, then the first 2 genres are chosen. If any of the genre score is very high compared to the others, the average will rise decreasing the chance of other genres being chosen. *Algorithm 1* describes the genre identification steps in short.

Pre-processing:

1. *Define Genres* and *Root Words List for each genre*
2. *Create a Seed list for each genre by breadth-first-search in a Thesaurus, using root words in the genre or the genre name*
3. *Create a Concept List for each genre using all the words in WordNet (not present in Seed Lists) and Named Entities in Wikipedia using Equation 1*

Input: Youtube Video Url

1. *Extract Title, Meta Description of the video and User Comments from Youtube to form the video descriptor*
2. *Lemmatize all the words in the descriptor removing stop word.*
3. *Use Equations 2-4 for genre identification of the given video*

Output: Genre Tags

Algorithm 1. Genre Identification of a Youtube Video

5 PARAMETER SETTING

The upweighting of document zones by giving more weightage to some portions of the text than others is common in automatic text summarization and information retrieval (Manning *et al.*, 2008). A common strategy is to use extra weight for words appearing in certain portions of the text like the title and use them as separate features, even if they are present in some other portion of the text (Giuliano *et al.*, 2011). As a rule-of-thumb the weights can be set as integral multiples, preferably prime, to reduce the possibility of ties (Manning *et al.*, 2008).

We follow this line of thought in our work and upweight certain portions of the text like the *title*, *meta data*, *user comments* separately. We also assign different weight to words belonging to different lists according to importance.

There are 6 parameters for the model we used: $w_1,\ w_2,\ w_3,\ p_1,\ p_2,\ p_3$. The parameters can be best trained if some label information is available. However, in the absence of any label information, we adopt a simple approach to parameter setting as mentioned above. We took the first set of integers, satisfying all the constraints in *Equations 2 and 3*, and assigned them to the 6 parameters: $w_1 = 3,\ w_2 = 2, w_3 = 1,\ p_1 = 3,\ p_2 = 2,\ p_3 = 1$.

Semi-Supervised Learning of Parameters

This work *does not* evaluate this dimension for parameter learning, since our objective has been to develop a system that requires no labeling information. However, if some category information is available, a robust learning of parameters is possible.

Equation 1 and 2 can be re-written as:

$$score\left(f_k^{position} \in genre_k; w_1, w_2, w_3\right) = w_1 \times X_{1,k}^{position} + w_2 \times X_{2,k}^{position} + w_3 \times X_{3,k}^{position}$$

$$score(video_k \in genre_k; p_1, p_2, p_3) = Y_k = \sum_{position} p_{position} \sum_j w_j \times X_{j,k}^{position}$$

$= \sum_i \sum_j w'_{ij} X_j^i \quad (where\ w'_{ij} = p_i \times w_j)$

$$Or, Y_k = \boldsymbol{W}.\boldsymbol{X}_k\ (where\ \boldsymbol{W} = \left[w'_{1,1}\ w'_{1,2} \ldots w'_{3,3}\right]^T_{9\times 1}, \qquad \boldsymbol{X}_k = [X^1_{1,k}\ X^1_{2,k} \ldots\ X^3_{3,k}]_{1\times 9}$$

$$Or, \boldsymbol{Y} = \boldsymbol{W^T}.\boldsymbol{X}$$

This is a linear regression problem which can be solved by the ordinary least squares[4] method by minimizing the sum of the squared residuals i.e. the sum of the squares of the difference between the observed and the predicted values (Bishop *et al.*, 2006). The solution for $\boldsymbol{W}$ is given by $\boldsymbol{W} = (\boldsymbol{X^T X})^{-1}\boldsymbol{X^T Y}$

A regularizer can be added to protect against over-fitting and the solution can be modified as: $\boldsymbol{W} = (\boldsymbol{X^T X} + \delta \boldsymbol{I})^{-1}\boldsymbol{X^T Y} \quad where\ \delta\ is\ a\ parameter\ and\ \boldsymbol{I}\ is\ the\ identity\ matrix.$

6 EVALUATION

6.1 Data Collection

The following 5 genres are used for evaluation: *Comedy*, *Horror*, *Sports*, *Romance and Technology*. 12,837 videos are crawled from the Youtube following a similar approach like (Cu *et al.*, 2010; Wu *et al.*, 2012; Song *et al.*, 2009). Youtube has 15 pre-defined categories like *Romance*, *Music*, *Sports*, *People*, *Comedy etc.* These videos are automatically categorized in Youtube based on the user-provided tags while uploading the video and the video description. We crawl the videos directly from those categories using the Youtube API. *Table 3* shows the number of videos from each genre.

Comedy	Horror	Sports	Romance	Tech	Total
2682	2802	2577	2477	2299	12837

Table 3: Number of Videos in Each Genre

Only the 1st page of user comments is taken with comment length less than 150 characters. Short length comments are chosen as they are typically to the point, whereas long length comments often stray off the topic. The user comments are normalized by removing all the punctuations and reducing words like "*loveeee*" to "*love*". The number of user comments varied from 0 to 800 for different videos. *Table 4* shows the average number of user comments for the videos in each genre.

Comedy	Horror	Sports	Romance	Tech
226	186	118	233	245

Table 4: Average User Comments for Each Genre

The first integer values satisfying the constraints in the equations are taken as parameter values, which are set as: $w_1 = 3,\ w_2 = 2, w_3 = 1,\ p_1 = 3,\ p_2 = 2,\ p_3 = 1.$

[4] http://en.wikipedia.org/wiki/Ordinary_least_squares

6.2 Baseline System

All the words in the video descriptor consisting of the title, meta-description of the video and the user comments are taken as features for the SVM. A Multi-Class Support Vector Machines Classifier[5] with various features, like combination of unigrams and bigrams, incorporating part-of-speech (POS) information, removing stop words, using lemmatization *etc.*, is taken as the baseline. *Table 5* shows the baseline system accuracy with various features. A *linear kernel* is used with *10-fold* cross validation. SVM with lemmatized unigrams and bigrams as features, ignoring stop words, gave the maximum accuracy of *84.36%*.

SVM Features	**F_1-Score(%)**
All Unigrams	82.5116
Unigrams+Without stop words	83.5131
Unigrams+ Without stop words +Lemmatization	83.8131
Unigrams+Without stop words +Lemmatization+ POS Tags	83.8213
Top Unigrams+Without stop words +Lemmatization+POS Tags	84.0524
All Bigrams	74.2681
Unigrams+Bigrams+Without stop words+Lemmatization	**84.3606**

Table 5: Multi-Class SVM Baseline with Different Features

6.3 YouCat Evaluation

Experiments are performed on the videos *with* and *without user comments* as well as *with* and *without concept expansion*, to find out their effectiveness in video categorization. The system does not tag every video. It will not tag a video if it does not find a clue in the video descriptor that is present in the seed list or the concept list (*i.e.* the scores are all *zero*); or when there are ties with scores for multiple genres being equal. The precision, recall and f_1-score for each genre are defined as:

$$precision = \frac{number\ of\ videos\ correctly\ tagged}{number\ of\ videos\ tagged} \times 100$$

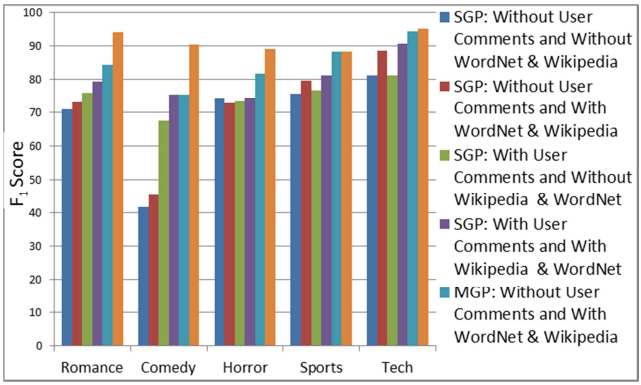
$$recall = \frac{number\ of\ video\ correctly\ tagged}{number\ of\ videos\ present\ in\ the\ genre} \times 100$$

$$f_1\ score = \frac{2 * precision * recall}{precision + recall}$$

Graph 1 shows the incremental *f_1-score* improvement for each of the genres with and without *concept expansion* as well as with and without incorporating *user comments*. It also shows the genre-wise f_1-score improvement for multi-genre prediction model.

[5] http://www.csie.ntu.edu.tw/~cjlin/libsvm/

The prediction is taken to be correct if the originally labeled tag is one of the predicted tags in a multi-genre prediction model. It may seem that the performance improvement for multiple genre identification, in our case, is trivial to achieve as the system can achieve 100% accuracy by simply assigning all the given genres to a video. This is because the prediction is taken to be correct if *any* of the predicted tags matches with the labeled tag. Thus an important performance measurement parameter is the *number of predicted tags* for each video. *Table 6* shows the average number of predicted tags for each video in each genre, with and without user comments.



SGP: Single Genre Prediction, MGP: Multiple Genre Prediction

Graph 1: Genre-wise F_1-Score Improvement for Different Models

Genre	Average Tags/Video Without User Comments	Average Tags/Video With User Comments
Romance	1.45	1.55
Comedy	1.67	1.80
Horror	1.38	1.87
Sports	1.36	1.40
Tech	1.29	1.40
Average	**1.43**	**1.60**

Table 6: Average Predicted Tags/Video in Each genre

Table 7 shows the confusion matrix when single genre prediction is done with User Comments, Wikipedia & WordNet. *Table 8* shows average f_1-score for the different models used.

Genre	Romance	Comedy	Horror	Sports	Tech
Romance	80.16	8.91	3.23	4.45	3.64
Comedy	3.13	77.08	3.47	9.03	7.29
Horror	10.03	9.34	75.78	3.46	1.38
Sports	0.70	7.30	0	89.05	2.92
Tech	0.72	5.07	0.36	1.81	92.03

Table 7: Confusion matrix for Single Genre Prediction

Model	Average F_1 Score
Multi-Class SVM Baseline: With User Comments	84.3606
Single Genre Prediction : Without User Comments + Without Wikipedia & WordNet	68.76
Single Genre Prediction : With User Comments + Without Wikipedia & WordNet	74.95
Single Genre Prediction : Without User Comments + With Wikipedia & WordNet	71.984
Single Genre Prediction : With User Comments+ With Wikipedia &WordNet	*80.9*
Multi Genre Prediction : Without User Comments + With Wikipedia & WordNet	84.952
Multi Genre Prediction : With User Comments + With Wikipedia & WordNet	**91.48**

Table 8: Average F_1-Score of Different Models

7 EVALUATION

7.1 Multi-Class SVM Baseline

The SVM has been taken as the baseline as it is found to perform the best in text classification and video categorization works. Ignoring stop words in the feature vector improved the accuracy of SVM over the all-unigram feature space. Further accuracy improvement is achieved by lemmatization. This is because all the related unigram features like *laugh, laughed, laughing etc.* are considered as a single entry *laugh*, which reduces the sparsity of the feature space.

The part-of-speech information further increased accuracy, as they help in *crude* word sense disambiguation. Consider the word *haunt* which has a noun synset and gloss as {*haunt, hangout, resort, repair, stamping ground -- (a frequently visited place)*}. It also has 3 verb synsets where the first verb sense is {*haunt, stalk -- (follow stealthily or recur constantly and spontaneously to; "her ex-boyfriend stalked her"; "the ghost of her mother haunted her")*}. Using POS information, the word haunt will have two entries now corresponding to *Noun_haunt* and *Verb_haunt*. Although the second sense is related to the *Horror* genre, the first sense is not which can only be differentiated using the POS tags.

Top unigrams help in pruning the feature space and removing noise which helps in accuracy improvement. Using *only bigrams* however decreases the accuracy as many unrelated pairs are captured which do not capture the domain characteristics. Using bigrams along with unigrams gives the highest accuracy. This is because the entities like *Michael Jordon* can be used as features as a whole, unlike in unigrams.

7.2 Overall Accuracy

Our system could not beat the multi-class SVM baseline of *84.36%* in single genre prediction; but it nevertheless achieved an f_1 score of *80.9%, without using any labeled data for training.* The multiple genre prediction, however, beats the baseline with *91.48%* f_1 score.

7.3 Effect of User Comments

The user comments often introduce noise through the off-topic conversations, spams, abuses *etc.*; the slangs, abbreviations and pragmatics prevalent in the user posts make proper analysis difficult. However, an improvement of *6 percentage point* and *9 percentage point* in the f_1 score for single genre prediction (without and with concept expansion respectively) using the comments, suggest that the greater context provided by the user comments provide more clues about the genre to help in genre identification. The corresponding improvement in the multiple genre prediction using concept expansion is around *7 percentage point*.

When concept expansion is not used, user comments contribute a performance improvement of *5 percentage point* in Romance, *1 percentage point* in Sports and a huge *26 percentage point* in Comedy. This suggests that the user information mostly helps in identifying funny videos, as well as romantic videos to some extent. Horror videos undergo mild performance degradation by incorporating user comments. Using concept expansion, user comments contribute an accuracy improvement of *6 percentage point* in Romance, a huge *30 percentage point* in Comedy and *2 percentage point* in the other genres.

7.4 Effect of Concept Expansion

In the genre identification task, using a seed set for each genre runs the risk of *topic drift*. This may occur as a concept may be identified to belong to an incorrect genre due to off-topic words by considering a larger context. However, less weightage is given to concept expansion than to a direct match in the seed list to alleviate this risk. In single genre prediction using concept expansion, an f_1 score improvement of 3 *percentage point* (when user comments are not used) and 6 *percentage point* (when user comments are used) show that Wikipedia and WordNet help in identifying unknown concepts with the help of lexical and world knowledge.

When user comments are not used, concept expansion contributes a performance improvement of *3 percentage point* in Romance, *4 percentage point* in Comedy, Sports and 7 *percentage point* in Tech. This suggests that the external knowledge sources help in easy identification of new technological concepts. Horror videos undergo mild performance degradation. Using the comments, concept expansion contributes an improvement of *8 percentage point* in Comedy and

9 percentage point in Tech. Again, the performance improvement in Comedy using Wikipedia can be attributed to the identification of the concepts like *Rotfl, Lolz, Lmfao etc.*

7.5 Average Number of Tags per Video in Multiple Genre Prediction

The number of predicted tags in multiple genre identification for each video, on an average, is *1.43* and *1.6* in the two cases (without and with user comments). This suggests that mostly a single tag and in certain cases bi-tags are assigned to the video. It is also observed that the average number of tags per video increases when user comments are used. This is due to the greater contextual information available from user comments leading to genre overlap.

7.6 Confusion between Genres

The confusion matrix indicates that Romantic videos are frequently tagged as Comedy. This is often because many Romantic movies or videos have light-hearted Comedy in them, which is identifiable from the user comments. The Horror videos are frequently confused to be Comedy, as users frequently find them funny and not very scary. Both Sports and Tech videos are sometimes tagged as Comedy. The bias towards Comedy often arises out of the off-topic conversation between the users in the posts from the *jokes, teasing etc.* Overall, from the precision figures, it seems Sports and Tech videos are easy to distinguish from remaining genres.

7.7 Issues

Many named entities in the Youtube media, especially unigrams, are ambiguous. Incorrect concept definition retrieval from the Wikipedia, arising out of ambiguity may inject noise into the system or can be ignored. For Example, a Sports video with the title "*Manchester rocks*" refers to the *Manchester United Football Club*. But Wikipedia returns a general article on the *city of Manchester* in England. None of the words in its definition matches any word in seed word lists and the entity is ignored.

Considering only WordNet synsets gives less coverage. Considering the gloss information helps to some extent. For example, if the word "*shot*" is not present in the seed list for *Sports*, then "*dunk*" cannot be associated to the *Sports* genre. But this association can be properly captured through the gloss of the WordNet first sense of "*dunk*" (- *a basketball shot in which the basketball is propelled downward into the basket*). However, it runs the risk of incorporating noise. Consider the word *good* and the gloss of one of its synsets {*dear, good, near -- with or in a close or intimate relationship*}. Here the word "*good*" is associated to *Romance* due to the presence of "*relationship*", which is incorrect.

Uploader provided video meta-data is typically small and require concept expansion to extract useful information. User comments provide a lot of information but incorporate noise as well. Auto-generated bot advertisements for products, off-topic conversation between users, fake urls, mis-spelt words, different forms of slangs and abbreviations mar the accuracy. For example, an important seed word for the *Romance* genre will not be recognized if "*love*" is spelt as "*luv*", which is common.

8 CONCLUSION

In this work, we propose a weakly supervised system, *YouCat*, for predicting *possible genre tags* for a video using the *video title*, *meta description* and the *user comments*. *Wikipedia* and *WordNet* are used for expanding the extracted concepts to detect cue words from a genre-specific seed set of words. The weak supervision arises out of the usage of a root words list (~ 1-3 words) used to describe the genre, usage of WordNet which is manually tagged and the simple parameter setting for the model. There are a number of parameters which have been simplistically set. Tuning the parameters using labeled data may improve the accuracy. An accuracy of *80.9%* in single genre prediction and *91.48%* in multiple genre prediction is obtained without using *any labeled data*, compared to the supervised multi-class SVM baseline of *84.36%* in single genre prediction. The accuracy suffers due to the inherent noise in the Youtube media arising out of the user comments and incorrect concept expansion due to ambiguity. A pre-processing filter that allows only relevant user comments about the video and a WSD module will boost the performance of the system. This work is significant as it does not use any manually labeled data for training and can be automatically extended for multiple genres with minimal supervision. This work also exhibits the usefulness of user information and concept expansion though WordNet and Wikipedia in video categorization.

References

1. Bishop, Christopher M. Pattern Recognition and Machine Learning (Information Science and Statistics). 2006. Springer-Verlag New York, Inc.
2. Borth, Damian and Hees, J\"{o}rn and Koch, Markus and Ulges, Adrian and Schulze, Christian and Breuel, Thomas and Paredes, Roberto, TubeFiler – an Automatic Web Video Categorizer, Proceedings of the 17th ACM international conference on Multimedia, MM '09
3. Cui, Bin and Zhang, Ce and Cong, Gao, Content-enriched classifier for web video classification, Proceedings of the 33rd international ACM SIGIR, 2010
4. Ekenel, Hazim Kemal and Semela, Tomas and Stiefelhagen, Rainer, Content-based video genre classification using multiple cues, Proceedings of the 3rd international workshop on Automated information extraction in media production, AIEMPro '10
5. Filippova, Katja and Hall, Keith B., Improved Video Categorization from Text Metadata and User Comments, Proceedings of the 34th international ACM SIGIR, 2011, pp. 835-842
6. Giuliano Armano and Alessandro Giuliani and Eloisa Vargiu, Experimenting Text Summarization Techniques for Contextual Advertising, Proceedings of the 2nd Italian Information Retrieval (IIR) Workshop, Milan, Italy, 2011
7. Huang, Chunneng and Fu, Tianjun and Chen, Hsinchun, Text-Based Video Content Classification for Online Video-Sharing Sites, Journal of the American Society for Information Science and Technology Volume 61, Issue 5, pages 891–906, 2010
8. Macdonald, Craig and Ounis, Iadh, Expertise drift and query expansion in expert search, Proceedings of the sixteenth ACM conference on Conference on information and knowledge

management, CIKM '07, 2007

9. Manning, Christopher D. and Raghavan, Prabhakar and Schtze, Hinrich. Introduction to Information Retrieval. Cambridge University Press, 2008
10. McCarthy, Diana and Koeling, Rob and Weeds, Julie and Carroll, John, Finding Predominant Word Senses in Untagged Text, Proceedings of the 42nd Meeting of the Association for Computational Linguistics (ACL'04), 2004
11. S. Zanetti, L. Zelnik-Manor, and P. Perona, A walk through the web's video clips, In Proc. of CVPR Workshop on Internet Vision, 2008.
12. Song, Yicheng and Zhang, Yong-dong and Zhang, Xu and Cao, Juan and Li, Jing-Tao, Google Challenge: Incremental-Learning for Web Video Categorization on Robust Semantic Feature Space, Proceeding MM '09 Proceedings of the 17th ACM International conference on Multimedia, 2009
13. Wu, Xiao and Ngo, Chong-Wah and Zhu, Yi-Ming and Peng, Qiang, Boosting web video categorization with contextual information from social web, World Wide Web Volume 15 Issue 2, 2012
14. Yang, Linjun and Liu, Jiemin and Yang, Xiaokang and Hua, Xian-Sheng, Multi-Modality Web Video Categorization, Proceeding MIR '07 Proceedings of the international workshop on Workshop on multimedia information retrieval, 2007
15. Yew, Jude and Shamma, David A. and Churchill, Elizabeth F., Knowing funny: genre perception and categorization in social video sharing. In Proceedings of CHI'2011. pp.297-306
16. Zhang, John R. and Song, Yang and Leung, Thomas, Improving Video Classification via YouTube Video Co-Watch Data, ACM Workshop on Social and Behavioural Network Media Access at ACM MM 2011
17. Zhang, Ruofei and Sarukkai, Ramesh and Chow, Jyh-Herng and Dai, Wei and Zhang, Zhongfei, Joint Categorization of Queries and Clips for Web-based Video Search, Proceeding MIR '06 Proceedings of the 8th ACM international workshop on Multimedia information retrieval, 2006
18. Zheshen Wang, Ming Zhao, Yang Song, Sanjiv Kumar, and Baoxin Li, YouTubeCat: Learning to Categorize Wild Web Videos, IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR), 2010.

Constrained decoding for text-level discourse parsing

Philippe Muller[1] *Stergos Afantenos*[1] *Pascal Denis*[2] *Nicholas Asher*[3]
(1) IRIT, Université de Toulouse, France
(2) Mostrare, INRIA, France
(3) IRIT, CNRS, France
{stergos.afantenos,muller,asher}@irit.fr, pascal.denis@inria.fr

Abstract

This paper presents a novel approach to document-based discourse analysis by performing a global A* search over the space of possible structures while optimizing a global criterion over the set of potential coherence relations. Existing approaches to discourse analysis have so far relied on greedy search strategies or restricted themselves to sentence-level discourse parsing. Another advantage of our approach, over other global alternatives (like Maximum Spanning Tree decoding algorithms), is its flexibility in being able to integrate constraints (including linguistically motivated ones like the Right Frontier Constraint). Finally, our paper provides the first discourse parsing system for French; our evaluation is carried out on the Annodis corpus. While using a lot less training data than earlier approaches than previous work on English, our system manages to achieve state-of-the-art results, with F1-scores of 66.2 and 46.8 when compared to unlabeled and labeled reference structures.

Keywords: Discourse Structure, Discourse Parsing, Dependency Structures, Constrained Decoding, A*.

Proceedings of COLING 2012: Technical Papers, pages 1883–1900,
COLING 2012, Mumbai, December 2012.

1 Introduction

Discourse analysis involves the identification of coherence relations between discourse units, which can be either elementary ones—typically clauses or sentences—or complex ones, spanning larger chunks of text. These larger units play similar roles to elementary ones. Coherence relations categorize discourse units in terms of their argumentative, thematic or causal links to other units. Together these relations and the units they relate form the global structure of a discourse.

This structure is important as it reflects the thematic organization at various levels of granularity, and constrains semantic interpretations, for instance with respect to anaphora resolution or temporal interpretation. Discourse processing is thus a crucial part of natural language understanding, and it has potentially many applications—opinion detection and classification, question answering, information extraction, recognizing textual entailment, evaluating text coherence, or knowledge extraction to name a few (Stede, 2011; Verberne et al., 2007; Somasundaran et al., 2009; Lin et al., 2011; Gerber et al., 2010)

Producing a discourse structure automatically is a complex task, however. Coherence involves syntactic and lexical factors, and relies heavily on the semantic interpretation of its parts. Most existing work tends to focus on restricted aspects of the full problem.

The task of building a discourse structure involves three subtasks: (1) identifying discourse units (DUs), (2) "attaching" DUs to one another, and (3) labeling their link with a coherence relation. Of these, the first one is usually considered easiest (see for instance Hernault et al., 2010), the second is arguably the hardest; and as a consequence, researchers have focussed attention on the third one, labelling discourse relations (Lin et al., 2009; Feng and Hirst, 2012; Sporleder and Lascarides, 2008; Wellner et al., 2006; Louis et al., 2010). Research on discourse structure also divides into two orthogonal categories: some researchers limit themselves to intra-sentential discourse structure (Wellner et al., 2006; Sagae, 2009; Joty et al., 2012); others tackle the problem of identifying the full discourse structure of a text (Hernault et al., 2010; Subba and Di Eugenio, 2009). The latter rely on "local" models to predict potential coherence relations, assuming independence between the decisions, and build the structure guided by greedy heuristics. The exception is (Baldridge and Lascarides, 2005), who use a generative model, in the case of specific task-oriented dialogues.

In this paper, we propose a more general approach to discourse structure prediction at the document level: (i) it performs a global search over the space of possible structures and optimizes a global criterion over the set of potential coherence relations; (ii) it can also take into account linguistically motivated constraints on the predicted structure. Specifically, our approach relies on the A* search algorithm, which is particularly well suited in allowing to capture constraints such as the so-called Right Frontier Constraint (or RFC), various versions of which have been a staple of theoretical linguistic approaches to discourse (Polanyi, 1988).

Another contribution of our paper is to provide a simple formalism that captures many of the commonalities across particular representation models for full discourse structure by considering a more general graph-based model, which can nevertheless integrate framework specific constraints. Previous work relies on several different discourse representation theories and corpora—e.g., the Penn Discourse Tree Bank (PDTB) of (Prasad et al., 2008), which assumes a light-weight linear structure; Rhetorical Structure Theory (RST) and the RST treebank (Carlson et al., 2003), which assumes a constituent tree structure; Segmented Discourse Representation Theory (SDRT) from which derive the Discor and Annodis corpora

(Baldridge et al., 2007; Afantenos et al., 2012), with directed acyclic graphs; or GraphBank (Wolf and Gibson, 2006), with little if any constraints on a graph-based structure.

Incidentally, we also deliver the first discourse parsing system for French. Our evaluation is performed on the ANNODIS corpus (Afantenos et al., 2012).

The paper is structured as follows. Section 2 positions in more detail our work with respect to the existing literature on discourse parsing. Section 3 describes our approach to discourse structure "decoding". Section 4 introduces the data we use from the Annodis corpus, and sections 5-6 present our experiment design. Finally section 7 reports our results and an analysis, especially with respect to comparable work.

2 Related work

Apart from a few exceptions, research on automatic discourse analysis has focused on specific aspects of the general problem. Most work concerns the task of discourse relation labeling between pairs of DUs. Examples of this line of work are: Marcu and Echihabi (2002), Sporleder and Lascarides (2005) and Lin et al. (2009). This setting makes an unwarranted assumption, as it assumes one decision concerning labeling is independent from another. Alternatively, researchers have considered the task of predicting full discourse structures, but only at the sentence level. An example of sentence-level discourse parsing is Soricut and Marcu (2003), which makes use of dynamic programming along with a standard bottom-up chart parsing. In this case, probabilities for each sentence level discourse tree are calculated as the product of the probabilities for the structure and the relation. More recently, Sagae (2009) has developed a shift-reduce algorithm for intra-sentential discourse analysis. Their stack contains the current discourse subtree and it consumes a sequence of EDUs. Like Soricut and Marcu (2003), Sagae (2009) use the RST Discourse Treebank. RST trees are "lexicalized" through head percolation using the so-called nucleus/satellite distinction. The shift operation removes the next EDU from the sequence and pushes a subtree containing only that EDU onto the stack. The reduce operation is either unary or binary: unary reduce just pops the stack and pushes a subtree with the popped item as the only child and it's lexicalized head as its mother, while the left and right binary reduce operations pop two nodes, attach them and push them back—left and right being used to judge nuclearity.

There are two main reasons why the full task of discourse parsing has eluded NLP researchers. The first is that annotating discourse structures is a very expensive procedure. There are but modest amounts of data that have been annotated, and structured prediction views each document as a single instance. Consequently, training is not very reliable. The second reason is that the two largest discourse-annotated corpora (the PDTB and the RST corpus) enforce strong constraints on the structure (namely attachment to adjacent DUs) and are thus naturally biased toward local approaches where only attachment to the left or right DU should be considered, ignoring the interdependence of local decisions. This problem can be tackled more easily at the sentence-level, where structures are simpler with only a few discourse units. Sentences can be considered independently of one another, and provide more training instances and more reliable predictions.

Among the few attempts to build document-level discourse parsers are Subba and Di Eugenio (2009) and duVerle and Prendinger (2009). Like Sagae (2009), Subba and Di Eugenio (2009) use a transition-based approach. As in the intra-sentential work cited above, the

shift operation places the next segment on top of the stack, but there is only a *binary* reduce operation which may result in the triggering of more reduce operations. In case no reduce operation is triggered then a shift is automatically performed. Consequently, only the reduce operations need to be learned. In case that the input string is empty but the stack is not, a reduce with the relation LIST is (continuously) performed. Subba and Di Eugenio (2009) use rich linguistic features and Inductive Logic Programming for training. All results are reported on an in-house corpus, and they barely surpass the baseline that consists in always attaching to the last DU.

duVerle and Prendinger (2009), and its sequel Hernault et al. (2010) both rely on locally greedy methods, and in line with all previous works, treat attachment prediction and relation label prediction as independent problems. Specifically, they start by computing probabilities of attachments for adjacent pairs of EDUs and greedily select the highest scoring. A second classifier, applied in cascade, determines the relation for the two DUs. The pair is replaced by the created "span" and the procedure continues in a bottom up way. The recent work of Feng and Hirst (2012) extend this approach by additional feature engineering but is restricted to sentence-level parsing. These three papers all use the RST-DT.

Joty et al. (2012) deserve special mention because they consider inter-dependence of local decisions, but nonetheless limit themselves on the level of intra-sentential parsing. As a first step, they compute the *joint* probabilities for structure plus relation for all possible combinations of structures and relations within a single sentence. For the computation of those joint probabilities they use a Dynamic CRF. Once all the possible joint probabilities have been computed, they perform a classic CKY chart parsing using dynamic programming. They use features representing text organization, dominance sets, contextual information and hierarchical dependencies. This is a very interesting approach but it does not easily scale up for the whole text, at least using a CRF-like approach. This is where we distinguish ourselves by adopting a local model and then a constraint-based decoding mechanism for identifying the optimal global structure.

Another relevant paper (Baldridge and Lascarides, 2005) presents a comparable problem, namely predicting the rhetorical structure of dialogues, from the Verbmobil corpus. Dialogues are considered as documents with relations between utterances, and the authors train a PCFG to produce tree representations of the dialogue structure. It is unclear how this approach, which works on very specific task-oriented dialogues, would perform in a more general framework, especially since they require some semantic features that were annotated manually.

3 Discourse decoding under constraints

In order to recover the rhetorical structure of a document, we take as our starting point two locally-trained classifiers predicting the attachment of discourse units and the labelling of their relations, much in the same way as (Hernault et al., 2010; Subba and Di Eugenio, 2009). Our models are also "local" in the sense that the training criterion that they optimize is still local and the features they use are defined over pairs of DUs. But we differ from previous approaches in the way we use the outputs of the local classifiers to predict the overall discourse structure, as well as in the use of constraints (such as the Right Frontier Constraint) during this "decoding" phase. Yet another difference is that we predict attachment decisions and relation labeling decisions in a joint fashion, rather than in pipeline as it is done in previous work.

Before getting into the details of the decoding, let's first consider the type of structure we intend to produce. An important challenge with discourse analysis is the lack of consensus among discourse theorists as to the relevant type of representations that one should use to encode discourse structure. These theoretical differences are directly reflected in the various existing discourse annotation corpora, as most of them are based on one particular theoretical framework. As a result, the different existing systems being trained on a specific resource are framework-specific. One of our goals in this paper is to abstract away from these differences and provide a more generic approach to the problem of discourse parsing.

Consider RST-DTs. RST representations are similar to constituent-based syntactic trees, since relational structures are recursively built bottom-up from elementary discourse units to form complex discourse units and adjacency is enforced at each level. Users of RST also often assume the so-called "nuclearity principle", by which complex RST segments have a distinguished EDU as a sort of head, a procedure similar to head percolation in syntax, when converting to a constituent-based to a dependency based representation. Other frameworks like SDRT or GraphBank assume more general structures (respectively directed acyclic graphs and graphs). One uniform way to capture commonalities between these different types of representation is to convert them into a dependency graph between EDUs. Many of the differences between frameworks can be encoded with different, additional structural constraints. To capture RST up to complex segments, one translates an RST tree recursively taking the nuclearity principle into account. SDRT also has complex segments, and we address how they can be dealt with in section 4. In SDRT, discourse interpretation is supposed to be an incremental process that respects a "right frontier constraint" (RFC) (Polanyi, 1988): discourse units are supposed to be processed one by one, and the current unit can only be attached to a node on the "right frontier" formed by the last introduced node and nodes "above" it (assuming a hierarchical structure). These theories distinguish between relations that are "coordinating" (additive relations, also called multinuclear in RST) or "subordinating" (expanding relations, or nuclear-satellite in RST). In that case subordinating relations add a level to the discourse hierarchy while coordinating stay at a given level of granularity.

Dependency structures have become very prominent for *syntactic* parsing, and a number of approaches have been applied successfully to the problem. So, a natural question is whether techniques developed therein could be directly used for discourse analysis. As noted, there have been some initial attempts at adapting shift-reduce parsers to discourse. An alternative to transition-based parsing is the graph-based parsing proposed e.g. by McDonald et al. (2005). This approach is particularly appealing since it builds upon an exact search procedure for finding the best possible dependency tree: it is the Maximal Spanning Tree (or MST) on the fully connected graph defined on the sentence words. One could in principle use this approach for discourse parsing, but as with transition-based parsing, there is no obvious way to easily integrate global constraints as the RFC.

Our solution is to express the problem of discourse parsing as a state-space search with different state-space definitions and constraints, and apply a general A* exploration strategy. A* search is shortest-path search through the state of possible results (dependency graphs in our case), which orders the search considering an estimated cost of a partial solution as the sum of the cost of the part of the solution already built and the estimated cost of the remaining part to be built. Transitions between states should be here the choice of an edge between two DUs, to be added to the desired solution. Since transition costs must

be additive, the cost of an edge will be -log its probability, as given by the models for attachment and relations. The general form of such a search is shown as algorithm 1. The estimation, or "heuristics", guarantees an optimal solution under certain conditions, the most common being that the heuristics is "admissible", i.e. it always underestimates the cost of the remaining exploration.

Algorithm 1 General A* decoding. S_0 is an initial state, problem dependent. Functions g and h are the cost of 'current' so far, and the estimated cost. An example of state generation is shown in algorithm 2

```
procedure astarSearch(S_0)
    queue ← {S_0}
    while not(queue.isEmpty()) do
        current ← removeBest(queue)                    ▷ best according to g+h(current)
        if isSolution(current) then return current
        else
            newStates ← generate(current)              ▷ well formed wrt desired constraints
            queue = queue ∪ newStates
        end if
    end while
end procedure
```

In contrast to the greedy approaches of (Hernault et al., 2010; Subba and Di Eugenio, 2009), we have more control on the solution yielded by the procedure. With an admissible heuristics, A* guarantees an optimal solution with respect to the cost function on state transitions (here, the probability of a given relation between two discourse units). Limited search can further be implemented as a special case if the state-space proves to be combinatorial, by restricting A* with a beam (a pending queue of fixed size, but then losing completeness). A* has also been used in syntactic parsing because of these advantages (Pauls and Klein, 2009). Another advantage offered by A* search and used in the previous paper, lies in its ordering of hypotheses, that easily yields the n best solutions by continuing the exploration of the state space. This is useful for instance to apply reordering or ranking techniques.

The most delicate aspect of using A* is in the heuristics chosen to guide the search. We will discuss possible heuristics in section 6, as they are rather orthogonal to the current discussion.

The state-space exploration works as an incremental building of a solution and must specify mainly: (1) a starting state for building a solution and (2) allowed states from a given state. For instance an MST approach could be implemented (inefficiently, though) with a starting state consisting of just one discourse unit, or as a fake node related to the others depending on the probability of a segment to be the head of the discourse (as is done in syntactic parsing). Following states could then only add a relation between a new node and one and only one of the already chosen nodes, until all nodes are attached.

To implement the RFC, we only need to restrict the previous procedure so that new nodes can only be attached to the set of accessible nodes assuming the RFC, see for details algorithm 2.

In case one wants to stay within the RST framework, the starting state would be empty, and new states should be built by adding a relation between two adjacent active units. Active units are all elementary units at the beginning, each EDU being replaced by a complex one

as they are attached during the decoding procedure.

Algorithm 2 Example state generation for building a tree incrementally under the RFC constraint, taking the first segment as the root. For simplicity, (1) relations are ignored since the best for each edge are incorporated in the cost evaluations, and (2) no difference is made between additive or expanding relation for updating the right frontier RF. The treatment below correspond to expanding relations only. For additive relations, the new attached unit replaces the attachment point in the RF, so the RF update should be new.RF = new.RF[:k-1] + [next].

Let I = $[u_1, ..., u_n]$ the list of elementary discourse units, in text order.
And let a general state S = $\langle V, E, RF \rangle$ where V is the list of DU to be attached, E is a set of edges making up a tree, and RF is a list of accessible nodes according to the right frontier constraint. We will note the state parts as S.V, S.E and S.RF.
Initially, S = $\langle I[2:], \emptyset, [u_1] \rangle$, as we take the first segment as the root. The following will generate a tree structure respecting the RFC, approximating a SDRT DAG:

```
procedure generate(S)
    result = ∅
    next = head(S.V)
    for k in 1 to len(S.RF) do          ▷ This loop generate hypotheses attaching 'next' to every u ∈ RF.
        new = newState()
        new.V = pop(copy(S.V))                          ▷ Next is removed from the nodes to attach.
        new.E = E ∪ (next,S.RF[k])                ▷ The attachment is added to the current structure.
        new.RF = new.RF[:k] + [next]      ▷ Next is appended to the RF after its parent, everything
                                                        else below the parent is thrown out.
        result.add(new)
    end for
    return result
end procedure
```

4 The corpus used

A number of different corpora have been annotated with discourse structures. These differ in the discourse formalism they are grounded in, and from our perspective in the type of constraint they impose on the attachment of DUs. The one with the heaviest constraints on discourse attachment, and consequently the simplest structures, is PDTB (Prasad et al., 2008), as most attachment are between *adjacent EDUs*, creating no further complex structures. To be fair, the main focus of PDTB is not discourse structure *per se*, but instead the study of explicit and implicit discourse relations—as signaled by discourse markers or absence thereof. Based on *Rhetorical Structure Theory* (RST) (Mann and Thompson, 1987; Marcu, 2000), the RST Discourse Treebank (RST-DT) does have recursive structures, but it imposes heavy constraints by still enforcing adjacency (Marcu, 2000). More specifically, in RST an EDU can be attached either to its adjacent EDUs (forming what is called a *span*) or to any other adjacent span, recursively thus creating a tree.[1] Less constrained structures are found in two other approaches, first from Wolf and Gibson (2006) (the GraphBank corpus) which creates graph structures with apparently no constraints whatsoever, second from SDRT (Asher and Lascarides, 2003) which creates directed acyclic graphs imposing only the RFC. Two different annotations campaigns have used the SDRT framework: DISCOR (Baldridge et al., 2007) for English and ANNODIS (Afantenos et al., 2012) for French.

[1] It can be the case that more than two DUs (*always adjacent*) could be attached together for relations such as LIST.

We have used the ANNODIS corpus: it is a collection of French discourse annotated newspaper and Wikipedia articles; specifically, we used the so-called "expert" annotations from the sub-corpus that deals with rhetorical relations. The chief reason for choosing a corpus based on SDRT is that it provides a compromise between simplistic approaches to discourse (attachment on adjacent DUs) and completely unrestricted approaches (such as the GraphBank corpus). SDRT manages to capture fine grained discourse phenomena, such as for example long distance attachments and pop-ups, while at the same time imposing a few constraints through the distinction between hierarchical and additive relations, such as the right frontier constraint.

The relation set used in the ANNODIS annotation campaign is a strict subset of the set of relations described in Asher and Lascarides (2003) (for example, there are no meta-relations); their semantics was also simplified in order to be accessible to naive subjects. Relation distribution is shown in table 1.

relation name	#	%	relation name	#	%
alternation	18	0.5	explanation	130	3.9
attribution	75	2.2	flashback	27	0.8
background	155	4.6	frame	211	6.3
comment	78	2.3	goal	95	2.8
continuation	681	20.3	narration	349	10.4
contrast	144	4.3	parralel	59	1.8
Eelab	527	15.7	result	163	4.9
elaboration	625	18.6	temploc	18	0.5
total # relations		3355	total # EDUs		3188
total # CDUs		1395	total # texts		86

Table 1: Corpus statistics from the ANNODIS corpus

In order to be able to apply techniques from syntactic dependency parsing, we transformed SDRT Annodis annotations into dependency graphs by replacing complex discourse units with their *recursive heads*. Annotations indeed mix EDU (elementary DU) and sets of EDUs, which are comparable to large spans in RST, with less constraints on their members, and this procedure is then another kind of head percolation. The *head* of a CDU is the highest DU in its subgraph (in terms of hierarchical/subordinate relations) and leftmost or older DU in the discourse with respect to additive/coordinate relations if there is more than one. In case that the DU is a complex one, the procedure is recursively applied until an EDU is reached, in which case that is the recursive head of the CDU. This transformation is graphically depicted in figure 1 which also contains the corresponding text and segmentation in the original French language.

5 Local models

Our discourse parsing is based on two locally-trained classifiers, one that predicts the attachment site of each DU, the other that predicts a discourse relation for attached pairs of DUs. In both cases, we trained probabilistic classifiers, using two different types of model: Naive Bayes (NB) and logistic regression (aka maximum entropy, or MaxEnt for short).[2]

[2]Pamameter estimation for the latter was performed using (Daumé III, 2004), `http://www.cs.utah.edu/~hal/megam/`. We also used utilities provided by the Orange library of (Curk et al., 2005).

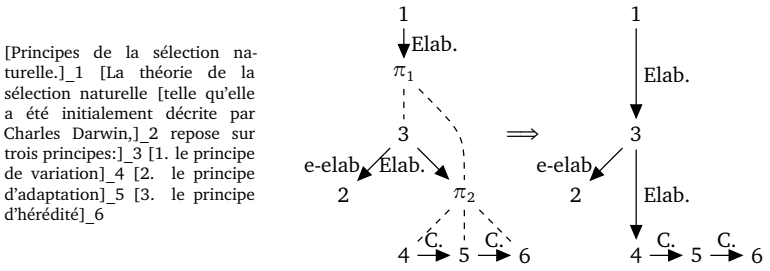


Figure 1: **An example of discourse annotation**. The nodes correspond to discourse units; the EDUs are represented by their numbering; the CDUs start with π. Dotted edges represent inclusion to a CDU while edges with arrows represent rhetorical relations. Elab. = Elaboration, e-elab = Entity Elaboration, C. = Continuation. The second graph is the result of the transformation.

The use of probabilistic models is guided by the way we combine the two models during decoding, and will be explained in Section 6.

We used two different, partially overlapping, feature sets for attachment and labeling. Overlapping features reflect an inclusion in the same sentence or paragraph, an EDU being the first of the paragraph, the number of tokens in an EDU, the number of intervening EDUs between source and target EDUs, whether the source is embedded in the target and conversely. Features specific to attachment include the presence of a discourse marker, whether the target is embedded in an EDU other than the source and a boolean feature triggered by a set of syntactic rules determining whether source (or target) is an apposition or relative clause embedded in its main clause. Features specific to labeling include: the presence of a verb, boolean features indicating which discourse relations are triggered from all discourse markers in the EDU, the syntactic category of the head token, the presence of a negation, tense agreement between head verbs of both source and target (the last three make use of syntactic dependency parses[3]) and features inspired from coreference resolution (based on pronouns and NPs).

Evaluation for the two tasks, based on 10-fold cross validation on the document level, is shown in table 2. For the relation labeling task, we use the whole set of 18 relations annotated in the Annodis corpus, but since this is a relatively small corpus, we also considered a smaller sets of relations. Following other hierarchies of relations (RST and PDTB), we grouped SDRT-inspired relations into four main groups: "structural" (parallel, contrast, alternation, conditional); "sequence" (result, narration, continuation); "expansion" (frame, elaboration, e-elaboration, attribution, comment, explanation); and "temporal" (temploc, goal, flashback, background). This corresponds roughly to the PDTB upper-level 4-way distinction, namely temporal, causal ("contingency"), comparison and expansion, with comparison being almost the same as structural without the logical relations, but the sets of fine-grain relations are somewhat different. Our 4-way coarse grain classification is also

[3]We have used Malt as a syntactic parser, trained on the french treebank `http://alpage.inria.fr/statgram/frdep/fr_stat_dep_malt.html`

more evenly distributed between relations (and instances, as it turns out).

For both tasks, we perform our experiments using a set that contains all possible pairs of EDUs, and a set that considers only pairs of EDUs whose distance is between 1 and 5 (noted "w5" for window of 5 in the table). This window was decided upon using a small development test, whose analysis revealed that around 92% of the attachment decisions fell within a window of 5 DUs. The class imbalance inherent to the attachment problem was thus reduced: the ratio of positive instances (i.e., attachments) went up roughly from 2% to 20%.

	MaxEnt	NB	Majority
w5 (18 relations)	**44.8**	34.7	19.1
full (18 relations)	43.3	32.9	19.7
w5 (4 relations)	**65.5**	62.1	51.2
full (4 relations)	63.6	60.1	50.1

	MaxEnt	NB
w5	**67.4**	61.1
full	63.5	51.3

Table 2: Relation classification accuracy (left) and F1 score for positive attachment (right), in %. For both classifications tasks the difference between Maxent and Naive Bayes is significant at $p<0.01$, using McNemar's test. The upper limit recall for the latter task in w5 configuration is 92%.

These results show that Maxent is the best model in isolation, for both tasks (it is better in both precision and recall). As expected, we also notice that the resampling increases performance in both cases; although it isn't reported here, it does however produce a small decrease in recall for the attachment task. On this task, Maxent appears to be more robust to class imbalance than NB, as shown by the relative differences between attachment F1-scores with and without resampling.

6 Parsing experiments

We divided the ANNODIS corpus in two parts: a main part and a small development set on which we had a look on the impact of some features, and most importantly on the distribution of distances between discourse units actually attached. As explained in Section 5, this leads to different sampling strategies for training the local classifiers. This will also impact decoding in pre-pruning the hypothesis space.[4]

Our various experiments are based on different combinations of classifier models (as detailed in the previous section) and decoding strategies. For attachment, we consider as instances either every pair of DUs in the same text or every pair in a distance equal to five or less. For labeling, the training is made only on attached discourse units in the training set and predictions are made on every pair tested for attachment. Features for each training procedures were detailed in section 5. As decoders, we tested a few baselines as well as MST and A*, all detailed further in this section. The last two algorithms take as input G, the complete graph over discourse units, where each edge (u, v) is labelled with the relation R having the best probability according to the relation labeling model, and the cost or weight

[4]Note that DUs are always ordered based on their left boundary. This will be important for A* decoding, which needs an ordered set of DUs in order to respect the right-frontier constraint. Practically, it means that embedded segments are attached only once their containing segment has been processed. An embedded segment is considered to be at distance one to its container.

of an edge is given by:

$$cost((u,v)) = -\log(Pr(attach(u,v) = True) \times max_R Pr(R|attach(u,v) = True))$$

This way of computing each arc cost means that we are in effect taking attachment and labeling decisions jointly, and not in a cascade as is done in the baselines.

Baselines We use two baseline decoders. The first one always selects the previous unit for attachment to the current one (noted "last"). We have also implemented a locally greedy approach similar to that of (Hernault et al., 2010). DUs are ordered based on their left boundary (embedded segments are considered to be at distance 1 from their containers). Then for all pair of adjacent units[5] (e_i, e_j) we greedily select the one that has the highest attachment probability. We remove e_i from the ordered list and continue the process until there is only one unit left in the list.

Using the Chu-Liu Edmonds algorithm The structures that result from the replacement of CDUs with their recursive heads in Annodis annotations are directed acyclic graphs, with few edges reaching a given node; they are thus very close to non-projective trees with directed arcs. Naturally then, we can apply a Maximum Spanning Tree (MST) approach, as applied by McDonald et al. (2005) in the context of syntactic dependency parsing for directed non-projective dependency trees. MST can consider an almost complete graph with a "root" as the only node with no incoming edges. In our case this node will be the first (leftmost) EDU. Using either NB or MaxEnt, we calculate the probabilities of attachment between each pair of EDUs, except for the first (root) EDU for which we calculate only the outgoing edges. We can then apply the Chu-Liu Edmonds algorithm (Chu and Liu, 1965; Edmonds, 1967), whose complexity is $O(n^3)$. MST is expected to perform well in the case that there are no additional constraints to be respected. Nonetheless, adding additional constraints is not a trivial matter.

Using the A* algorithm The general schema for A* search has been shown in section 3. Here we detail the heuristics that we have used to guide the search. In A* search, the pending queue is ordered by the estimated cost of a solution whose intermediary state is the current state s. The estimated cost is $f(S) = g(S) + h(S)$ where g is the cost of what decisions have already been taken, here the sum of the cost of selected relations so far (see above). When we build a tree under the right-frontier constraint, we have a set of discourse units yet to be attached. A heuristic yields an optimal solution if it underestimates the remaining cost (it is then an "admissible heuristic"), so that a usual way of estimating this cost is to solve the remaining problem while leaving out some constraints. To be useful the heuristics must also discriminate between comparable states, and be as close as possible to the real cost, so for instance the trivial h(s) = 0, while admissible, is useless. A more classical approach is to consider what would be the best possible decision at a given stage, if no constraint was present. For instance here, we could take, for the estimated cost of attaching a given unit, the best cost of attaching this unit to any other DU already chosen. The remaining cost of the solution is then the sum over the set of remaining DUs. Let's call this h_best. As there can be a lot of variance in the costs, another practical solution is to consider the average of attachments to every remaining nodes. This is potentially not

[5]Two units are adjacent if their distance equals 1.

admissible any more, as it can overestimate the real cost, but can yield good solutions faster. We will call this heuristics h_average. When tested on the development set, the predictions made using h_best and h_average were almost the same so we used h_average in the real experiments because it made decoding faster. Then:

$$f(S) = g(S) + h_{average}(S) = \sum_{(u,v)\in S.E} cost(u,v) + \sum_{u\in S.V} \left(\frac{\sum_{(v\in I/\{u\}} cost(v,u)}{||I||}\right)$$

For each experimental setup, we perform a document-based ten-fold cross-validation[6] on the main part of the corpus.

7 Results

This section reports on the performance obtained for our different systems. Recall that our overall goal is to evaluate labeled discourse structures, encoded here as labeled dependency graphs, produced by the different combinations of local models and decoding strategies. We also have two different training algorithms (NB vs. Maxent) and two settings based on pre-pruning possible attachment points or not. Finally, we are also interested in comparing the joint decoding of the attachments and the relation labels compared to the pipe-lining of the two procedures.

For evaluation, we take the most natural metric for dependency graphs: we compare the set of edges predicted to the reference edges, with precision, recall and F1-measure. We average on the set of all edges on all tested documents (as mentioned in the previous section, we did a cross-validation using 10 document-based folds).

Table 3 presents the results only for attachment of DUs. Besides pruning of the hypothesis space, we tested prediction of attachment alone, and prediction of attachment taking into account the probabilities of the best relations predicted to weight possible attachment (noted "joint unlabeled" evaluation in table 3). Statistical significance was tested by comparing set of scores on documents using Wilcoxon sign-ranked test for paired samples.

Training model	Naive Bayes			Maxent			∅
Decoding method	greedy	MST	A*	greedy	MST	A*	Last
attachment alone (w5)	61.2	65.7	**66.2**	62.1	65.7	65.7	62.4
attachment alone	58.5	62.0	62.1	62.2	*65.7*	*65.7*	62.4
joint/unlabelled (w5)	59.7	61.7	64.8	62.2	65.1	*65.3*	62.4
joint/unlabelled	57.9	57.0	59.6	62.3	65.1	*65.4*	62.4

Table 3: Results for unlabeled structures i.e. attachment of DUs (F1 scores, %). Windowed attachment is marked with (w5). Bold scores are the best overall while italicized scores are the best on a given setup (line). Joint unlabeled evaluation are evaluation of attachments when relations are also used to evaluate the probability of a link between DUs. A* and MST decoding do not differ significantly, but differ from all other methods. Confidence intervals at 95% are all about ± 0.9-1.2% wrt to given scores. Predicting relations does not seem to make a difference for attachment prediction.

[6]Each fold contains instances from a tenth of all documents, every document appearing in only one fold.

We see that A* and MST decoding perform at the same level without significant differences, but largely outperform all other methods. The type of learner used does not seem to make a difference in the pruned version, while Maxent is clearly better when given the whole decision space. This is clearly in line with the extra robustness noted in the classification results in the previous section.

We can observe that the best overall scores are close the F1 score for the pure attachment classification task. This seems to indicate that the impact of our global decoding strategy is not directly captured in the (edge-based) evaluation metric, and is therefore mostly a matter of exhibiting structures with desirable properties, like obeying the RFC. These might in turn be potentially useful for latter processings, such as anaphora resolution.

Training model		Naive Bayes			Maxent			
Decoding method		greedy	MST	A*	greedy	last	MST	A*
joint(w5)	4 rels	38.9	29.3	41.7	42.2	42.2	31.6	*44.1*
joint	4 rels	38.7	26.7	39.6	44.6	44.5	30.0	**46.8**
pipe-line(w5)	4 rels	39.5	42.1	42.5	42.1	42.2	*44.3*	*44.3*
pipe-line	4 rels	38.7	40.8	40.8	44.5	44.5	**46.8**	**46.8**
joint(w5)	18 rels	22.0	8.2	23.7	28.7	28.6	4.8	*30.1*
joint	18 rels	23.4	4.1	24.0	34.2	34.1	5.4	**36.1**
pipe-line(w5)	18 rels	22.5	24.0	24.5	28.7	28.6	*30.2*	*30.2*
pipe-line	18 rels	23.9	24.7	24.8	34.0	34.1	**36.1**	**36.1**

Table 4: Results of for full, labelled structures (F1 scores, %). Windowed attachment is marked with (w5). The 'last' baseline now uses a maxent model for prediction of relations. Bold scores are the best overall while italicized scores are the best on a given setup (line). Confidence intervals at 95% are all about ± 2% wrt to given scores. Best scores on each line are significantly better than the next one at $p<0.01$, except for ties. The best joint and pipe-lined scores are not significantly different from each other.

We now turn to the results on *labeled* discourse structures. The main thing to observe here is that the best decoding methods are still MST and A*, but the (expected) drop induced by relation labeling has confused the attachment results in the case of joint decoding: pipe-lining relation prediction after unlabeled attachment performs significantly better than joint decoding, at least for the best systems. It is also noticeable that pruning the attachment space is not worth it in this configuration (apart from efficiency considerations, of course). The main consequence is that we should refine our relation prediction model before drawing definite conclusions, and extend the approach to larger corpora. In hindsight, fully separating predictions within sentences or between sentences, as done in comparable work, is also something that should have been tried (we only provided features to that effect).

How can we compare these results to similar work? Taking directly Hernault et al. (2010) published scores is not easy, since they produce RST constituent trees, and use dedicated measures, namely the Parseval measures, which compares common subtrees. That is why we tried to reproduced their overall method of decoding on our data (the greedy procedure). We can still have a look at their classification scores for attachments, and the range of their evaluation for the whole structure. In their framework, it is equivalent to the tasks they call

"structure" and "nuclearity", the first being finding a pair to link, the second one being the choice of direction of the attachment, choosing the "head" of the result. When using perfect segmentation, as we do, (Hernault et al., 2010) have a F1 score of 68.4%. In (Subba and Di Eugenio, 2009), structure is predicted as relation argument spans with a F1 score of 70%, but this is similar to the attach-to-last baseline, and nuclearity is at 50% (the same baseline is at 48). It is also interesting to consider simple attachment predictions as made in (Feng and Hirst, 2012). Without knowledge of the rest of the structure, they have a F1 score of 69.84% on attachment decisions.

Again, it is hard to compare, since we don't have information about baselines comparable to ours, but we can observe the scores are in the same close ranges, while we have a simpler model with less features, trained on much less instances. Hernault et al. (2010) have 17k positively attached pairs, 77k overall, for a ratio of 23%, while we have, in the pruned version, 2.5k positive instances out of 13k (ratio $\leq$20%), and 2.7k out of 125k for the full version (2%).

For labelled structured, (Subba and Di Eugenio, 2009) report a F1 score of 35%, with 15 relations and a baseline of 22%, slightly less than our 18-relation model, while the accuracy of their relation labeler reaches 60% (a much better score than ours).

Finally, Baldridge and Lascarides (2005), who use 30 relations on the very specialized corpus of Verbmobil dialogues, report F1 scores of 68% for unlabeled structures (very close to dependency graphs), and 43% for labelled structures. The task is arguably easier since dialogues are more constrained, and since they manually annotated some features used by their probabilistic model (e.g. semantic tags, utterance mood).

Conclusions

We have proposed a general approach to discourse structure prediction at the document level, performing a global search over the space of possible structures while optimizing a global criterion over the set of potential coherence relations, in order to take into account linguistically motivated constraints on the predicted structure (e.g. the RFC). We tested this approach on a corpus of French texts which assumes theoretical aspects from SDRT, but it could and will be adapted to other type of discourse annotations, such as RST corpora, using a simple algorithm alluded to above for translating RST into dependency graphs. The results for our general approach are at least comparable to similar approaches which are arguably more specific to a given corpus and have significantly more data to train. Our relation prediction model is slightly less accurate, being induced from a smaller dataset and poorer features than the best models, but the global decoding improvements are significant over other decoding approaches, while the predictions respect the desired properties on discourse structures. Besides improving our relation model, we intend to separate completely the predictions of links for intra-sentential discourse units from other relations, as many approaches have shown that sub-problem to be much easier. We also intend to have a more structured approach to learning the structures, first by integrating decoding within the learning phase, as is done in the syntactic analysis literature we took inspiration from, and secondly by studying the usefulness of k-best parsing on the overall result.

Acknowledgments

We wish to thank our colleagues M. Serrurier and J. Mengin.

References

Afantenos, S., Asher, N., Benamara, F., Bras, M., Fabre, C., Ho-Dac, M., Draoulec, A. L., Muller, P., Pery-Woodley, M.-P., Prevot, L., Rebeyrolles, J., Tanguy, L., Vergez-Couret, M., and Vieu, L. (2012). An empirical resource for discovering cognitive principles of discourse organisation: the annodis corpus. In Calzolari, N., Choukri, K., Declerck, T., Doğan, M. U., Maegaard, B., Mariani, J., Odijk, J., and Piperidis, S., editors, *Proceedings of the Eight International Conference on Language Resources and Evaluation (LREC'12)*, Istanbul, Turkey. European Language Resources Association (ELRA).

Asher, N. and Lascarides, A. (2003). *Logics of Conversation*. Studies in Natural Language Processing. Cambridge University Press, Cambridge, UK.

Baldridge, J., Asher, N., and Hunter, J. (2007). Annotation for and Robust Parsing of Discourse Structure on Unrestricted Texts. *Zeitschrift fur Sprachwissenschaft*, 26:213–239.

Baldridge, J. and Lascarides, A. (2005). Probabilistic head-driven parsing for discourse structure. In *Proceedings of the Ninth Conference on Computational Natural Language Learning (CoNLL)*.

Carlson, L., Marcu, D., and Okurowski, M. E. (2003). Building a discourse-tagged corpus in the framework of rhetorical structure theory. In van Kuppevelt, J. and Smith, R., editors, *Current Directions in Discourse and Dialogue*, pages 85–112. Kluwer Academic Publishers.

Chu, Y. J. and Liu, T. H. (1965). On the shortest arborescence of a directed graph. *Science Sinica*, 14:1396–1400.

Curk, T., Demšar, J., Xu, Q., Leban, G., Petrovič, U., Bratko, I., Shaulsky, G., and Zupan, B. (2005). Microarray data mining with visual programming. *Bioinformatics*, 21:396–398.

Daumé III, H. (2004). Notes on CG and LM-BFGS optimization of logistic regression. Paper available at `http://pub.hal3.name#daume04cg-bfgs`, implementation available at `http://hal3.name/megam/`.

duVerle, D. and Prendinger, H. (2009). A novel discourse parser based on support vector machine classification. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 665–673, Suntec, Singapore. Association for Computational Linguistics.

Edmonds, J. (1967). Optimum branchings. *Journal of Research of the National Bureau of Standards*, 71B(233–240).

Feng, V. W. and Hirst, G. (2012). Text-level discourse parsing with rich linguistic features. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 60–68, Jeju Island, Korea. Association for Computational Linguistics.

Gerber, M., Gordon, A., and Sagae, K. (2010). Open-domain commonsense reasoning using discourse relations from a corpus of weblog stories. In *Proceedings of the NAACL HLT 2010 First International Workshop on Formalisms and Methodology for Learning by Reading*, pages 43–51, Los Angeles, California. Association for Computational Linguistics.

Hernault, H., Prendinger, H., duVerle, D. A., and Ishizuka, M. (2010). HILDA: A Discourse Parser Using Support Vector Machine Classification. *Dialogue and Discourse*, 1(3):1–33.

Joty, S., Carenini, G., and Ng, R. (2012). A novel discriminative framework for sentence-level discourse analysis. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 904–915, Jeju Island, Korea. Association for Computational Linguistics.

Lin, Z., Kan, M.-Y., and Ng, H. T. (2009). Recognizing implicit discourse relations in the Penn Discourse Treebank. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 343–351, Singapore. Association for Computational Linguistics.

Lin, Z., Ng, H. T., and Kan, M.-Y. (2011). Automatically evaluating text coherence using discourse relations. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 997–1006, Portland, Oregon, USA. Association for Computational Linguistics.

Louis, A., Joshi, A., Prasad, R., and Nenkova, A. (2010). Using entity features to classify implicit discourse relations. In *Proceedings of the SIGDIAL 2010 Conference*, pages 59–62, Tokyo, Japan. Association for Computational Linguistics.

Mann, W. C. and Thompson, S. A. (1987). Rhetorical Structure Theory: A Framework for the Analysis of Texts. Technical Report ISI/RS-87-185, Information Sciences Institute, Marina del Rey, California.

Marcu, D. (2000). *The Theory and Practice of Discourse Parsing and Summarization*. The MIT Press.

Marcu, D. and Echihabi, A. (2002). An unsupervised approach to recognizing discourse relations. In *Proceedings of ACL*, pages 368–375.

McDonald, R. T., Pereira, F., Ribarov, K., and Hajic, J. (2005). Non-projective dependency parsing using spanning tree algorithms. In *HLT/EMNLP*.

Pauls, A. and Klein, D. (2009). K-best a* parsing. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 958–966, Suntec, Singapore. Association for Computational Linguistics.

Polanyi, L. (1988). A formal model of the structure of discourse. *Journal of Pragmatics*, 12:601–638.

Prasad, R., Dinesh, N., Lee, A., Miltsakaki, E., Robaldo, L., Joshi, A., and Webber, B. L. (2008). The Penn Discourse TreeBank 2.0. In *Proceedings of LREC 2008*.

Sagae, K. (2009). Analysis of discourse structure with syntactic dependencies and data-driven shift-reduce parsing. In *Proceedings of the 11th International Conference on Parsing Technologies*, IWPT '09, pages 81–84, Stroudsburg, PA, USA. Association for Computational Linguistics.

Somasundaran, S., Namata, G., Wiebe, J., and Getoor, L. (2009). Supervised and unsupervised methods in employing discourse relations for improving opinion polarity classification. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 170–179, Singapore. Association for Computational Linguistics.

Soricut, R. and Marcu, D. (2003). Sentence level discourse parsing using syntactic and lexical information. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1*, pages 149–156. Association for Computational Linguistics.

Sporleder, C. and Lascarides, A. (2005). Exploiting linguistic cues to classify rhetorical relations. In *Proceedings of Recent Advances in Natural Langauge Processing (RANLP)*, Bulgaria.

Sporleder, C. and Lascarides, A. (2008). Using automatically labelled examples to classify rhetorical relations: An assessment. *Natural Language Engineering*, 14(3):369–416.

Stede, M. (2011). *Discourse Processing*. Synthesis Lectures on Human Language Technologies. Morgan & Claypool Publishers.

Subba, R. and Di Eugenio, B. (2009). An effective discourse parser that uses rich linguistic information. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 566–574, Boulder, Colorado. Association for Computational Linguistics.

Verberne, S., Boves, L., Oostdijk, N., and Coppen, P.-A. (2007). Evaluating discourse-based answer extraction for why-question answering. In *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '07, pages 735–736, New York, NY, USA. ACM.

Wellner, B., Pustejovsky, J., Havasi, C., Rumshisky, A., and Saurí, R. (2006). Classification of discourse coherence relations: an exploratory study using multiple knowledge sources. In *Proceedings of the 7th SIGdial Workshop on Discourse and Dialogue*, SigDIAL '06, pages 117–125, Stroudsburg, PA, USA. Association for Computational Linguistics.

Wolf, F. and Gibson, E. (2006). *Coherence in Natural Language: Data Stuctures and Applications*. The MIT Press.

Incremental Learning of Affix Segmentation

Wondwossen Mulugeta[1], *Michael Gasser*[2], *Baye Yimam*[1]
(1) ADDIS ABABA UNIVERSITY, Addis Ababa, Ethiopia
(2) INDIANA UNIVERSITY, Bloomington, USA
wondisho@aau.edu.et, gasser@cs.indiana.edu, bayemekonnen@gmail.com

ABSTRACT

This paper presents a supervised machine learning approach to incrementally learn and segment affixes using generic background knowledge. We used Prolog script to split affixes from the Amharic word for further morphological analysis. Amharic, a Semitic language, has very complex inflectional and derivational verb morphology, with many possible prefixes and suffixes which are used to show various grammatical features. Further segmentation of the affixes into valid morphemes is a challenge addressed in this paper. The paper demonstrates how incremental and easy-to-complex examples can be used to learn such language constructs. The experiment revealed that affixes could be further segmented into valid prefixes and suffixes using a generic and robust string manipulation script by the help of an intelligent teacher who presents examples in incremental order of complexity allowing the system to gradually build its knowledge. The system is able to do the segmentation with 0.94 Precision and 0.97 Recall rates.

KEYWORDS: Amharic, Morphology, Segmentation, Incremental Learning, ILP, Machine Learning

Proceedings of COLING 2012: Technical Papers, pages 1901–1914,
COLING 2012, Mumbai, December 2012.

1 Introduction

Amharic is a Semitic language, related to Hebrew, Arabic, and Syriac. Next to Arabic, it is the second most spoken Semitic language with around 27 million speakers (Sieber, 2005; Gasser, 2011). As the working language of the Ethiopian Federal Government and some regional governments in Ethiopia[1], most documents in the country are produced in Amharic. There is also an enormous production of electronic and online accessible Amharic documents.

One of the fundamental computational tasks for a language is analysis of its morphology, where the goal is to derive the root and grammatical properties of a word based on its internal structure. Morphological analysis, especially for complex languages like Amharic, is vital for development and application of many practical natural language processing systems such as machine-readable dictionaries, machine translation, information retrieval, spell-checkers, and speech recognition.

While various approaches have been used for other languages, Amharic morphology has so far been attempted using only rule-based methods. In our previous work, we have tried to apply a machine learning approach to learn morphological rules. In the experiment, we were able to learn various affixes attached to the stem and analyze the internal stem structure of the verb which is one crucial task in Semitic morphology. The major limitation of the work concerns words made up of the stem and more than one adjacent prefix or suffix; in those cases the system fails to segment the affixes. This work presents the continuation of our previous system and attempts to further segment the affixes into valid prefixes and suffixes using generic and incremental learning without any initial knowledge of the prefixes and suffixes of the language.

2 Amharic Verb Morphology and Affixation

Amharic, with all its complex word formation nature, has been more or less thoroughly studied by linguists (Sieber, 2005; Dawkins, 1960; Bender, 1968). In addition to lexical information, the morphemes in an Amharic verb convey subject and object person, number, and gender; tense, aspect, and mood; various derivational categories such as passive, causative, and reciprocal; polarity (affirmative/negative); relativization; and a range of prepositions and conjunctions.

2.1 Amharic Verb Morphology

For Amharic, like most other languages, verbs have the most complex morphology. In addition to the affixation, reduplication, and compounding common to other languages, in Amharic, as in other Semitic languages, verb stems consist of a root + vowels + template merger (e.g., sbr + ee + CVCVC, which leads to the stem seber ሰበር[2] 'broke') (Yimam, 1995; Bender, 1968). This non-concatenative process makes morphological analysis more

[1] Some of these are: Addis Ababa City Council, Amhara Region, Benishangul-Gumuz Region and Dire Dawa Administrative Council

[2] Amharic is written in the Geez writing system. For our morphology learning system we romanize Amharic orthography, and we cite these romanized forms in this paper.

complex than in languages whose morphology is characterized by simple affixation. The affixes also contribute to the complexity. Verbs can take up to four prefixes and up to five suffixes, and the affixes have an intricate set of co-occurrence rules.

Grammatical features on Amharic verbs are not only shown using the affixes. The intercalation pattern of the consonants and the vowels that make up the verb stem will also be used to determine various grammatical features. For example, the following two verbs have the same prefixes and suffixes and the same root while the pattern in which the consonants and the vowels intercalate is different, resulting in different grammatical information.

*?-**sebr**-alehu* (እሰብራለሁ)→ *1s pers. sing. simplex imperfective*
Gloss*: 'I will break'*
*?-**seber**-alehu* (እሰበራለሁ)→ *1stpers.sing.passive imperfective*
Gloss*: 'I will be broken'*

FIGURE 1 – Stem template variation example

In this second case, the difference in grammatical feature is due to the affixes rather than the internal root template structure of the word.

*te-**seber**-ku* (ተሰበርኩ)→ *1st pers. sing. passive perfective*
Gloss*: 'I was/have been broken'*
***seber**-ku* (ሰበርኩ)→ *1st pers. sing. simplex perfective*
Gloss*: 'I broke'*

FIGURE 2 – Affix variation example

As in many other languages, Amharic morphology is also characterized by alternation rules governing the form that morphemes take in particular environments. The alternation can happen either at the stem affix intersection points or within the stem itself. Suffix-based alternation is seen, for example, in the second person singular feminine imperfect and imperative. Amharic is also characterized by alternation between morphemes of the affixes. For example, the prefix ***'ye'*** if it comes before the negative prefix ***'al'***, alternation occurs and the form changes to ***'yal'*** where the ***'e'*** sound gets deleted.

2.2 Affixation in Amharic

Languages having multiple morphemes concatenated to form prefixes and suffixes show some interrelationship and co-occurrence sequences. Affixes have predefined slots in a language. The slots constrain the occurrence of the affixes, and a generic morphological learning system should be flexible enough to learn the slots and the morphemes that can fill each of them. Such morphology learning systems may be unsupervised (Goldsmith, 2001; Hammarström & Borin, 2011; De Pauw & Wagacha, 2007) or supervised (Oflazer *et al* 2001; Kazakov, 2000). Unsupervised systems are trained on unanalyzed word forms and have the obvious advantage of not requiring segmented data. The segmentation result will help to learn rules by using thin supervised examples.

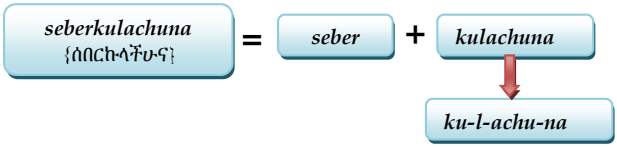


FIGURE 3 – Stem Suffix Analysis Example

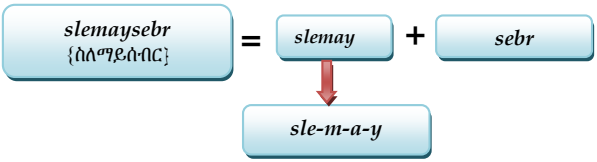


FIGURE 4 – Stem Prefix Analysis Example[3]

3 Incremental Affix Segmentation and ILP

Incremental learning dictates the use of less complex structures to be learned at early stages and move on to more complex and sophisticated structures using knowledge of previous structures as a basis. Such learning process can be implemented using Inductive Logic Programming (ILP) which is a machine learning approach that learns rules from positive and negative examples.

3.1 Incremental Learning

The problem of language acquisition has been one of the central issues in cognitive science, as well as in linguistics. Within the framework of Universal Grammar (Chomsky, 1981), language acquisition is assumed to be the process of setting the values of parameters, which are conceived of as innately-specified points of grammatical variations that have multiple consequences for the different aspects of the surface grammar. Here, Chomsky argued that language is so complex that the only way it could be learned is through innate constraints on what was a possible grammar using its parameters. More recently and in opposition to Chomsky's and others' innatist (or nativist) view, there have been arguments from psychologists and cognitive linguists who support empiricist theories of language acquisition. Among other things, they argue that innate constraints are not needed (this is one instance of the large "nature vs. nurture debate"). One of these simplifications come from what is "child-directed speech" (CDS), the simplified speech that adults naturally use when speaking to children. But of course adults adjust CDS as children get older, making it more and more complex. Thus, one argument claims that the only way children are able to learn language is through the graded simplification made by adults to the input that the child receives. This supports the idea of incremental learning, by which examples with less

[3] *The last two suffixes {a-y} are the result of the actual suffixes {al - y} transformed due to the assimilation process*

complex word structure are presented first and the knowledge of affixes acquired from early training becomes the basis of the more complex knowledge acquired later.

The ability to acquire and use language and its constructs is a key aspect that distinguishes humans from other beings. Learning is the process of acquiring knowledge over time from different realities we are exposed to. The same is true for acquiring language related facts and rules by human beings (Pirrelli & Herreros, 2005). The brain learns by observing, constantly labelling and creating its own rules that define or explain what has been observed. This learning process demands massive amount of data or exposure to relevant and interesting instances to deduce rules from. In cases where no such data is available or the aim is to learn from few examples, incremental learning through strategic example coverage would be suitable. Inspired by features of child language acquisition, the best way to learn language is by applying child language learning methods. Children observe and are able to identify similarities and add to their database such common features relating it to the meaning or the form of a word. For example, as presented by Sara Finley, when a child encounters words like {*dogs*, *cats*, *chairs*, *boys}*, he can discover part of the word forming feature similarity through the suffix 's', plural marker in this case (Finley, 2012). Thus, distributional cues are very important for children to find where relationship between words lie and find patterns for future use.

At early ages it is common to see children make such mistakes of segmenting part of the main word as affix or attaching affix on fully formed words. These actions are considered to be part of the learning process.

In morphology, learning constituents of a word in distributional or structural cues proved to be effective (Cavar, 2005) and be linked with incremental learning to teach the learner in a more logical manner. Such language processing by means of data-oriented methods emphasize the assumption that human language perception and production works with representations of concrete past language experiences, rather than with abstract grammar rules (Rens & Remko, 1996). The next section describes such incremental learning for affix segmentation task.

3.2 Incremental Affix Segmentation

Incremental learning of morphological affix segmentation results in knowledge acquisition when the system encounters new affixes, through the further segmentation of the string based on previous knowledge (Altenbek 2009). The first step in the segmentation process is to detach the affix from the main stem. This has been done using our previous system that employs inductive logic programming to learn stems and affixes as well as internal stem structure from examples (Mulugeta & Gasser, 2012). The system takes the main word and keeps database of valid affixes where the challenge relies on how to further analyze the affix into a valid list. In this regard, Prolog programming language is more suited for such action due to its easy knowledge acquisition and database manipulation features.

While we focus on Amharic verb affix segmentation and morphology learning, our goal is a general-purpose ILP morphology learner that automatically segments affixes based on previous knowledge during the learning process. Thus we seek background knowledge that is plausible across languages that can be combined with language-specific examples and

intelligent ways of presenting examples to yield rule hypotheses that generalize to new examples in the language.

3.3 Inductive Logic Programming

In induction, one begins with some plausible and selected examples during the training phase. Then, it determines what general conclusion can logically be derived from those examples. For morphological analysis, the learning data would be expected to guide the construction of word formation rules, the affix segmentation and interactions between the constituents of a word.

There have been only a few attempts to apply Inductive Logic Programming (ILP) to morphology. Most of these have dealt with languages with relatively simple morphology handling few affixations (Kazakov, 2000; Manandhar et al, 1998; Zdravkova et al, 2005). These attempts consider the affixes extracted as one singleton morpheme which is not the case for complex languages like Amharic. However, their results are found to be encouraging. The enhancement to such learning systems would be the task of further analysis of the affixes. The analysis shall include but not limited to prefix and suffix knowledge capturing through segmentation process to help build database of prefixes and suffixes for deep grammar scrutiny. This learning and knowledge acquisition task has been done using CLOG.

CLOG is a Prolog based ILP system, developed by Manandhar *et al* (1998)[4], for learning first order decision lists (rules) on the basis of positive examples only. A rule in Prolog is a clause with one or more conditions. The right-hand side of the rule (the body) is a condition and the left-hand side of the rule (the head) is the conclusion. The operator between the left and the right hand side (the sign '***:-***') means ***if***. The body of a rule is a list of goals separated by commas, where commas are understood as conjunctions. For a rule (the head) to be true, all of its conditions/goals must be evaluated to be true. In the expression below, there are two ways of evaluating the goal ***p*** even with two different results. Accordingly, ***p*** is true if ***q*** and ***r*** are true or if ***s*** and ***t*** are true[5].

$$\left.\begin{array}{l} p :- q, r. \\ p :- s, t. \end{array}\right\} \quad p \Longleftrightarrow (q \wedge r) \vee (s \wedge t)$$

*Where **q**, **r**, **s** and **t** could be facts or predicates and **p** is a predicate with any number of arguments.*

CLOG relies on output completeness, which assumes that every form of an object is included in the example and everything else is excluded (Mooney & Califf, 1995). We preferred CLOG over other ILP systems because it requires only positive examples and runs faster than the other variants (Manandhar *et al*, 1998). CLOG uses a hill climbing strategy to build the rules, starting from a simple goal and iteratively adding more rules to satisfy the goal until there are no possible improvements. The evaluation of the rules generated by the learner is validated using a gain function that compares the number of positively and negatively covered examples in the current and previous learning stages (Manandhar et al, 1998).

---

[4] CLOG is a freely avalable ILP system at: (http://www-users.cs.york.ac.uk/suresh/CLOG.html)
[5] Refer to http://en.wikipedia.org/wiki/Prolog for detailed illustration on Prolog

4 Experiments and Integration with Morphology Learning System

ILP is a rarely used method for morphology and language related learning. The approach demands well crafted background knowledge with the level of depth required for supervision and a set of positive and/or negative examples to learn from. Our previous experiment, which also uses ILP, is able to extract verb morphological rules and internal stem structure as well as orthographic alternation rules from examples on Amharic subject markers (Mulugeta & Gasser, 2012). Our system takes examples of the form shown in Figure 5 and extract morphological rules based on the various generic background knowledge crafted for the learning task.

In Figure 5, the predicate *'stem'* provides a word and its stem to permit the extraction of the affixes and root template structure of the word. The first two parameters specify the input word and the stem of the word after affixes are removed. The third parameter is the codification of the grammatical features (tense-aspect-mood, voice, subject and object) of the word. The codification is a simple knowledge about the various grammatical features of the word. For example, the fourth value in the third argument represents the object marker of the word where 2 means second person singular masculine, 6 means third person plural neuter and so forth.

```
stem([s,e,b,e,r,k,u,l,h],[s,e,b,e,r] [1,1,1,2]).
stem([s,e,b,e,r,k,l,a,c,h,w],[s,e,b,e,r], [1,1,2,6]).
stem([s,e,b,e,r,x,l,n],[s,e,b,e,r], [1,1,3,8]).
```

FIGURE 5 – Sample training examples for the learning process

The background knowledge added to handle the affix segmentation and database manipulation is generic in its nature making it applicable for any language of interest. In addition, the background predicate also includes scripts for string manipulation and root extraction. Both are language-independent, making the approach adaptable to other similar languages.

The previous system is able to generate rules of the following structure by taking the examples of the form shown in Figure 5 above.

```
stem(Word, Stem, [1, 2, 7, 0]):-
        set_affix(Word, Stem, [y], [], [u], []),
        feature([1, 2, 7, 0], [simplex, imperfective, tppn, noobj]),
        template(Stem, [1, 0, 1, 1]).

stem(Word, Stem, [2, 1, 1, 2]):-
        set_affix(Word, Stem, [te], [], [kulh], []),
        feature([2, 1,1, 2], [passive, perfective, fpsn, spsm]),
        template(Stem, [1, 0, 1, 0, 1]).
```

FIGURE 6 – Learned affix identification rule example

Accordingly, the rules learned through ILP contain three major background predicates:

- The ***'set_affix'*** predicate uses a combination of multiple 'split' operations to identify the prefixes and suffixes attached to the input word. This predicate is used to learn the affixes from examples presented by taking only the Word and the Stem parameters (the first two arguments from the example). The last four arguments of ***set_affix*** predicate represent the prefix and suffix pairs in the Word and Stem parameters.
- The ***'template'*** predicate is used to extract the valid template for Stem. The predicate manipulates the stem to identify positions for the vowels. This predicate uses the list of vowels (vocal) in the language to assign '0' for the vowels and '1' for the consonants.
- The ***'feature'*** predicate is used to associate the identified affixes and root CV pattern with the known grammatical features from the example. This predicate uses a codified representation of the grammatical features in the language, which is also encoded as background knowledge. This predicate is the only language-dependent background knowledge we have used in our implementation.

The two example rules in Figure 6 show that the prefixes *[y]* and *[te]* as well as the suffixes *[u]* and *[kulh]* are extracted from the examples with the respective root template structure. The output is limited with no further segmentation of the affixes to relate it with the grammatical features for further analysis. The current experiment includes a module which tries to do affix segmentation in incremental manner. The following algorithm and script presents how the segmentation is done in an incremental manner based on the experiment setup. While the algorithm is generic for any affix presented, the illustration shown later demonstrates that the order in which the examples are presented will dictate the knowledge acquired by the learner.

For each Suffix A extracted
    Take A as a possible Suffix
    Take any nonempty leftmost segment B of A
    Check if B exists in the Suffix database
    If B is a valid Suffix
        Remove A from the Suffix database
        Assign A to be the remaining substring
        Repeat the suffix segmentation process
End if there are no strings to segment

FIGURE 7 – Suffix segmentation algorithm

```
segS([ ]):-!.
segS(A):-
        findall(D, (append(C,D,A),C\==[ ],suffix(C),segS(D)), Segs),
        Segs==[ ]->assertz(suffix(A));!.

seg_suffix([ ],[ ]).
seg_suffix(A,[C|B]):-
        append(C,D,A),
        C\==[ ],
        suffix(C),
        seg_suffix(D,B).
```

FIGURE 8 – Suffix segmentation and list generation Prolog script

* ***segS*** *segments suffixes and updates the database whenever a new suffix is identified through the assertz predicate.*
* ***seg_suffix*** *convert one affix string into a list of morphemes enclosed in a square bracket. For example, [kulhna] will be changed to [[ku],[lh],[na]] based on prior affix knowledge.*

The system works progressively by picking examples from the training data and learning the affixes, stem template structure and alternation rules. Along with affix extraction, the systems takes each affix from the example and iteratively build its database and segment upcoming affixes based on this knowledge. The following analysis and knowledge acquisition example illustrates how the affix segmentation incrementally learns a suffix based on the examples presented.

Iteration 1
Initial Suffix Database: *{Ø}*
Training Example 1: *stem([s,e,b,e,r,k,u],[s,e,b,e,r] [1,1,1,0]).*
Stem Extraction Result: *[ ], [s,e,b,e,r], [k,u]*
Affix Identification*: [k,u] *no further segmentation as the database is empty*
Updated Suffix Database: *{ suffix([k,u])}*

Iteration 2
Initial Suffix Database: *{ suffix([k,u])}*
Training Example 2: *stem([s,e,b,e,r,k,u,l,h],[s,e,b,e,r] [1,1,1,2]).*
Stem Extraction Result: *[ ], [s,e,b,e,r], [k,u,l,h]*
Affix Identification*:{[k,u], [l,h]} *as [k,u] is already identified earlier*
Updated Suffix Database: *{ suffix([k,u]), suffix([l,h])}*

Iteration 3
Initial Suffix Database: *{ suffix([k,u]), suffix([l,h])}*
Training Example 2: *stem([s,e,b,e,r,k,u,l,h,n,a],[s,e,b,e,r] [1,1,1,2]).*
Stem Extraction Result: *[ ], [s,e,b,e,r], [k,u,l,h,n,a]*
Affix Identification*:{[k,u],[l,h],[n,a]} *as [k,u] and [l,h]are already identified earlier*
Updated Suffix Database: *{ suffix([k,u]), suffix([l,h]), suffix([n,a])}*

Iteration 4
Initial Suffix Database: *{ suffix([k,u]), suffix([l,h]), suffix([n,a])}}*
Training Example 2: *stem([s,e,b,e,r,k,u,l,a,c,h,u],[s,e,b,e,r] [1,1,1,6]).*
Stem Extraction Result: *[ ], [s,e,b,e,r], [k,u,l,h,n,a]*
Affix Identification*:{[k,u],[l,a,c,h,u]} *as [k,u] is already identified earlier*
Updated Suffix Database: *{ suffix([k,u]), suffix([l,h]), suffix([n,a]), suffix([l,a,c,h,u])}*

FIGURE 9 – Suffix learning and segmentation process illustration

The illustration in Figure 9 (only suffix learning as a show case) shows that, at the beginning of the learning process (iteration 1), the prefix and suffix database is empty assuming that the first examples to be presented shall have only a single prefix and suffix. These prefixes and suffixes shall be taken as the primary knowledge acquired. The upcoming and remaining affixes shall be learned based on this previous knowledge. After iteration 4, four suffixes are leaned that are results of incremental learning starting from an empty database and systematic presentation of examples. It should be noted here that if the example in iteration 2 has been presented first, the initial database would have been *{suffix([k,u,l,h])}* and the first example with the suffix *[k,u]* would not have any impact on its previous knowledge of suffix. This indicates that the system requires the skill of the teacher to guide the learner with more logical flow.

One potential drawback of such incremental learning is the need for an intelligent teacher. What does our teacher need to know? The most important requirement is information about the number of prefixes and suffixes that a word contains. The main constraint in presentation of words is the order of the example based on the number of affixes it contains. We would expect a literate native speaker of the language with a little linguistic training to have this awareness. One advantage of the current implementation is that, the rules learned with the segmented affixes are easily understandable by linguists. This will help the teacher to restructure the examples to formulate more logical rules and segments.

5 Results and Error Analysis

ILP has proven to be applicable for word formation rule extraction for languages with simple rules like English. Our experiment shows that the approach can also be used for complex languages with more sophisticated background predicates and more examples. While Amharic has more prefixes and suffixes for various grammatical features, our system is able to further segment the affixes into possible and valid prefixes and suffixes. With 140 training examples containing words, the stem and codified morphological features, the system is able to learn and extract 6 prefixes and 25 suffixes. Moreover, the system has learned 70 stem affix extraction rules. As stated in the experiment section, one major limitation of the approach is that the system is not able to look back on previously acquired affixes and do further segmentation.

```
stem(Word, Stem, [2, 1, 1, 2]):-
        set_affix(Word, Stem, [te], [], [[ku],[lh]], []),
        feature([2, 1,1, 2], [passive, perfective, fpsn, spsm, pos]),
        template(Stem, [1, 0, 1, 0, 1]).

stem(Word, Stem, [2, 1, 1, 2]):-
        set_affix(Word, Stem, [[al],[te]], [], [[ku],[lh],[m]], []),
        feature([2, 1,1, 2], [passive, perfective, fpsn, spsm, neg]),
        template(Stem, [1, 0, 1, 0, 1]).
```

FIGURE 10 – New Rules with Affix Segmentation Result

Another main advantage of such learned rules is the ability to review the rules and be verified by a linguist for correctness. This benefit can also be used by the teacher to decide on how to order the examples for better knowledge acquisition. If a certain example presentation generates wrong or incorrect segmentation, the teacher can easily rearrange the list of examples and see a corrected or better segmented result. Such experiments by the example provider could also help to identify complexity of word structure which is taken to be trivial for experts.

An experiment was also done to see the effect of order of examples has on the segment learning predicate. The attempt showed that most of the errors in the segmentation process arise from the ordering of examples. Thus, for example, if the system encounters the affix *[kulh]* before *[ku]*, then, the system puts *[kulh]* in its database of suffixes rather than segmenting it into *[[ku],[lh]]*. With the same token for the suffixes *[ku]* and *[k]* which are subject first person and second person masculine markers, the order of presentation might confuse the learner. If the system encounters a word with the suffix *[k]* first, then the second suffix will be spuriously segmented into *[[k], [u]]* .This necessitates that example presentation to the system should be done in an incremental way by an intelligent teacher. One of the limitations of the system, as explained above, is lack of correcting previously acquired knowledge to reformulate such rules.

In Amharic, as in many other languages with multiple affixes, the affixes may change their form in particular phonological or orthographic environments. In finite-state morphology, these changes are captured in alternation rules. Although our ILP system succeeds in learning some of the alternation rules that play a role in root-template combination and at the boundaries between stems and prefixes or suffixes, we have not yet incorporated the learning of alternation rules into the component of the system that learns to segment prefixes and suffixes through incremental presentation of examples. In the future, we will experiment with the possibility of taking advantage of the teacher's knowledge of the alternate forms of an affix to learn from successive presentations of the same affix in different environments.

The other limitation of ILP for morphology learning is the inability to learn rules from incomplete examples. In languages such as Amharic, there is a range of complex interactions among the different morphemes, but we cannot expect every one of the thousands of morpheme combinations to appear in the training set. When examples are limited to only some of the valid morpheme combinations, CLOG is inadequate because it is not able to use variables as part of the body of the predicates to be learned.

To measure the effectiveness of the system, precision and recall is used to see the ratio of valid segmentation that exists in the data with the segmentation done by the system. From the 140 words provided to the system, a linguist extracted 221 valid segmentations. It should be noted here that some of the segmentations might appear in a number of instances in the data. From the same data set the system is able to extract 227 segmentations while 215 of this segmentation are correct segmentations that match with the linguist's analysis. The system has shown over generation of segments. The following figure shows the precision and recall values according to the statistics presented above.

$$Precision = \frac{Number\ of\ Correct\ Segmentation}{Total\ Number\ of\ Segmentations\ Found} = \frac{215}{227} = 0.94$$

$$Recall = \frac{Number\ of\ Correct\ Segmentation}{Total\ Number\ of\ Segmentations\ in\ the\ Data} = \frac{215}{221} = 0.97$$

FIGURE 11 – Precision and Recall result

The precision and recall result is satisfactory enough to implement the system in large scale examples and words with more complexity.

We are currently implementing the concept of mutual information to build knowledge from already known facts and rules entertaining partial information. The concept of partial information could be integrated with ILP to generate more rules from rules handling features and affix co-occurrences not found in the training example.

Conclusion

We have shown in this paper that ILP can be used to fast-track the process of learning morphological rules of complex languages like Amharic with a relatively small number of examples. Our experiment also showed that affixes could further be segmented into possible valid prefixes and suffixes during the learning process by building knowledge of affixes on the fly using incremental learning methods. Our previous implementation have gone beyond simple affix identification and confronts one of the challenges in template morphology by learning the root-template extraction as well as stem-internal alternation rule identification exhibited in Amharic and other Semitic languages. The current update aiming on affix manipulation also succeeds in segmenting affixes into valid prefixes and suffixes using database generated on the fly during the training phase. As the rules and segmentations are presented in easy and human understandable list of rules, the teacher could restructure the examples and feed the learner to gain better result as needed.

References

Altenbek, G. (2009). *Kazakh Segmentation System of Inflectional Affixes.* Xinjiang University, Harbin, China: Natural Science Foundation of China

Armbruster, C. H. (1908). *Initia Amharic: an Introduction to Spoken Amharic.* Cambridge: Cambridge University Press.

Beesley, K. R. and L. Karttunen. (2003). *Finite State Morphology*. Stanford, CA, USA: CSLI Publications.

Bender, M. L. (1968). *Amharic Verb Morphology: A Generative Approach.* Ph.D. thesis, Graduate School of Texas.

Bratko, I. and King, R. (1994). *Applications of Inductive Logic Programming*. SIGART Bull. 5, 1, 43-49.

Cavar D, Rodrigues P. and Schrementi G. (2005), *Using Morphological and Distributional Cues for Inductive Part-of-Speech Tagging*, CiteSeer.

Chomsky, Noam. (1981). *Lectures on government and binding.* Dordrecht, Dordrecht Foris

Dawkins, C. H. (1960). *The Fundamentals of Amharic*. Sudan Interior Mission, Addis Ababa, Ethiopia.

De Pauw, G. and P.W. Wagacha. *(2007). Bootstrapping Morphological Analysis of Gĩkũyũ Using Unsupervised Maximum Entropy Learning.* Proceedings of the Eighth INTERSPEECH Conference, Antwerp, Belgium.

Finley Sara (2012), *Morpheme Segmentation in School-Aged Children.* In A. Fine (Ed.) University of Rochester Working Papers in the Language Science Vol 6.

Gasser, M. (2011). *HornMorpho: a system for morphological processing of Amharic, Oromo, and Tigrinya*. Conference on Human Language Technology for Development, Alexandria, Egypt.

Goldsmith, J. (2001). *The unsupervised learning of natural language morphology.* Computational Linguistics, 27: 153-198.

Hammarström, H. and L. Borin. (2011). *Unsupervised learning of morphology*. Computational Linguistics, 37(2): 309-350.

Kazakov, D. (2000). *Achievements and Prospects of Learning Word Morphology with ILP*, Learning Language in Logic, Lecture Notes in Computer Science.

Kazakov, D. and S. Manandhar. (2001). *Unsupervised learning of word segmentation rules with genetic algorithms and inductive logic programming*. Machine Learning, 43:121–162.

Koskenniemi, K. (1983). *Two-level Morphology: a General Computational Model for Word-Form Recognition and Production.* Department of General Linguistics, University of Helsinki, Technical Report No. 11.

Manandhar, S. , Džeroski, S. and Erjavec, T. (1998). *Learning multilingual morphology with CLOG.* Proceedings of Inductive Logic Programming. 8th International Workshop in

Lecture Notes in Artificial Intelligence. Page, David (Eds) pp.135–44. Berlin: Springer-Verlag.

Mooney, R. J. (2003). *Machine Learning*. Oxford Handbook of Computational Linguistics, Oxford University Press, pp. 376-394.

Mooney, R. J. and Califf, M.E. (1995). *Induction of first-order decision lists: results on learning the past tense of English verbs*, Journal of Artificial Intelligence Research, v.3 n.1, p.1-24.

Mulugeta, W and Gasser, M. (2012). *Learning morphological rules for Amharic verbs using inductive logic programming*. SALTMIL-AfLaT Workshop on Language Technology for Normalisation of Less-Resourced Languages. Istanbul.

Oflazer, K., M. McShane, and S. Nirenburg. (2001). *Bootstrapping morphological analyzers by combining human elicitation and machine learning*. Computational Linguistics, 27(1):59–85.

Pirrelli V and Herreros I, (2007), *Learning Morphology by Itself*, On-line Proceedings of the Fifth Mediterranean Morphology Meeting (MMM5). Fréjus 15-18 , University of Bologna.

Rens Bod, Remko Scha (1996) *Data-Oriented Language Processing: An Overview*. ILLC Technical Report LP-96-13, Department of Computational Linguistics, University of Amsterdam

Sieber, G. (2005). *Automatic Learning Approaches to Morphology*, University of Tübingen, International Studies in Computational Linguistics.

Zdravkova, K., A. Ivanovska, S. Dzeroski and T. Erjavec, (2005). *Learning Rules for Morphological Analysis and Synthesis of Macedonian Nouns.* In Proceedings of SIKDD 2005, Ljubljana.

Yimam, B. (1995). *Yamarigna Sewasiw (Amharic Grammar).* Addis Ababa: EMPDA.

Semi-supervised Noun Compound Analysis with Edge and Span Features

Yugo MURAWAKI Sadao KUROHASHI
Graduate School of Informatics, Kyoto University
Yoshida-honmachi, Sakyo-ku, Kyoto, 606-8501, Japan
{murawaki, kuro}@i.kyoto-u.ac.jp

ABSTRACT
In this paper, we propose the use of *spans* in addition to edges in noun compound analysis. A span is a sequence of words that can represent a noun compound. Compared with edges, spans have good properties in terms of semi-supervised parsing. They can be reliably extracted from a huge amount of unannotated text. In addition, while the combinations of edges such as sibling and grandparent interactions are, in general, difficult to handle in parsing, it is quite easy to utilize spans with arbitrary width. We show that spans can be incorporated straightforwardly into the standard chart-based parsing algorithm. We create a semi-supervised discriminative parser that combines edge and span features. Experiments show that span features improve accuracy and that further gain is obtained when they are combined with edge features.

TITLE AND ABSTRACT IN JAPANESE

スパンとエッジ特徴量を用いた半教師あり名詞句解析

名詞句解析において，エッジだけでなくスパンを手がかりとして使うことを提案する．スパンは名詞句を表しうる単語列であり，エッジと比べて半教師あり学習に適した性質を持っている．すなわち，大量の生テキストから高い信頼性をもって抽出可能である．さらに，エッジは解析時に組み合わせ (兄弟や孫の関係など) を考えることが一般に難しいのに対して，スパンは任意の長さの組み合わせを自明に利用できる．この論文では，スパンが動的計画法による標準的な構文解析手法に簡単に組み込めることを示し，エッジとスパン特徴量を組み合わせた半教師ありの識別型構文解析器を提案する．実験により，スパン特徴量が解析精度を改善し，エッジと組み合わせることでさらに精度が向上することが示された．

KEYWORDS: noun compound analysis, semi-supervised learning, parsing, span.

KEYWORDS IN JAPANESE: 名詞句解析, 半教師あり学習, 構文解析, スパン.

Proceedings of COLING 2012: Technical Papers, pages 1915–1932,
COLING 2012, Mumbai, December 2012.

1 Introduction

Words are used as a basic unit in a broad range of applications in natural language processing. However, it often happens that what we need to recognize turn out to be longer than single words. They are phrases, noun compounds in particular, that consist of more than one word.

A noun compound is not just a sequence of words but has a latent structure. Consider the following example in Japanese.

[*jidou* [*onsei ninshiki*]]
automation speech recognition
automatic speech recognition

The brackets indicate the internal structure of the noun compound. In addition to the right-branching structure, it has another possible interpretation.

[[*jidou onsei*] *ninshiki*]
automation speech recognition
recognition of automatic speech

Our goal is to recognize that the former is semantically coherent while the latter is not. In order to analyze the internal structures of noun compounds, we need some automatic method because they are too large in number and too productive to be covered by a hand-crafted lexicon. This task is called noun compound analysis.

Noun compound analysis can be seen as a task of dependency parsing (Lauer, 1995). However, it is different from usual full-sentence parsing in that part-of-speech tags help little, if at all. A noun compound is just a sequence of nouns and lacks grammatical markers. For this reason we take fully lexicalized approaches in noun compound analysis.

Fully lexicalized approaches often suffer from the data sparseness problem. Apart from the observation that nouns have much higher domain specificity than other words, a model needs to learn the lexical association of a pair of words. However, we can never create an annotated corpus that covers the combinations of tens of thousands of words.

To overcome data sparseness, it seems promising to take semi-supervised approaches that exploit a huge amount of unannotated text. In fact, recent studies have shown that web statistics greatly improve the accuracy of noun compound analysis (Lapata and Keller, 2004; Nakov and Hearst, 2005a; Bergsma et al., 2010; Pitler et al., 2010).

One problem with incorporating web statistics into the dependency model is that dependency relations (edges) are latent and cannot be observed in unannotated text. Edge counts are approximated by bigrams of successive words (Nakov and Hearst, 2005a) or rely on a search engine's `NEAR` operator (Lapata and Keller, 2004). These counts are noisy as they may not represent true dependency relations.

In this paper, we propose the use of *spans* in addition to edges. A span is a sequence of words that can form a noun compound. Spans have two advantages over edges. First, unlike noisy edges, spans can be reliably extracted from unannotated text without abandoning a large portion of data. Second, it is quite easy to handle spans with arbitrary width in a parsing model. We show that web span counts can be used straightforwardly in the standard chart-based parsing algorithm. By contrast, combinations of edges such as sibling and grandparent interactions cannot easily be incorporated into dynamic programming.

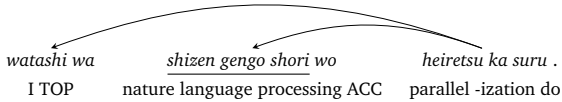


Figure 1: *Bunsetsu*-based dependency parsing for "I parallelize natural language processing." The dependency relations are drawn between *bunsetsu* phrases. The internal structure of the noun compound (underlined) is left unanalyzed.

We create a semi-supervised discriminative parser that can combine multiple factors: features learned directly from training data, web-derived edge features and web-derived span features. In addition, we introduce web-derived paraphrase features, which can be seen as mixtures of edges and spans.

Experiments show that span features improve accuracy and are robust across domains. It is also shown that the edge and span features play complementary roles. The combination of these features boost performance in out-of-domain data.

2 Related work

2.1 Background of the task

The phrase structure grammars dominated English parsing research for a long time although dependency parsing has seen rapid progress in the last decade. The most influential annotated corpus for the phrase structure grammars would be the Penn Treebank (Marcus et al., 1993). Unfortunately, the original Penn Treebank does not annotate the internal structures of noun compounds but leaves them flat. The situation changed when Vadas and Curran (2007a) added internal structures to noun compounds in the Penn Treebank and gave rise to supervised approaches to English noun compound analysis (Vadas and Curran, 2007b; Bergsma et al., 2010; Pitler et al., 2010).

Japanese parsing faces a similar situation but dependency parsing is the preferred choice in order to handle its flexible word order (Uchimoto et al., 1999; Kudo and Matsumoto, 2002). In Japanese dependency parsing, dependency relations are drawn between phrasal units called *bunsetsu* although there is an attempt at word (morpheme)-based dependency parsing (Flannery et al., 2011). In a *bunsetsu* phrase, one or more content words are followed by zero or more function words (morphemes). This means that, as illustrated in Figure 1, the internal structure of a noun compound is left unanalyzed because it is contained in a *bunsetsu* phrase. Thus noun compound analysis has a complementary relationship with full-sentence dependency parsing.

2.2 Noun compound analysis

Noun compound analysis, also called noun compound bracketing, is the task of analyzing the internal structure of a given noun compound. There is an old debate between what are called the adjacency model and the dependency model. Figure 2 compares the two models for three-word noun compounds, which were the primary focus of early work. The adjacency model (Marcus, 1980; Liberman and Sproat, 1992; Pustejovsky et al., 1993; Resnik, 1993) examines the lexical association between neighboring words. It checks if the pair of N_2 and N_3 is more strongly associated than the pair of N_1 and N_2. The dependency model (Lauer, 1995) compares the association between N_1 and N_3 against that between N_1 and N_2. Lauer (1995) and subsequent studies demonstrate that the dependency model performs better than

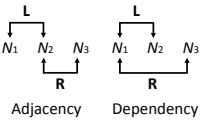


Figure 2: Adjacency and dependency models. A reproduction of Figure 1 in Lauer (1995).

the adjacency model. In this paper, however, we show that a generalization of adjacency is useful for noun compound analysis.

Since dependency relations (edges) are latent, annotated data are required to learn the true lexical association. Otherwise we need some approximation methods with which the lexical association is estimated from unannotated text. In an unsupervised setting, Lauer (1995) investigated two methods of approximation:

1. counts from two-word noun compounds, and
2. co-occurrences of a pair of nouns within some fixed window.

The former was shown to outperform the latter.

Subsequent studies have focused on the use of web statistics. We can today obtain huge amounts of text from the web. With this situation, various studies in various fields of natural language processing report performance improvement with the use of web-scale text (Banko and Brill, 2001; Brants et al., 2007; Sasano et al., 2009). In noun compound analysis, Lapata and Keller (2004) and Nakov and Hearst (2005a) used search engine hit counts. Bergsma et al. (2010) and Pitler et al. (2010) utilized Google's N-gram.

Web-based approaches need approximation methods because the web is essentially unannotated text. Lapata and Keller (2004) used a search engine's `NEAR` operator, which might correspond to co-occurrence statistics. Nakov and Hearst (2005a) relied on phrase search. The result can be interpreted as bigrams of successive words. Bergsma et al. (2010) and Pitler et al. (2010) seem to have used bigram counts (plus unigram counts to calculate probabilities).

2.3 Use of arbitrarily sized chunks

The use of arbitrarily sized chunks is relatively new in natural language analysis. Traditionally it has been done by decomposing an input into minimal elements. In probabilistic context-free grammars, for example, the probability of generating a tree is the product of the probabilities of generating each derivation rule. Similarly, a first-order dependency parser defines the score of a dependency tree as the sum of the score of all edges in the tree (McDonald et al., 2005).

The use of arbitrarily sized chunks resulted in a huge success in statistical machine translation (Koehn et al., 2003). Recent studies successfully make use of arbitrarily sized chunks in various tasks of natural language analysis too. Chunks range from sequences (Wood et al., 2011) to tree fragments (Post and Gildea, 2009) and subtrees (Johnson et al., 2007) of phrase-structure grammars. They are usually realized by non-parametric Bayesian models, which provide a way to balance between data fitting and model complexity.

A drawback of non-parametric Bayesian inference is high computational cost that makes it difficult to scale to the web. This is especially the case when Markov chain Monte Carlo sampling is used for inference because it is difficult to parallelize in a theoretically sound way. For this reason, we seek a different kind of statistics that are applicable to a huge amount of web text.

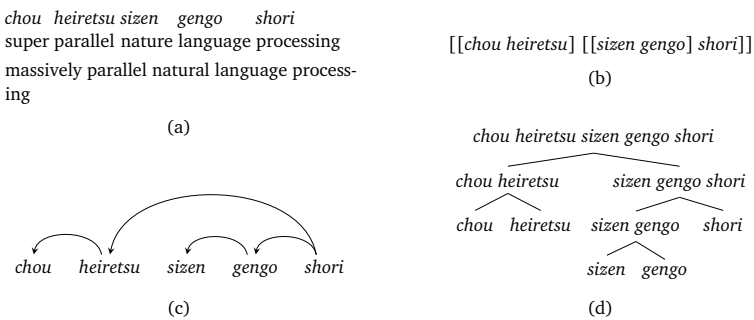


Figure 3: Various representations of a noun compound. (a) Word sequence with word-by-word and full translations. It is taken as the input by our parser. (b) Bracketed representation for its internal structure. (c) Equivalent dependency tree with edges. (d) Span-based binary tree representation.

3 Noun compound analysis

3.1 Task settings

In noun compound analysis, the model takes each noun compound as input and outputs its internal structure. The input is a word sequence as shown in Figure 3a. Since Japanese does not delimit words by white-space, we assume that a word sequence is provided either manually or by some automatic analyzer.

The internal structure of a noun compound can be denoted by brackets (Figure 3b). Bracketing of a noun compound can equivalently be represented as a binary tree. We assume the head-final order for dependency. In other words, a non-final word always modifies a word on its right. With this assumption, we can transform the bracketing structure of a noun compound into an equivalent dependency tree (Figure 3c) and thus noun compound bracketing can be formalized as a parsing problem. Also we can avoid some complex issues that arise from bidirectional parsing (Eisner and Satta, 1999; Johnson, 2007).[1] Alternatively we can use a span-based binary tree representation (Figure 3d). As the output we may think of any representation above.

Formally, our parser is given a word sequence $\boldsymbol{n} = n_1, \cdots, n_L$ as input. Its goal is to output the correct tree $\boldsymbol{t}$. We ignore noun compounds if length $L < 3$ because we assume that their structures are unambiguous. We consider a (semi-)supervised setting. The annotated noun compounds $\mathcal{T} = \{(\boldsymbol{n}_i, \boldsymbol{t}_i)\}_{i=1}^{T}$ are used to train our parser.

3.2 Initial dependency parser

We treat noun compound analysis as a structure prediction problem. Previous studies focused on three word noun compounds that only require a single binary decision per input (Lauer, 1995; Lapata and Keller, 2004; Nakov and Hearst, 2005a; Bergsma et al., 2010), handled longer noun compounds but relied on a series of local decisions (Barker, 1998; Vadas and Curran, 2007b), or used a pseudo-generative model based on a local discriminative classifier (Pitler et al., 2010). By contrast, we directly score entire trees for a given noun compound. We do not

[1] We speculate that our span features, described below, are technically applicable to bidirectional parsing. We leave the application to non-Japanese languages for future work.

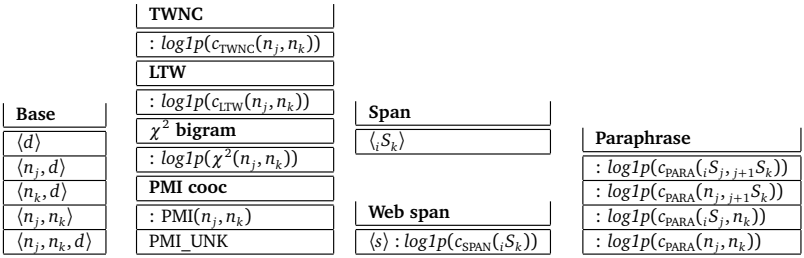


Table 1: Features used by our parser. We consider the combination of spans ${}_iS_j$ and ${}_{j+1}S_k$, and an edge is drawn between n_j and n_k. $\langle x \rangle$ denotes a template that is expanded into multiple features. The left-hand side of the colon is the feature's name. Omitted when obvious. The right-hand side is the feature's value. Binary-valued when omitted. (a) Base edge features. $d = k - j$ is the distance between n_j and n_k (1, 2, 3, 4 or ≥ 5). (b) Web-derived edge features. (c) Span features. $s = k - i + 1$ is the width of the span ${}_iS_k$ (2, 3, 4, 5 or ≥ 6). (d) Paraphrase features.

re-impelemt earlier models but, for comparison, incorporate them as features of our parser.

We begin with a first-order projective dependency model (Eisner, 1996; McDonald et al., 2005), which will be extended later. Specifically we use a high-dimensional linear classifier. The score of a dependency tree is defined as the sum of the score of all edges in the tree,

$$score(\boldsymbol{n}, \boldsymbol{t}) = \sum_{(j,k) \in edges(\boldsymbol{t})} \boldsymbol{w} \cdot \boldsymbol{\phi}(j,k)$$

where $edges(\boldsymbol{t})$ returns all edges in $\boldsymbol{t}$, $\boldsymbol{\phi}(j,k)$ gives a feature vector for the edge between n_j and n_k, and $\boldsymbol{w}$ is the corresponding weight vector that will be learned during training.

Table 1a shows features used by the initial parser, all of which are binary-valued. Unlike full-sentence parsing, noun compound analysis heavily relies on the edge distance d because an overwhelming majority of non-final words modify words to their immediate right. Also d helps the model capture some suffix-like words' tendency to being modified by their left-hand neighbors.

Given $\boldsymbol{w}$ and $\boldsymbol{n}$, we want to find $\boldsymbol{t}$ such that

$$\boldsymbol{t} = \operatorname*{argmax}_{\boldsymbol{t}'} score(\boldsymbol{n}, \boldsymbol{t}').$$

Following (McDonald et al., 2005), we adopt a lexicalized CKY chart parsing algorithm. Just like the one for context-free grammars, our algorithm uses bottom-up dynamic programming. For the word sequence $n_1, \cdots, n_L$, we consider a *span* ${}_iS_j = n_i, \cdots, n_j$ $(i \leq j)$, which holds a score. Our algorithm is simpler than that of McDonald et al. (2005) because we assume the head-final order. Thus whereas bidirectional parsing needs to keep 3 indices for each span, we only need one.

We begin with single-word spans, iteratively combine a pair of spans ${}_iS_j$ and ${}_{j+1}S_k$ to create a larger span ${}_iS_k$, and end up with ${}_1S_L$, the span for the whole word sequence. When ${}_iS_j$ and ${}_{j+1}S_k$ are combined, an edge is drawn between n_j and n_k. Given j, the score of ${}_iS_k$ is the sum of its own edge score and the scores of ${}_iS_j$ and ${}_{j+1}S_k$. To create ${}_iS_k$, we check every possible pair of subspans ${}_iS_j$ and ${}_{j+1}S_k$ by iterating over j and select the one best.

Algorithm 1 Passive-aggressive training (PA-I, prediction-based updates, weight averaging).

Input: training data $\mathcal{T} = \{(\boldsymbol{n}_i, \boldsymbol{t}_i)\}_{i=1}^{T}$

$\boldsymbol{w} = \mathbf{0}$; $\boldsymbol{v} = \mathbf{0}$
for $n = 1..N$ **do**
  shuffle $\mathcal{T}$
  for $(\boldsymbol{n}, \boldsymbol{t}) \in \mathcal{T}$ **do**
    predict $\hat{\boldsymbol{t}} = \text{argmax}_{\boldsymbol{t}}\ score(\boldsymbol{n}, \boldsymbol{t})$
    calculate cost ρ, the number of misidentified edges
    if $\rho > 0$ **then**
      loss $l = score(\boldsymbol{n}, \hat{\boldsymbol{t}}) - score(\boldsymbol{n}, \boldsymbol{t}) + \rho$
      $\tau = \min\{C, l/\|\Phi(\boldsymbol{n}, \boldsymbol{t}) - \Phi(\boldsymbol{n}, \hat{\boldsymbol{t}})\|^2\}$
      $\boldsymbol{w} = \boldsymbol{w} + \tau(\Phi(\boldsymbol{n}, \boldsymbol{t}) - \Phi(\boldsymbol{n}, \hat{\boldsymbol{t}}))$
      $\boldsymbol{v} = \boldsymbol{v} + \boldsymbol{w}$
    end if
  end for
end for
$\boldsymbol{w} = \boldsymbol{v}/(N * T)$

We use an online learning algorithm for training. We implement the online passive-aggressive algorithm (Crammer et al., 2006). Specifically we use the variant named PA-I, with prediction-based updates (Crammer et al., 2006) and weight averaging (Collins, 2002). Algorithm 1 gives the pseudo-code, in which $\Phi(\boldsymbol{n}, \boldsymbol{t}) = \sum_{(j,k) \in edges(\boldsymbol{t})} \boldsymbol{\phi}(j, k)$. We set C as 1.0.

3.3 Noun compound extraction

Next we extend the initial parser with web statistics. While previous studies relied on search engine hit counts (Lapata and Keller, 2004; Nakov and Hearst, 2005a) or n-grams (Bergsma et al., 2010; Pitler et al., 2010), we directly utilize an unannotated text corpus.

In preparation for calculating web statistics, we extract noun compounds from the web corpus. To do this, we first apply the morphological analyzer JUMAN[2] to each sentence to segment it into a word sequence. We then use the dependency parser KNP[3] to identify noun compounds. At a pre-processing step before dependency parsing, KNP chunks a given word sequence into *bunsetsu* phrases. We examine each nominal *bunsetsu* phrase, drop function words that follow content words, and extract sequences of noun and noun-like words. Extracted noun compounds are clean since word segmentation and phrase chunking can be done highly accurately. Note that extracted noun compounds include two-word ones.

3.4 Web-derived edge features

3.4.1 Two-word noun compounds (TWNC and LTW)

Using extracted noun compounds, we introduce web-derived edge features that measure the association between child n_j and head n_k. Following Lauer (1995), we begin with the simplest method of approximation, namely the use of counts from two-word noun compounds. Let $c_{\text{TWNC}}(n_j, n_k)$ be the number of two-word noun compounds that consist of n_j and n_k. We take the log of $c_{\text{TWNC}}(n_j, n_k)$ (to be precise, we use $log1p(x) = log(1 + x)$ to avoid zeros) and

[2] http://nlp.ist.i.kyoto-u.ac.jp/EN/index.php?JUMAN
[3] http://nlp.ist.i.kyoto-u.ac.jp/EN/index.php?KNP

add the result as one additional feature of the parser (**TWNC**, first of Table 1b). As usual, its corresponding weight is tuned using training data.

One advantage of this method is that the result is very clean because we only use a reliable portion of data. However, it has more chance of suffering from the problem of data sparseness than methods that exploit full data. Certain dependency relations might appear only in noun compounds with three or more words.

Alternatively, we focus on the last two words of every noun compound, which in our assumption always have a dependency relation. We take *log1p* of $c_{\text{LTW}}(n_j, n_k)$, the number of such word pairs (**LTW**, second of Table 1b).

3.4.2 Chi-squared bigram measure (χ^2 bigram)

Nakov and Hearst (2005a) used the bigram count $c(n_j, n_k)$, or the number of pages returned by a search engine in response to queries for the exact phrase "$n_j\ n_k$." Instead we collect all bigrams from noun compounds extracted from the web corpus. Either way, bigrams are not clean because they may not represent true dependency relations.

Nakov and Hearst (2005a) empirically showed that the χ^2 dependency measure performed better than other measures. The χ^2 measure is defined as follows:

$$\chi^2(n_j, n_k) = \frac{N(AD - BC)^2}{(A+C)(B+D)(A+B)(C+D)}$$

where $A = c(n_j, n_k)$, $B = c(n_j, \overline{n_k})$, $C = c(\overline{n_j}, n_k)$, $D = c(\overline{n_j}, \overline{n_k})$, and $N = A+B+C+D$ ($c(n_j, \overline{n_k})$ is the number of bigrams in which n_j is followed by a word other than n_k). Zero counts are replaced by 0.5. We take *log1p* of the χ^2 measure and add the result as one additional feature of the parser (χ^2 **bigram**, third of Table 1b).

3.4.3 PMI co-occurrence measure (PMI cooc)

Co-occurrences of n_j and n_k within noun compounds are yet another option although co-occurrences are also rough approximations of dependency relations. Since co-occurrence statistics require much larger space than successive bigrams, we only store pairs of words whose co-occurrence counts are greater than or equal to 10.

We follow Pitler et al. (2010) and use the pointwise mutual information (PMI)[4]

$$\text{PMI}(n_j, n_k) = log \frac{p_{\text{cooc}}(n_j, n_k)}{p_{\text{left}}(n_j) p_{\text{right}}(n_k)},$$

where $p_{\text{cooc}}(n_j, n_k)$ is the probability of the pair n_j, n_k appearing in the same noun compound in this order, $p_{\text{left}}(n_j)$ is the probability of n_j appearing in the left side of co-occurrence pairs, and $p_{\text{right}}(n_k)$ is defined in a similar manner.

We append $\text{PMI}(n_j, n_k)$ as a new feature. Another feature PMI_UNK is fired alternatively if $p_{\text{cooc}}(n_j, n_k) = 0$, $p_{\text{left}}(n_j) = 0$ or $p_{\text{right}}(n_k) = 0$ (**PMI cooc**, last of Table 1b).

[4]We could try any combination of (1) χ^2 and PMI, and (2) bigrams and co-occurrences. We did not investigate this further because experiments showed that simple log-counts performed very well.

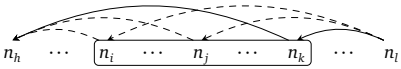


Figure 4: Span as constraints. The box denotes span ${}_iS_k$. The dashed edges are ruled out by the span while the solid ones are still possible.

3.5 Span features

The crux of our parsing model is the use of span features in addition to edge features. To do this, we first rewrite the score function in accordance with the parsing algorithm:

$$score(\boldsymbol{n}, \boldsymbol{t}) = \sum_{(i,j,k) \in spans(\boldsymbol{t})} \boldsymbol{w} \cdot \boldsymbol{\phi}(i, j, k)$$

where $spans(\boldsymbol{t})$ returns all spans with width ≥ 2, which we call non-trivial spans. Non-trivial spans correspond to non-leaf nodes in Figure 3d. The 3-tuple (i, j, k) represents each non-trivial span ${}_iS_k$ that consists of two subspans ${}_iS_j$ and ${}_{j+1}S_k$ ($i \leq j \leq k$). Edge features introduced in Sections 3.2 and 3.4 only use j (child) and k (head).

Now we introduce span features, which add a score to ${}_iS_k$. It is clear that even with this extension, we can still use the CKY algorithm for parsing. For each non-trivial span ${}_iS_k$, we add its span score only after selecting the best pair of subspans ${}_iS_j$ and ${}_{j+1}S_k$ by iterating over j because span features do not depend on j. The training algorithm is the same as before.

The selection of a span constrains multiple edges at once as illustrated in Figure 4. For span ${}_iS_k$, only n_k can interact with the outside of the span: n_k can be modified by n_h ($h < i$), and n_k can modify n_l ($k < l$). However, n_j ($i \leq j < k$) cannot be modified by n_h or cannot modify n_l. This property is useful when, for example, the widely used term "*jidou onsei ninshiki*" (automatic speech recognition) is followed by "*shisutemu*" (system). The selection of a span for the first three words rules out the edge between "*jidou*" (automatic) and "*shisutemu*" (system), and the edge between "*onsei*" (speech) and "*shisutemu*" (system) although both edges are plausible when context is ignored.

We employ two types of span features (**Span** and **Web span** in Table 1c). One is a set of binary features, each of which corresponds to a non-trivial span appearing in the training data. The other is a set of web-derived features, grouped by span width. We take *log1p* of $c_{\text{SPAN}}({}_iS_k)$, the number of times ${}_iS_k$ appears as the noun compound in the web corpus. For $c_{\text{SPAN}}({}_iS_k)$, we do not consider a noun compound nested in a longer noun compound because they cannot be identified confidently. Thus web-derived spans are clean even though we exploit the whole data.

Span features can be seen as a generalization of adjacency employed in early studies (Lauer, 1995). While adjacency is the measure of association between two successive words, span features cover not only two successive words but longer word sequences. As seen above, the parser can easily handle spans with arbitrary width. By contrast, combinations of edges can be handled with dynamic programming only if they are restricted to certain patterns such as consecutive siblings and grandparents (McDonald and Pereira, 2006; Koo and Collins, 2010).

3.6 Paraphrase features

In preliminary experiments, we discovered that many noun compounds took the form of predicate-argument pair. Such a predicate noun compound typically contains a *sahen* noun,

which functions as a verb when followed by light verbs such as *suru* (to do), *dekiru* (can do) and *sareru* (to be done). A nominal predicate-argument pair can be paraphrased by the combination of a noun phrase and a verbal phrase. For example, the noun compound "*sizen gengo shori*" (natural language processing) has corresponding explanatory expressions including

sizen gengo wo ‖ *shori suru*
nature language ACC processing do
to process natural language(s)

(‖ denotes a phrasal boundary). Conversely, this expression suggests the bracket structure "[[*sizen gengo*] *shori*]" ([[natural language] processing]).

We collect pairs of predicate and argument noun compounds from the web corpus and incorporate them as paraphrase features. We follow Kawahara and Kurohashi (2001) for extracting predicate-argument pairs. Although parsing errors are inevitable, we can circumvent this problem by exploiting the constraints of Japanese dependency structures: head-final and projective. The simplest example would be the second-to-last phrase of a sentence, which always depends on the last phrase. With such constraints, we can focus on syntactically unambiguous dependency pairs. For a newspaper corpus, 20.7% of dependency relations are extracted and their accuracy is 98.3% (Kawahara, 2012, p.c.).

For an argument noun phrase, we accept it only if it has the nominative (*ga*) or accusative (*wo*) case marker. For a predicate, we do not distinguish the type of light verbs ("to do," "can do", "to be done" and others). Note that predicate noun compounds may be longer than one word. The following examples are two-word noun compounds that can be used as predicates.

soshiki ka	*shouryou seisan*
organ -ization	little-volume production
organization (the act of organizing)	little-volume production

Paraphrase features conform with the three-argument feature function $\boldsymbol{\phi}(i, j, k)$ and can be used straightforwardly in dynamic programming. We employ four features as shown in Table 1d. The first feature takes *log1p* of $c_{\text{PARA}}({}_iS_j, {}_{j+1}S_k)$, the number of times span ${}_iS_j$ is used as an argument of a verbal phrase derived from span ${}_{j+1}S_k$. The second feature is based on $c_{\text{PARA}}(n_j, {}_{j+1}S_k)$, which resembles that of the first feature but uses the head word n_j instead of the span ${}_iS_j$. The third and fourth features are defined in similar manners. These features are mixtures of edges and spans.

Nakov and Hearst (2005a) incorporated hyphen, concatenation and other paraphrase-based cues into noun compound bracketing. For example, *cell-cycle* and *healthcare* reinforce the bracket structures "[[*cell cycle*] *analysis*]" and "[[*health care*] *reform*]" respectively. They have no Japanese counterpart, however. Other tasks of linguistic analysis in which simple paraphrase features are used include word segmentation (Kaji and Kitsuregawa, 2011) and PP attachment (Nakov and Hearst, 2005b; Bansal and Klein, 2011).

4 Experiments

4.1 Data

In-domain data We first built annotated data for both training and testing. We used the NTCIR1 TMREC test collection (Kageura et al., 1999).[5] It consisted of 1,870 Japanese paper

[5] We chose this test collection simply because it can also be used in future research on applications of noun compound analysis.

abstracts in the field of computer science. Gold-standard segmentation and POS tagging were provided.[6]

Following Nakagawa and Mori (2002), we extracted uninterrupted noun sequences as noun compounds. We discarded chunking errors, and single-word and two-word noun compounds. We randomly selected 3,100 noun compounds and manually annotated them with dependency relations.

Out-of-domain data We constructed two sets of annotated noun compounds for testing. We used J-STAGE,[7] an online collection of electronic journals. We collected Japanese papers in the fields of agriculture (**Out-of-domain 1**) and material science (**Out-of-domain 2**).

We extracted noun compounds through the procedure described in Section 3.3. We discarded segmentation and chunking errors, and two-word noun compounds. We randomly selected 1,000 noun compounds for each set and manually annotated them with dependency relations.

Web corpus The web corpus from which web statistics was calculated was compiled through procedures proposed by Kawahara and Kurohashi (2006). It consisted of about 70 million Japanese web pages.

4.2 Models

We trained and tested the parser with various combinations of features. For each model, we run 20 iterations for online learning. We conducted 5-fold cross-validation on the in-domain data. For out-of-domain data, we trained each model on the whole in-domain data.

For comparison, we also examined three baseline methods.
Left-branching Every non-final word modifies its immediate right neighbor.
Right-branching Every non-final word modifies the final word.
Random Choose a dependency tree at random.

4.3 Evaluation measures

We measured the performance of the parser with unlabeled attachment score (UAS). UAS is defined as the proportion of correctly identified dependency relations. Note that we did not exclude the second last word, which in our assumption always modified the last word. We allowed annotators to break the assumption although we found none. We used McNemar's test of significance to evaluate the degree of difference between a pair of model outputs.

4.4 Results

Table 2 shows unlabeled attachment scores. The left-branching baseline was strong because an overwhelming majority of non-final words modified their immediate right neighbors. The discriminative parser managed to beat the left-branching baseline even with the **Base** features alone.

Not surprisingly, porting to out-of-domain data resulted in drops in accuracy. These disparities can be explained by the fact that while for in-domain data, 64.6% of edges (word pairs, regardless of distance) in test data were observed at least once in training data, the number dropped drastically to 5.0% and 5.4% for out-of-domain data.

[6] Segmentations were sometimes inconsistent with those of the morphological analyzer JUMAN, which was used for building web statistics. This might have a slightly unfavorable impact on performance.

[7] `https://www.jstage.jst.go.jp/browse/`

Model	In-domain (7,717 edges)	Out-of-domain 1 (2,370 edges)	Out-of-domain 2 (2,389 edges)
Left-branching	88.32	86.20	86.40
Right-branching	49.03	53.54	52.66
Random	68.47	69.11	69.61
Base	94.27	88.44	88.32
+ TWNC	94.32	90.17**	92.42**
+ LTW	94.27	88.31	89.74**
+ χ^2 bigram	94.13	87.43	88.70
+ PMI cooc	94.30	87.93	88.91
+ Span	94.40	88.23	88.87*
+ Web span	94.35	88.86	90.41**
+ Span + Web span	94.43	88.86	90.83**
+ Paraphrase	94.08	88.06	88.74
+ Span + Web span + TWNC	94.54	90.13**	92.21**
+ Span + Web span + Paraphrase	94.46	89.49*	91.53**
+ Span + Web span + TWNC + Paraphrase	**94.64***	**90.30****	**93.01****

Table 2: Unlabeled attachment scores. * and ** mark statistically significant improvement over the **Base** model with $p < 0.05$ and $p < 0.01$ respectively.

Among the four types of web-derived edge features, the simplest **TWNC** feature performed best, consistently improving accuracy. The gains obtained for out-of-domain data were remarkable. Somewhat unexpectedly, the **LTW** feature performed much worse than **TWNC**. The χ^2 **bigram** and **PMI cooc** features were consistently beaten by **TWNC**.

For in-domain data, the **Span** features alone resulted in a performance gain slightly larger than **TWNC**. However, they seemed too domain-specific as they did not work well for out-of-domain data. By contrast, the **Web span** features brought consistent gains to both in- and out-of-domain data. Adding only the **Paraphrase** features to the **Base** model had a negative impact for in-domain data and out-of-domain 1.

The results indicate the complementary nature of the edge, span and paraphrase features. The combination of these features generally boosted performance. This was especially true for out-of-domain data. 0.63% and 0.70% gains were obtained when the **Paraphrase** features were added to the **Base** + **Span** + **Web span** model even though **Paraphrase** alone did not work well. **TWNC** alone worked well for out-of-domain data, but further gain was obtained when the **Span**, **Web span** and **Paraphrase** features were added. The highest scores were achieved by the **Base** + **Span** + **Web span** + **TWNC** + **Paraphrase** model (hereafter, the full model) for all datasets.

4.5 Discussion

We found that web span features were useful for complementing weak edges. In "[*fonon* [*jiyuu enerugī*]]" ([phonon [free energy]]), for example, the well-known term "*jiyuu enerugī*" (free energy) is modified ad hoc by "*fonon*" (phonon). The edge between "*fonon*" (phonon) and "*enerugī*" was so weak that in the **Base** model it was unable to override the strong preference for short-distance dependency. On the other hand, the web span feature strongly supported "*jiyuu enerugī*" (free energy). A powerful span means that non-head words within the span must not be modified from outside the span. For this reason, the edge between "*fonon*" (phonon) and "*jiyuu*" (free) was ruled out.

However, the span features sometimes had an adverse impact on parsing. Consider the following example.

[[*shinka gēmu*] *riron*]
evolution game theory
evolutionary game theory.

The full model wrongly output "[*shinka* [*gēmu riron*]]" ([evolutionary [game theory]]). This noun compound can be interpreted as a fusion of "*shinka gēmu*" (evolutionary game) and more prominent "*gēmu riron*" (game theory). In other words, the non-head word (game) of the latter is modified by the grafting of the former. However, our span features do not allow such an operation.

This inherent weakness of span might also explain the poor performance of paraphrase features. We investigated the weight vector of the full model. Somewhat surprisingly, we found that among four paraphrase features, only $c_{\text{PARA}}(n_j, {}_{j+1}S_k)$ (the argument's head word and the predicate's span) had a positive weight. In other words, the argument span ${}_iS_j$ was considered useless or even harmful by the model. One possible reason is the productivity of the argument. ${}_iS_j$ in paraphrase features imposes the condition that n_j must not be modified from outside the span. However, such a modification occurs very often. Even if the correct pair ${}_iS_j, {}_{j+1}S_k$ is covered by the web corpus, it is often blocked by another pair ${}_{i'}S_j, {}_{j+1}S_k$ $(i < i')$, which usually has larger counts.

While we use flat spans, the same is true of genuine tree models. Our span features show some similarity to adaptor grammars (Johnson et al., 2007), a generalization of probabilistic context-free grammars.[8] An adaptor grammar directly considers a distribution of subtrees rooted by a common non-terminal instead of decomposing them into derivation rules. A subtree is completely expanded into terminals. If subtrees are collapsed into terminal sequences, they become spans.

The adaptor grammar does not allow subtrees to be modified partially. This is unfavorable in general because "*parallel processing*" is productively modified by an adverb and transformed into "*massively parallel processing*" for example. To address this problem, we need to handle incompletely expanded trees that are to be completed by a *substitution* operator and/or we need to introduce an *insertion* operator (Shindo et al., 2011). For the semi-supervised setting of noun compound analysis, we may need collapsed versions of these operations.

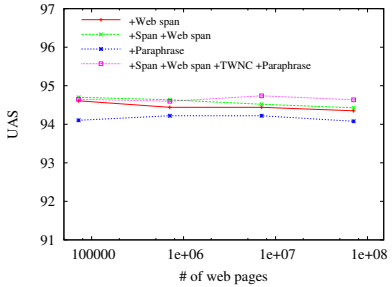
4.6 Effect of corpus size

Finally, we investigated the effect of the size of the corpus from which we calculated web statistics. We reduced the number of web pages to 1/10, 1/100 and 1/1000. The models were trained and tested as before.

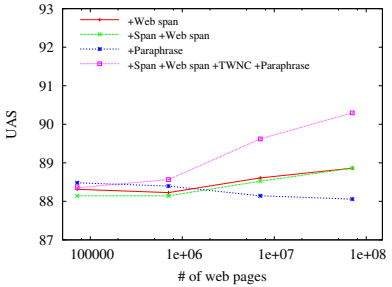
Figures 5(a)-(c) show UASs in relation to corpus size. In-domain data did not receive benefit from the increase of unannotated text. It seems that the Base and Span features, which were learned directly from annotated data, were too informative for the other features to work with.

For out-of-domain data, accuracy largely consistently improved with the corpus size except for the **Base** + **Paraphrase** model. The graphs indicate that further gain can be achieved by

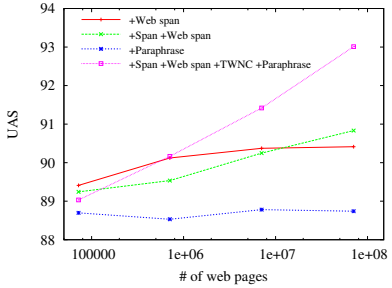
[8] Adaptor grammars are not irrelevant to dependency parsing as dependency grammars can be transformed into context-free grammars (Johnson, 2007). The original adaptor grammars do not allow self-recursion, which is integral to dependency-derived CFGs, but this restriction was overcome by a variational inference scheme (Cohen et al., 2010).



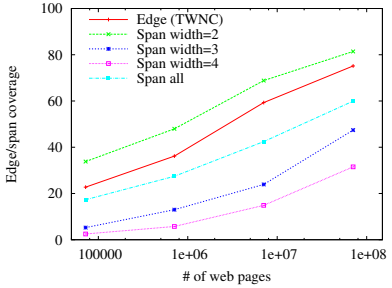
(a) In-domain.



(b) Out-of-domain 1.



(c) Out-of-domain 2.



(d) Out-of-domain 1.

Figure 5: Effect of corpus size. (a)-(c) Unlabeled attachment scores in relation to corpus size. (d) Edge/span coverage in relation to corpus size.

simply enlarging the corpus. This is supported by Figure 5d, which depicts how many edges and spans in test data appear at least once in the web statistics. There is much room for improving coverage.

Conclusion

In this paper, we proposed a semi-supervised method for noun compound analysis that combined span features with edge features. Experiments show that span features improve accuracy and that further gain is obtained when they are combined with edge features.

Words within noun compounds are arranged in a rather fixed order, and adding a word in between appears to impair semantic coherence. The very fact that people often choose bracketing to denote the internal structure of a noun compound may be an intuitive justification of the use of spans since the bracket structure obscures dependency but shows spans more clearly.

Acknowledgment

This work was partly supported by JST CREST.

References

Banko, M. and Brill, E. (2001). Mitigating the paucity-of-data problem: exploring the effect of training corpus size on classifier performance for natural language processing. In *Proceedings of the First International Conference on Human Language Technology Research (HLT 2001)*, pages 1–5.

Bansal, M. and Klein, D. (2011). Web-scale features for full-scale parsing. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 693–702.

Barker, K. (1998). A trainable bracketer for noun modifiers. In *Proceedings of the 12th Biennial Conference of the Canadian Society for Computational Studies of Intelligence on Advances in Artificial Intelligence*, pages 196–210.

Bergsma, S., Pitler, E., and Lin, D. (2010). Creating robust supervised classifiers via web-scale N-gram data. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 865–874.

Brants, T., Popat, A. C., Xu, P., Och, F. J., and Dean, J. (2007). Large language models in machine translation. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL 2007)*, pages 858–867.

Cohen, S. B., Blei, D. M., and Smith, N. A. (2010). Variational inference for adaptor grammars. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 564–572.

Collins, M. (2002). Discriminative training methods for hidden Markov models: Theory and experiments with perceptron algorithms. In *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing*, pages 1–8.

Crammer, K., Dekel, O., Keshet, J., Shalev-Shwartz, S., and Singer, Y. (2006). Online passive-aggressive algorithms. *Journal of Machine Learning Research*, 7:551–585.

Eisner, J. (1996). The new probabilistic models for dependency parsing: An exploration. In *COLING 1996 Volume 1: The 16th International Conference on Computational Linguistics*, pages 340–345.

Eisner, J. and Satta, G. (1999). Efficient parsing for bilexical context-free grammars and head-automaton grammars. In *Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 457–464.

Flannery, D., Miyao, Y., Neubig, G., and Mori, S. (2011). Training dependency parsers from partially annotated corpora. In *Proceedings of 5th International Joint Conference on Natural Language Processing*, pages 776–784.

Johnson, M. (2007). Transforming projective bilexical dependency grammars into efficiently-parsable CFGs with unfold-fold. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 168–175.

Johnson, M., Griffiths, T. L., and Goldwater, S. (2007). Adaptor grammars: A framework for specifying compositional nonparametric Bayesian models. In *Advances in Neural Information Processing Systems*, volume 19, pages 641–648.

Kageura, K., Yoshioka, M., Takeuchi, K., Koyama, T., Tsuji, K., Yoshikane, F., and Okada, M. (1999). Overview of TMREC tasks. In *Proceedings of the First NTCIR Workshop on Research in Japanese Text Retrieval and Term Recognition*, page 415.

Kaji, N. and Kitsuregawa, M. (2011). Splitting noun compounds via monolingual and bilingual paraphrasing: A study on Japanese Katakana words. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 959–969.

Kawahara, D. and Kurohashi, S. (2001). Japanese case frame construction by coupling the verb and its closest case component. In *Proceedings of the First International Conference on Human Language Technology Research (HLT 2001)*, pages 204–210.

Kawahara, D. and Kurohashi, S. (2006). Case frame compilation from the web using high-performance computing. In *Proceedings of The 5th International Conference on Language Resources and Evaluation (LREC-06)*, pages 1344–1347.

Koehn, P., Och, F. J., and Marcu, D. (2003). Statistical phrase-based translation. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology - Volume 1*, pages 48–54.

Koo, T. and Collins, M. (2010). Efficient third-order dependency parsers. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 1–11.

Kudo, T. and Matsumoto, Y. (2002). Japanese dependency analysis using cascaded chunking. In *Proceedings of the 6th Conference on Natural Language Learning*, pages 1–7.

Lapata, M. and Keller, F. (2004). The web as a baseline: Evaluating the performance of unsupervised web-based models for a range of NLP tasks. In *HLT-NAACL 2004: Main Proceedings*, pages 121–128.

Lauer, M. (1995). Corpus statistics meet the noun compound: Some empirical results. In *Proceedings of the 33rd Annual Meeting of the Association for Computational Linguistics*, pages 47–54.

Liberman, M. and Sproat, R. (1992). The stress and structure of modified noun phrases in English. In Sag, I. A. and Szabolcsi, A., editors, *Lexical Matters*, pages 131–181. Center for the Study of Language.

Marcus, M. P. (1980). *Theory of Syntactic Recognition for Natural Languages*. MIT Press.

Marcus, M. P., Marcinkiewicz, M. A., and Santorini, B. (1993). Building a large annotated corpus of English: the Penn Treebank. *Computational Linguistics*, 19(2):313–330.

McDonald, R., Crammer, K., and Pereira, F. (2005). Online large-margin training of dependency parsers. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 91–98.

McDonald, R. and Pereira, F. (2006). Online learning of approximate dependency parsing algorithms. In *11th Conference of the European Chapter of the Association for Computational Linguistics*, pages 81–88.

Nakagawa, H. and Mori, T. (2002). A simple but powerful automatic term extraction method. In *COLING-02 on COMPUTERM 2002: Second International Workshop on Computational Terminology - Volume 14*, pages 29–35.

Nakov, P. and Hearst, M. (2005a). Search engine statistics beyond the n-gram: Application to noun compound bracketing. In *Proceedings of the Ninth Conference on Computational Natural Language Learning (CoNLL-2005)*, pages 17–24.

Nakov, P. and Hearst, M. (2005b). Using the web as an implicit training set: Application to structural ambiguity resolution. In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*, pages 835–842.

Pitler, E., Bergsma, S., Lin, D., and Church, K. (2010). Using web-scale N-grams to improve base NP parsing performance. In *Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010)*, pages 886–894.

Post, M. and Gildea, D. (2009). Bayesian learning of a tree substitution grammar. In *Proceedings of the ACL-IJCNLP 2009 Conference Short Papers*, pages 45–48.

Pustejovsky, J., Bergler, S., and Anick, P. (1993). Lexical semantic techniques for corpus analysis. *Computational Linguistics*, 19(2):331–358.

Resnik, P. S. (1993). *Selection and Information: A Class-Based Approach to Lexical Relationships*. PhD thesis, University of Pennsylvania.

Sasano, R., Kawahara, D., and Kurohashi, S. (2009). The effect of corpus size on case frame acquisition for discourse analysis. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 521–529.

Shindo, H., Fujino, A., and Nagata, M. (2011). Insertion operator for bayesian tree substitution grammars. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 206–211. Association for Computational Linguistics.

Uchimoto, K., Sekine, S., and Isahara, H. (1999). Japanese dependency structure analysis based on maximum entropy models. In *Proceedings of the Ninth Conference of the European Chapter of the Association for Computational Linguistics*, pages 196–203.

Vadas, D. and Curran, J. (2007a). Adding noun phrase structure to the Penn Treebank. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 240–247.

Vadas, D. and Curran, J. (2007b). Large-scale supervised models for noun phrase bracketing. In *Proceedings of the Conference of the Pacific Association for Computational Linguistics (PACLING)*, pages 104–112.

Wood, F., Gasthaus, J., Archambeau, C., James, L., and Teh, Y. W. (2011). The sequence memoizer. *Communications of the Association for Computing Machines*, 54(2):91–98.

Learning Effective and Interpretable Semantic Models using Non-Negative Sparse Embedding

Brian Murphy Partha Pratim Talukdar Tom Mitchell
Machine Learning Department
Carnegie Mellon University
{bmurphy,ppt,tom}@cs.cmu.edu

Abstract

In this paper, we introduce an application of matrix factorization to produce corpus-derived, distributional models of semantics that demonstrate cognitive plausibility. We find that word representations learned by Non-Negative Sparse Embedding (NNSE), a variant of matrix factorization, are sparse, effective, and highly interpretable. To the best of our knowledge, this is the first approach which yields semantic representation of words satisfying these three desirable properties. Though extensive experimental evaluations on multiple real-world tasks and datasets, we demonstrate the superiority of semantic models learned by NNSE over other state-of-the-art baselines.

Keywords: distributional semantics, sparse coding, neuro-semantics, vector-space models, interpretability, word embeddings.

Proceedings of COLING 2012: Technical Papers, pages 1933–1950,
COLING 2012, Mumbai, December 2012.

1 Introduction

State-of-the-art distributional models of semantics, also termed vector-space models or word embeddings, derive word-representations in an unsupervised fashion from large corpora. They are primarily based on observed co-occurrence patterns, but are typically subsequently reduced in dimensionality using techniques such as Clustering, Latent Dirichlet Allocation (LDA) (Blei et al., 2003), or Singular Value Decomposition (SVD). They have proven effective as components of a wide range of NLP applications, and in the modelling of cognitive operations such as judgements of word similarity (Sahlgren, 2006; Turney and Pantel, 2010; Baroni and Lenci, 2010), and the brain activity elicited by particular concepts (Mitchell et al., 2008). However, with few exceptions (e.g. Baroni et al., 2010), the *representations* they derive from corpora are lacking in cognitive plausibility.

For instance, one of the SVD-based models described in this paper models similarity very successfully, revealing the set of words *mango, plum, cranberry, blueberry, melon* as the cosine-distance nearest neighbours of *pear*. However, the latent SVD dimension for which *pear* has its largest weighting is hard to interpret – its most strongly positively associated tokens are *action, records, government, record, search*, and negatively associated tokens are *sound, need, s, species, award*. In addition, the representation of *pear* is a dense vector of small positive or negative values on several hundred dimensions which are also active for words of all types and domains, whether semantically similar (e.g., other living things, and concrete objects) or dissimilar (e.g., abstract nominals, and function and content words of other parts-of-speech).

Certain cognitive arguments against such a representation are based on economy of storage (see Schunn, 1999; Murphy, 2004; Griffiths et al., 2007 for broader discussion on this point). A-priori it seems unlikely that the same compact set of features are sufficient and necessary to describe all semantic domains of a full adult vocabulary. Some very specific words may need more detailed descriptions (i.e., more properties) and generic words less. Properties are restricted to particular classes of concepts – limbs to animals, functions to manipulable artefacts, wheels to vehicles – and even within restricted semantic domains some features can be very specific to certain concepts, such as screens to electronic devices, antennae to insects and stripes to certain kinds of animal. It would also be uneconomical for people to store all negative properties of a concept, such as the fact that dogs do not have wheels, or that airplanes are not used for communication. And indeed in feature norming exercises (Garrard et al., 2001; McRae et al., 2005; Vinson and Vigliocco, 2008) where participants are asked to list the properties of a word, the aggregate descriptions are typically limited to approximately 10-20 characteristics for a given concrete concept, with limited overlap in features across semantic domain. So, for cognitive plausibility, we claim that a feature set should have three characteristics: it should only store positive facts; it should have a wide range of feature types, to cover all semantic domains in the typical mental lexicon; and only a small number of these should be active to describe each word/concept.

In the context of distributional models of semantics, these characteristics correspond to a non-negative and sparse embedding of lexical meaning. Of course, a raw co-occurrence matrix (whether based on local structural features, or document-region features) or certain derived measures such as positive pointwise mutual information (PPMI) will also be sparse and non-negative, but they lack the discovery of effective synonymy in the latent features of a dimensionality-reduced matrix. As a result, in this paper, we propose to apply sparse matrix factorization, with a constraint of non-negativity on the word-latent matrix.

In this paper, we introduce a new application of matrix factorization to produce a corpus-derived

distributional model of semantics with these desirable characteristics. We find that such word representations are effective in tasks of interest, are sparse, and are also interpretable, when learned by Non-Negative Sparse Embedding (NNSE) – a variation on Non-Negative Sparse Coding, which is a matrix factorization technique previously studied in the machine learning community (Hoyer, 2002; Mairal et al., 2010). To the best of our knowledge, this is the first approach whose output embeddings satisfy all these three desirable properties. We first compare its performance against a wide range of word-embedding methods (including LDA, SVD-based models, and Collobert & Weston's 2008 method) using a suite of behavioural benchmarks that are based on human judgements of lexical similarity. On these tasks the SVD and NNSE models have a decisive advantage, and we follow with an exploration of optimal dimensionality, also on a neurosemantics task (Mitchell et al., 2008). While both SVD and NNSE models have similar peak performance, their dependence on dimensionality is opposed: small numbers of features are optimal for SVD models, and large numbers of feature dimensions are optimal for NNSE models.

We then go on to explore the reasons for these differences, but examining the degree of sparsity seen in the SVD and NNSE models considered, and how easily each dimensional feature can be interpreted, using crowd-sourced judgements (Boyd-Graber et al., 2009). Here, the advantages of the NNSE models are very clear.

The paper is organized as follows: in Section 2, we breifly review the SVD and NNSE matrix factorization methods. Section 3 describes the main experiments, which evaluate various word embedding models in terms of felicity on human conceptual tasks, their degree of sparsity, and their ease of interpretability. In Section 4 we review related work, and then in Section 5, we conclude with a discussion of this new model's relation to other such models, and add some suggestions for further work.

2 Methods

Let $X = [\mathbf{x}_1, \mathbf{x}_2, \ldots \mathbf{x}_n] \in \mathbb{R}^{m \times n}$ be the input representation, where m is the number of words and n is the number of input dimensions (derived from corpus co-occurrence patterns). Each element $X_{i,j}$ represents the value of the j^{th} dimension for the i^{th} word. We are interested in embedding the m words in a k dimensional space, where $k < n$ usually. In other words, we are interested in estimating matrix $A \in \mathbb{R}^{m \times k}$, where the i^{th} row, $A_{i,:}$, represents the new embedding for the i^{th} word.

Although we compare many different word embedding techniques in our experiments, we describe below the two techniques, Singular Value Decomposition (SVD) and Non-Negative Sparse Embedding (NNSE), which we found to be most effective in our evaluation tasks (see, Section 3). In addition to task performance, we find that the embeddings learned by NNSE are also sparse and interpretable, two critical properties which are missing from the embeddings learned by SVD.

2.1 Singular Value Decomposition (SVD)

Singular Value Decomposition (SVD) is a matrix factorization technique which identifies the dimensions within each model with the greatest explanatory power, and which also has the effect of combining similar dimensions (such as synonyms, inflectional variants, topically similar documents) into common components, and discarding more noisy dimensions in the data. Given our input matrix $X \in \mathbb{R}^{m \times n}$, SVD performs the following decomposition:

$$X = USV^T \tag{1}$$

where $q = \min(m, n)$, $U \in \mathbb{R}^{m \times q}$, S is a $q \times q$ a diagonal matrix with sorted (largest to smallest) singular values on its diagonal, and $V \in \mathbb{R}^{n \times s}$. From this decomposition, we get our required embedding A as follows:

$$\begin{aligned} r &= \min(q, k) \\ A &= U(:, [1:r]) \end{aligned}$$

Please note that when X is centered, SVD and Principal Components Analysis (PCA) yield the same embedding. Latent Semantic Analysis (LSA) (Deerwester et al., 1990), a very popular technique used in Information Retrieval, performs SVD on a word by document matrix. We note that in our case, the matrix X is not restricted to be a word-document matrix.

In order to perform SVD on large and sparse X, we use the Implicitly Restarted Arnoldi method (Lehoucq et al., 1998; Jones et al., 2001). We refer the interested reader to these references for more details on the algorithm.

2.2 Non-Negative Sparse Embedding (NNSE)

In this section, we describe the Non-Negative Sparse Embedding (NNSE) method, a variation on Non-Negative Sparse Coding (NNSC), which is a matrix factorization technique previously studied in the machine learning community (Hoyer, 2002; Mairal et al., 2010). Given our input matrix X, NNSE returns a sparse embedding for the words in X (each word's input representation corresponds to a row in X). NNSE achieves this by solving the following optimization problem:

$$\begin{aligned} \arg \min_{A \in \mathbb{R}^{m \times k}, D \in \mathbb{R}^{k \times n}} \quad & \sum_{i=1}^{m} \left(||X_{i,:} - A_{i,:} \times D||^2 + \lambda ||A_{i,:}||_1 \right) \\ \text{where,} \quad & D_{i,:} D_{i,:}^T \leq 1, \ \forall 1 \leq i \leq k \\ & A_{i,j} \geq 0, \ \forall 1 \leq i \leq m, \ \forall 1 \leq j \leq k \end{aligned} \tag{2}$$

where $D \in \mathbb{R}^{k \times n}$ is a dictionary with k entries, $\lambda \in R$ is a hyperparameter, and $||A_{i,:}||_1 = \sum_{j=1}^{k} |A_{i,j}|$ is the l_1 regularization of the i^{th} row of A. The goal here is to decompose the input matrix X into two matrices, A and D, subject to the constraints that the length of the dictionary entries (rows of D) are upper bounded by 1, the rows of A are sparse, and that entries in A are all non-negative (i.e., $A_{i,j} \geq 0$). Unlike NNSC, NNSE does not impose a constraint of non-negativity on D.

Alternatively, this can also be thought of as a mixture model, with the A providing mixing proportion over the dictionary entries in D. This problem is also known as *basis pursuit* (Chen et al., 2001). The NNSE formulation can also be equivalently posed as a matrix factorization problem, and in this respect, both SVD and NNSE are different types of matrix decomposition algorithms.

We note that the NNSE objective is not convex with respect to the coefficients A and dictionary D. However, it is convex with respect to each variable when the others are kept fixed. We use the online algorithm in Mairal et al. (2010) to solve the NNSE optimization problem shown above. We refer the reader to Section 3 of Mairal et al. (2010) for details on the algorithm. The online algorithm is guaranteed to convergence, and it returns the non-negative sparse embedding matrix A (along with the dictionary D), whose i^{th} rows is the new embedding for the for the word whose input representation is given by the i^{th} row of matrix X. We note that overcomplete decomposition, i.e., $k > n$, is possible in case of NNSE. For all experiments in this paper, we set $\lambda = 0.05$ and implement NNSE using the SPAMS package[1].

[1] SPAMS Package: http://spams-devel.gforge.inria.fr/

3 Experiments

In this section we try to answer two main questions: what broad categories of word embedding methods are most effective in modelling cognition; and which of the well-performing models are more cognitively plausible. Several of the models evaluated were already available, and were adopted from Ratinov et al. (2010) (Collobert & Weston, HLBL), and from Řehůřek and Sojka (2010) (LDA topic model based on the English Wikipedia).

The SVD and sparse non-negative models on which we concentrate, were constructed from scratch, based on both LSA-style word-document co-occurrence counts (i.e., word-region features) and HAL-style word-dependency co-occurrence counts (i.e., word-collocate features). Counts were computed from a large English web-corpus, Clueweb (Callan and Hoy, 2009), over a fixed 40,000 word vocabulary. These were the most frequent word-forms found in the American National Corpus (Nancy Ide and Keith Suderman, 2006) summing to 97% of its token count, and should approximate the scale and composition of the vocabulary of a university-educated speaker of English (Nation and Waring, 1997).

The dependency counts were taken from a 16 billion word portion of Clueweb, extracted with the Malt parser, which achieves accuracies of 85% when deriving labelled dependencies on English text (Hall et al., 2007). The features extracted are pairs of dependency relation and lexeme, corresponding to each edge linked to a target word of interest (e.g., the word *coach* might have the dependency features *successful_adj, fires_obj, hires_subj*). This kind of model (Lin, 1998; Padó and Lapata, 2007; Baroni and Lenci, 2010) can be viewed as a more linguistically informed variant of flat window models (e.g., HAL, Lund et al., 1995) or directional models (Schütze and Pedersen, 1993; Bullinaria and Levy, 2007; Murphy et al., 2012). As is common with such models, a co-occurrence frequency cut-off (of 20 – see Bullinaria and Levy, 2007 for a systematic evaluation of this parameter) was used to reduce the dimensionality of the frequency matrix, and to discard noisy counts.

The document co-occurrence counts were taken from 10 million documents of Clueweb. The document model can be viewed as a variant of LSA (Deerwester et al., 1990; Landauer and Dumais, 1997), with differences in the frequency normalisation procedure. A frequency cut-off of 2 was used, so all counts of 1 were discarded (Bradford, 2008).

All models we constructed used positive pointwise-mutual-information (3,4) as an association measure to normalize the observed co-occurrence frequency $p(w,f)$ for the varying frequency of the target word $p(w)$ and its features $p(f)$. PPMI up-weights co-occurrences between rare words, yielding positive values for collocations that are more common than would be expected by chance, and discards negative values that represent patterns of co-occurrences that are *rarer* than one would expect by chance (i.e., if word distributions were independent). It has been shown to perform well generally, with both word- and document-level statistics, in raw and dimensionality reduced forms (Bullinaria and Levy, 2007; Turney and Pantel, 2010). The previous frequency cut-off is also important here, as PMI is positively biased towards hapax co-occurrences.

$$\mathrm{PPMI}_{wf} = \begin{cases} \mathrm{PMI}_{wf} & \text{if } \mathrm{PMI}_{wf} > 0 \\ 0 & \text{otherwise} \end{cases} \tag{3}$$

$$\mathrm{PMI}_{wf} = log\left(\frac{p(w,f)}{p(w)p(f)}\right) \tag{4}$$

The SVD matrix factorisation was first computed separately on the dependency co-occurrence matrix, and on the document co-occurrence matrix, with an output for each of 1000 latent dimensions. For

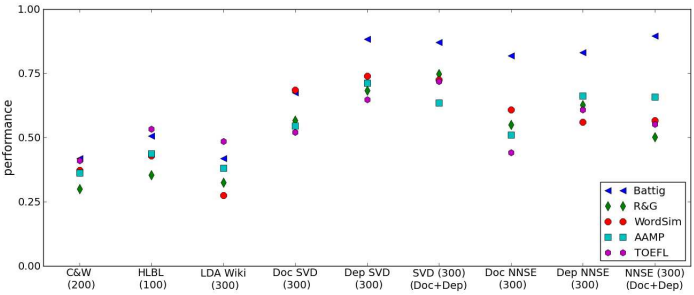


Figure 1: Behavioral evaluation of range of model types, with number of dimensions in parentheses (see Section 3.1.1 for more details).

this step we used Python/Scipy implementation of the Implicitly Restarted Arnoldi method (Lehoucq et al., 1998; Jones et al., 2001) which was coherent with the PPMI normalization used, since a zero value represented both negative target-feature associations, and those that were not observed or fell below the frequency cut-off. The combined SVD model was computed using conventional SVD on the 2000-dimensional concatenation of the output of the previous step – that is the two left singular vectors from the dependency and document models respectively.

While in principle it would be possible to produce the NNSE models directly on the co-occurrence data, here we chose to do this by using the SVD left-singular vectors as a stand-in for the full data. In other words, this is our input representation matrix $X \in \mathbb{R}^{m \times n}$ with $m = 35560$ words and $n = 2000$ input dimensions. Using these lower-noise approximate intermediate matrices reduces the computational task dramatically.

In the rest of this section, we evaluate the following. How effective are the different representations, and different dimensionalities, in various extrinsic evaluation tasks (Section 3.1)? How sparse are the SVD and NNSE representations (Section 3.2)? And how interpretable are these representations (Section 3.3)? All the NNSE representations used in the experiments in this section will be available at `http://www.cs.cmu.edu/~bmurphy/NNSE/`.

3.1 Evaluating Model Performance

3.1.1 Behavioral Experiments

The cognitive plausibility of computational models of word meaning has typically been tested using behavioural benchmarks, such as emulating elicited judgements of pairwise similarity or of category membership (Lund and Burgess, 1996; Rapp, 2003; Sahlgren, 2006).

Here we used five such tests. The two categorization tests were the Battig (Battig and Montague, 1969) test-set consisting of 82 nouns, each assigned to one of 10 concrete classes; and the AAMP (Almuhareb and Poesio, 2004) test-set containing 402 nouns in a range of 21 concrete and abstract classes from WordNet (Miller et al., 1990). Both these tests were performed with the Cluto clustering package (Karypis, 2003) and cosine distances, and success was measured as percentage purity over clusters based on their plurality class. Two sets of similarity judgements were used: the Rubenstein

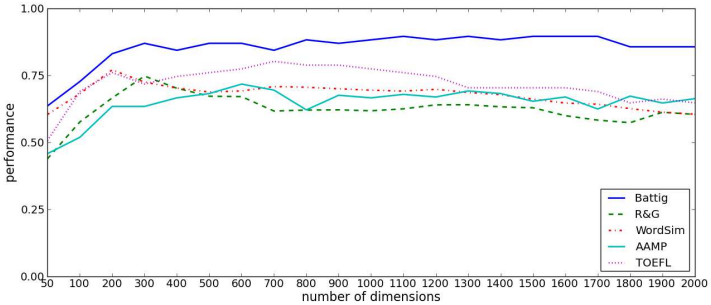


Figure 2: Behavioral evaluation of SVD models with range of dimensionality (see Section 3.1.1 for more details).

and Goodenough (1965) set of 65 concrete word pairs, and the strict-similarity subset of 203 pairs (Agirre et al., 2009) selected from the WordSim353 test-set (Finkelstein et al., 2002). Performance was evaluated with the Spearman correlation between the aggregate human judgements and pairwise cosine distances between word vectors in the model in question. Finally the TOEFL benchmark (Landauer and Dumais, 1997) consists of aggregate records from an examination task for learners of English, who have to identify a synonym among a set of distractors. Performance was measured as the percentage correct over 80 questions, if the cosine-distance of the target-synonym pair was smaller than the distance between the target and any of the distractor words.

For the NNSE and SVD models constructed here, an initial dimension setting of 300 was chosen, as this is in the middle range of those typically used in the literature. Where there was a choice with the other models we used the largest dimensionality available. For each of these two dimensionality reduction techniques, one model was constructed on the basis of document co-occurrences only (labelled "Doc" in the Figure 1), one using dependency co-occurrences only ("Dep"), and one using both sets of features combined ("Doc+Dep").

From Ratinov et al. (2010) we took both the Collobert & Weston model (200 unscaled dimensions) and the HLBL model (100 scaled dimensions).[2] The LDA model was the default implementation included with the Gensim package (Řehůřek and Sojka, 2010): an incremental algorithm based on Hoffman et al. (2010), over the English Wikipedia, using TF-IDF adjusted co-occurrences.

As in clear from Figure 1 the SVD and NNSE models out-perform the other embeddings, though we cannot exclude that these models would be competitive with other parameter settings than those available, such as with a larger source corpus, or a different number of explanatory dimensions. Among the SVD and NNSE models, those based on dependency and combined (dependency and document) co-occurrences perform similarly well, and seem to have some advantage over document co-occurrences alone. And at this dimensionality setting SVD and NNSE seem similarly successful. As a result, in the subsequent analyses we concentrate on the combined SVD and combined NNSE models.

[2]These proved most effective among the scaling/dimensionality settings available at http://metaoptimize.com/projects/wordreprs/.

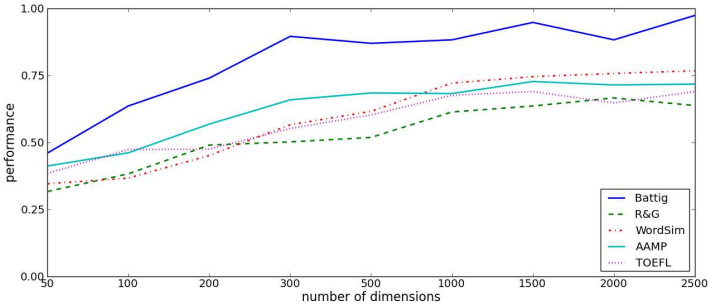


Figure 3: Behavioural evaluation of NNSE models with range of dimensionality (see Section 3.1.1 for more details).

We next examined how the choice of dimensionality affects the chosen SVD model (i.e., using both document and dependency co-occurrences) as evaluated with these behavioural tasks. As can be seen in Figure 2 performance peaks around the 200-300 dimension mark. In some tasks there is a fall-off in performance for higher dimensionalities, presumably as later dimensions are more noisy and/or irrelevant to these unsupervised tasks.

Finally we consider the effect of dimensionality for the NNSE models. In Figure 3 we see a very different pattern. For low dimensionalities it dramatically under-performs the SVD models, but at larger scales it continues to have an upward trend, peaking at similar levels.

In summary, these experiments demonstrate that NNSE and SVD models have similar peak performance, but with very different demands for dimensionality. NNSE needs at least 1000 dimensions to perform well, which may be understandable given that these dimensions must adequately describe all 40,000 words in the vocabulary used, and each word has on average 50 features active. The SVD model has very good performance at a dimensionality in the low hundreds, since it can pack information much more densely into a given number of features, and performance falls off as more noisy or irrelevant dimensions are added.

3.1.2 Neurosemantic Decoding Experiments

Mitchell et al. (2008) introduced a new task in *neurosemantic decoding* – using models of semantics to learn the mapping between concepts and the neural activity which they elicit during neuroimaging experiments. The dataset used here is that described in detail in Mitchell et al. (2008) and released publicly[3] in conjunction with the NAACL 2010 Workshop on Computational Neurolinguistics (Murphy et al., 2010). The Functional MRI (fMRI) data is from 9 participants while they performed a property generation task for each of 60 everyday concrete concepts: 5 exemplars of each of 12 semantic classes (mammals, body parts, buildings, building parts, clothes, furniture, insects, kitchen utensils, miscellaneous functional artifacts, work tools, vegetables, and vehicles). Each concept was presented six times and the resulting data-points were averaged to yield a single brain image for each concept, made up of approximately 20 thousand features (each a three-dimensional pixel, or "voxel").

[3] http://www.cs.cmu.edu/afs/cs/project/theo-73/www/science2008/data.html

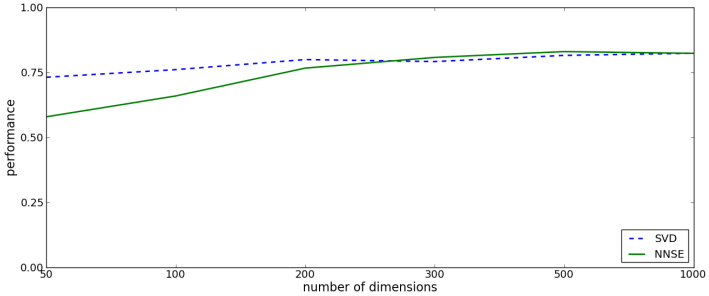


Figure 4: Neural activity test of NNSE and SVD models at range of dimensionalities (see Section 3.1.2 for details).

As in Mitchell et al. (2008), a linear regression model was used to learn the mapping from semantic features to brain activity levels. For each participant and selected fMRI feature we train a model to learn its activation as a regularised linear combination of the semantic features:

$$f = \mathbf{C}\beta + \lambda||\beta||^2 \tag{5}$$

where f is the vector of activations of a specific fMRI feature for different concepts, the matrix $\mathbf{C}$ contains the values of the semantic features for those concepts, β is the vector of weights we must learn for each of those (corpus-derived) features, and λ tunes the degree of regularisation.

The linear model was estimated with a least squared errors method and $L2$ regularisation, selecting the lambda parameter from the range 0.0001 to 5000 using Generalized Cross-Validation (see Hastie et al., 2011, p.244). The activation of each fMRI voxel in response to a new concept that was not in the training data was predicted by a β-weighted sum of the values on each semantic dimension, building a picture of expected the global neural activity response for an arbitrary concept. Again following Mitchell et al. (2008) we use a leave-2-out paradigm in which a linear model for each neural feature is trained in turn on all concepts minus 2, having selected the 500 most stable voxels in the training set using the same correlational measure across stimulus presentations. For each of the 2 left-out concepts, we predict the global neural activation pattern, as just described. We then try to correctly match the predicted and observed activations, by measuring the cosine distance between the model-generated estimate of fMRI activity and the that observed in the experiment. If the sum of the matched cosine distances is lower than the sum of the mismatched distances, we consider the prediction successful – otherwise as failed.

As can be seen in Figure 3.1.2 the peak performance of both combined SVD and combined NNSE models are close to identical. Again we see an upward trend for the NNSE model. However in the SVD model there is no fall-off for larger dimension sizes, which can be attributed to the supervised nature of this test – the feature selection and regularisation stages in the construction of the linear models should be able to ignore or down-weight noisy or irrelevant features. It should also be noted that the signal/noise ratio in brain data is quite low, and this test may be subject to a performance ceiling (see Levy and Bullinaria, 2012; Murphy et al., 2012).

	SVD_{300}	$NNSE_{50}$	$NNSE_{300}$	$NNSE_{1000}$
Sparsity level (% of zeros)	0	81.94	90.39	99.95
Average number of words per dimension	35560.0	6422.4	3418.5	1818.2
Average number of dimensions per word	300.0	9.0	28.8	51.1

Table 1: Comparison of sparsity level of SVD and NNSE models, the top 2 performing representations in Section 3.1. Compared to the dense (non-sparse) SVD representation, NNSE results in significantly sparser embeddings than SVD.

3.2 Evaluating Level of Sparsity

In this section, we evaluate the sparsity of NNSE relative to the dense representations of SVD. The degree of sparsity in these two top-performing representations (Section 3.1) are compared in Table 1, for SVD at $k = 300$, and for NNSE with $k = 50, 300, 1000$. Given one such representation $A \in \mathbb{R}^{n \times m}$, where $A_{i,:}$ is the new embedding for the i^{th} word, *sparsity level* measures the fraction of zero entries in this matrix out of the total $n \times m$ entries. *Average number of words per dimension* measures the average number of non-zero entries for each column of A, where each column corresponds to one dimension. *Average number of dimensions per word* computes the average number of non-zero entries for each row of A, where each row is a representation for one word, as already mentioned above. We note that since SVD is a dense representation, varying k in this case is not going to result in changes in the level of sparsity and values of the other two metrics compared in Table 1. Hence, in Table 1, we only present the results for SVD_{300} for reference.

From Table 1, based on the sparsity level measurements, we observe that NNSE results in significantly sparser representations compared to the dense representation learned by SVD. We observe that as k increases, NNSE estimates even sparser representations. This might possibly be due to the fact that with a small number of available dimensions (e.g., $k = 50$), each dimension is used to represent multiple latent concepts, resulting in non-zeros values for larger number of entries in each column and thereby in the full matrix A. In other words, each dimension acts as a mixture of latent concepts. However, we note that even at such settings (i.e., $k = 50$), NNSE achieves high sparsity levels (81.9%) compared to the dense SVD.

From Table 1, we observe that as k increases, the average number of words per dimension decreases, from 6422.4 with $k = 50$ to 1818.2 with $k = 1000$, i.e., each dimension becomes sparser as the number of available dimensions increases. This may be due to the fact, with smaller k (e.g., $k = 50$), NNSE has to compress the input data X into a smaller number of dimensions, resulting in coarser granularity for each dimension.

Finally, from Table 1, we observe that as k increases, NNSE uses a larger number of dimensions per word, but even then, the resulting embeddings are significantly sparser compared to SVD, where each word is represented using all available k dimensions.

3.3 Evaluating Interpretability

In the previous two sections, we evaluated performance of various word embeddings in different external tasks, and also the level of sparsity in these representations. Based on the performance evaluations, we found SVD and NNSE to be the most effective. In this section, we evaluate how interpretable these representations are. In other words, we would like to measure how coherent each of the dimensions of these representation are, i.e., given a representation A, we would like to

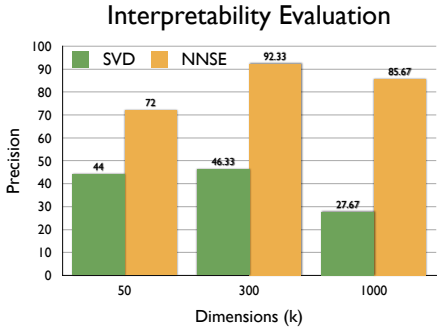


Figure 5: Comparison of interpretability of representations learned by SVD and NNSE for varying number of dimensions, k. Following (Boyd-Graber et al., 2009), word intrusion detection precision is used as the evaluation metric (higher precision implies higher interpretability). We observe that across all values of dimensions k, NNSE results in significantly more interpretable representations than SVD. See Section 3.3 for details.

determine how coherent each column (dimension) of A is.

Word Intrusion Detection Task: Following (Boyd-Graber et al., 2009), we use precision on a word intrusion detection task as the measure of coherence. The evaluation proceeds as follows: given a dimension (column) $A_{:,j}$, we first reverse-sort the words based on membership values of those words in the column (i.e., $A_{i,j},\ \forall 1 \leq i \leq n$). Next, we create a set consisting of the top 5 words from this ranked list, and also one word from the bottom half of this list, which is also present in the top 10th percentile of some other column in A. Thus, cardinality of this set is 6. The last word added from the bottom half is called an *intruder*. The goal is then to evaluate whether human subjects can identify this intruder word in a random ordering of the set. This process is repeated for each dimension, and the resulting precision is computed. This precision is used as the evaluation metric. Please note that we can compute precision in this setting as we know the true identity of the intruder word, it is just that this identity is hidden from the human evaluator. The idea behind this test is that if the current dimension is coherent, and thereby interpretable, then the human evaluator will be easily able to pick out the intruder word, thereby resulting in higher precision (higher is better). Example of such a set constructed from a dimension of the NNSE_{1000} is shown below,

{bathroom, closet, attic, balcony, quickly, toilet}

where *quickly* is the intruder, as the rest of the words form a coherent set listing different parts of a house. Following (Boyd-Graber et al., 2009), this evaluation scheme is also known as *reading tea leaves*, where it was used to measure coherence of probabilistic topic models.

We use Amazon Mechanical Turk (MTurk)[4] to get human judgements in the word intrusion detection task. Given an embedding matrix $A \in \mathbb{R}^{m \times k}$, we constructed n intruder sets as described above, one for each for the k dimensions. For $k > 300$, we randomly selected 300 dimensions for evaluation.

[4] http://mturk.amazon.com

Model	Top 5 Words (per dimension)
SVD_{300}	well, long, if, year, watch
	plan, engine, e, rock, very
	get, no, features, music, via
	features, by, links, free, down
	works, sound, video, building, section
$NNSE_{1000}$	inhibitor, inhibitors, antagonists, receptors, inhibition
	bristol, thames, southampton, brighton, poole
	delhi, india, bombay, chennai, madras
	pundits, forecasters, proponents, commentators, observers
	nosy, averse, leery, unsympathetic, snotty

Table 2: Examples of top 5 words for 5 randomly chosen dimensions from each of SVD_{300} and $NNSE_{1000}$. We observe that the dimensions in $NNSE_{1000}$ are much more semantically coherent (and thereby interpretable) compared to those in SVD_{300}.

Detecting the intruding word in each such set was presented as a separate decision point, called Human Intelligence Task (HIT). Each HIT was evaluated by 3 different workers (turkers), and a compensation of $0.01 was provided for each feedback. Majority voting over the three responses was used as the final decision on a given set (HIT).

Discussion: Experimental results comparing precision of the SVD and NNSE representations on the word intrusion task for different value of k are presented in Figure 5. From this figure, we observe that, for all values of k, representations learned by NNSE are considerably more interpretable compared to those estimated by SVD. We find that interpretability for both of them peak at $k = 300$.

For qualitative comparison, top 5 words from 5 randomly selected dimensions each of SVD_{300} and $NNSE_{1000}$ are presented in Table 2. From this, we get further anecdotal evidence about the higher interpretability of NNSE compared to SVD.

4 Related Work

Corpus-derived models of semantics have been extensively studied in the NLP and machine learning communities (Collobert and Weston, 2008; Ratinov et al., 2010; Turney and Pantel, 2010; Socher et al., 2011; Huang et al., 2012). Additionally, dimensionality reduction techniques such as SVD, and topic distributions learned by probabilistic topic models such LDA (Blei et al., 2003) can also be used to induce word embeddings. Although the embeddings learned by these methods have many overlapping properties, to the best of our knowledge, none of these previous proposals satisfy the three desirable properties: effective in practice, sparse, and interpretable. We find that Non-Negative Sparse Encoding (NNSE), a variation on a matrix factorization technique previously studied in the machine learning community, can result in semantic models which satisfy all three properties listed above.

In terms of interpretability, the NNSE is dramatically more effective than equivalent SVD models, and is comparable with LDA models evaluated in Boyd-Graber et al. (2009). In this context, it is interesting to compare it to another sparse interpretable model based on corpus co-occurrences, such as Strudel (Baroni et al., 2010). One major difference is that Strudel explicitly extracts descriptive properties, and that it does not involve a latent dimension discovery step. This has the advantage that

Weight	Top Words (per weighted dimension)
0.69	bike, mtb, bikes, harley, motorcycle, davidson, bicycle, cycling, biker, giro
0.35	canoe, raft, scooter, kayak, skateboard, bicycle, tractor, lawnmower, wheelchair
0.15	sedan, dealership, dealerships, dealer, convertible, camry, minivan, pickup, corolla
0.10	attorney, malpractice, lawyer, attorneys, lawyers, wrongful, litigation, injury, accident
0.08	earnhardt, speedway, irl, indy, racing, schumacher, mclaren, nascar, roush, race

Table 3: Examples of top 5 weighted dimensions in NNSE for the token *motorbike*, where each dimension is characterised by its top words.

Weight	Top Words (per weighted dimension)
0.84	raspberry, peach, pear, mango, melon, strawberry, banana, berry, cranberry, citrus
0.25	peaches, apricots, pears, cherries, blueberries, figs, oranges, plums, raspberries
0.10	birch, fir, spruce, pine, elm, mahogany, aspen, cypress, willow, cedar
0.08	kreme, pillsbury, falafel, brownie, pretzel, pesto, horseradish, oreo, guacamole, pita
0.07	patties, slices, cubes, tacos, wedges, chunks, enchiladas, burritos, fajitas, tortillas

Table 4: Examples of top 5 weighted dimensions in NNSE for the token *pear*, where each dimension is characterised by its top words.

it can model human tasks that directly involve properties and their types, such as the feature norming exercises mentioned earlier (which our NNSE model is not capable of currently). Conversely, it has a disadvantage in that it does not address the ubiquitous synonymy and polysemy among property labels. For example, the Strudel representation for *motorbike* has the properties (ordered by their strength of association): *ride, rider, sidecar, park, road, helmet, collision, vehicle, car, moped*.

The corresponding representation in NNSE is dominated by the five dimensions listed in Table 3. For each dimension we show the words that characterize its meaning, and the magnitude of its contribution to *motorbike*. These seem to describe the following respective classes which may correspond to topical usages of the word: (motor)cycling; leisure vehicles; vehicle sales; accidents and litigation; and racing. The corresponding listing for *pear* in Table 4, also exhibits some different senses of the word, covering *fruit*, *food* more generally, and *trees*.

Overall, we find that NNSE model is an alternative coding of the information stored in the SVD model. These models differ however in how that information is distributed across dimensions. The SVD model, as expected, is maximally compact, compressing as much information as possible into the minimum number of dimensions, but yielding individual dimensions that are very hard to interpret. The NNSE model is equally effective in modeling human judgements and brain activity elicited during lexical semantic tasks, but has a sparse structure which closely matches some common assumptions in cognitive science about conceptual representations.

5 Conclusion

In this paper, we propose the novel application of Non-Negative Sparse Embedding (NNSE), a variation on a constrained matrix factorization technique previously studied in the machine learning community (Hoyer, 2002; Mairal et al., 2010), to learn corpus-derived semantic models (word embeddings). To the best of our knowledge, this is the first approach which learns embeddings

which are effective in practice, sparse, and interpretable – a desirable list of properties which was not achievable by previously proposed methods.

There are still many ways in which we can extend and improve this new embedding method. First of all, we would like to test it as a component of core NLP tasks, such as chunking, named-entity-recognition, and parsing. We also plan to compare the individual NNSE dimensions to other benchmarks that explicitly cover categories and properties, such as feature norms, WordNet, and other collections of human judgements such as the 20Q data (Palatucci et al., 2009). Finally, we plan to look more closely at the relative contributions of different sources of input representations, such as dependency and document co-occurrences that underlie the current model, to examine the relationship between topical and attributional meanings that they correspond to.

Acknowledgments

We are thankful to Khaled El-Arini (CMU) and Min Xu (CMU) for many useful discussions on sparse coding. We thank Justin Betteridge (CMU) for his help with parsing the corpus, and Yahoo! for providing the M45 cluster over which the parsing was done. We also thank Marco Baroni (University of Trento) and Seshadri Sridharan (CMU) for assistance in preparing the behavioural benchmarks. We thank CMU's Parallel Data Laboratory (PDL) for making the OpenCloud cluster available. We are thankful to the anonymous reviewers for their constructive comments. This research has been supported in part by DARPA (under contract number FA8750-09-C-0179), NIH (NICHD award 1R01HD075328-01), and Google. Any opinions, findings, conclusions and recommendations expressed in this paper are the authors' and do not necessarily reflect those of the sponsors.

References

Agirre, E., Alfonseca, E., Hall, K., Kravalova, J., and Pas, M. (2009). A study on similarity and relatedness using distributional and WordNet-based approaches. *Proceedings of NAACL-HLT 2009*.

Almuhareb, A. and Poesio, M. (2004). Attribute-based and value-based clustering: An evaluation. In *Proceedings of EMNLP*, pages 158–165.

Baroni, M. and Lenci, A. (2010). Distributional Memory: A General Framework for Corpus-Based Semantics. *Computational Linguistics*, 36(4):673–721.

Baroni, M., Murphy, B., Barbu, E., and Poesio, M. (2010). Strudel: A corpus-based semantic model based on properties and types. *Cognitive Science*, 34(2):222–254.

Battig, W. F. and Montague, W. E. (1969). Category Norms for Verbal Items in 56 Categories: A Replication and Extension of the Connecticut Category Norms. *Journal of Experimental Psychology Monographs*, 80(3):1–46.

Blei, D. M., Ng, A. Y., and Jordan, M. I. (2003). Latent Dirichlet Allocation. *Journal of Machine Learning Research*, 3(4-5):993–1022.

Boyd-Graber, J., Chang, J., Gerrish, S., Wang, C., and Blei, D. (2009). Reading tea leaves: How humans interpret topic models. In *Proceedings of the 23rd Annual Conference on Neural Information Processing Systems*.

Bradford, R. B. (2008). An empirical study of required dimensionality for large-scale latent semantic indexing applications. *Proceedings of the 17th ACM, CIKM*, pages 153–162.

Bullinaria, J. A. and Levy, J. P. (2007). Extracting semantic representations from word co-occurrence statistics: A computational study. *Behavior Research Methods*, 39(3):510–526.

Callan, J. and Hoy, M. (2009). The ClueWeb09 Dataset.

Chen, S., Donoho, D., and Saunders, M. (2001). Atomic decomposition by basis pursuit. *SIAM review*, pages 129–159.

Collobert, R. and Weston, J. (2008). A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th international conference on Machine learning*, pages 160–167. ACM.

Deerwester, S., Dumais, S., Furnas, G., Landauer, T., and Harshman, R. (1990). Indexing by latent semantic analysis. *Journal of the American society for information science*, 41(6):391–407.

Finkelstein, L., Gabrilovich, E., Matias, Y., Rivlin, E., Solan, Z., Wolfman, G., and Ruppin, E. (2002). Placing search in context: the concept revisited. *ACM Transactions on Information Systems*, 20(1):116–131.

Garrard, P., Lambon Ralph, M. A., Hodges, J. R., and Patterson, K. (2001). Prototypicality, distinctiveness, and intercorrelation: Analyses of the semantic attributes of living and nonliving concepts. *Cognitive Neuropsychology*, 18(2):25–174.

Griffiths, T. L., Steyvers, M., and Tenenbaum, J. B. (2007). Topics in semantic representation. *Psychological Review*, 114(2):211–244.

Hall, J., Nilsson, J., Nivre, J., Eryigit, G., Megyesi, B., Nilsson, M., and Saers, M. (2007). Single Malt or Blended? A Study in Multilingual Parser Optimization. *Proceedings of CoNLL Shared Task Session*, pages 933–939.

Hastie, T., Tibshirani, R., and Friedman, J. (2011). *The Elements of Statistical Learning*, volume 18 of *Springer Series in Statistics*. Springer, 5th edition.

Hoffman, M. D., Blei, D. M., and Bach, F. (2010). Online Learning for Latent Dirichlet Allocation. In Lafferty, J., Williams, C. K. I., Shawe-Taylor, J., Zemel, R. S., and Culotta, A., editors, *Proceedings of NIPS*.

Hoyer, P. O. (2002). Non-negative sparse coding. In *Proceedings of the 12th IEEE Workshop on Neural Networks for Signal Processing*, pages 557–565. Ieee.

Huang, E., Socher, R., Manning, C., and Ng, A. (2012). Improving word representations via global context and multiple word prototypes. In *Proceedings of ACL*.

Jones, E., Oliphant, T., Peterson, P., and Et Al. (2001). SciPy: Open source scientific tools for Python.

Karypis, G. (2003). CLUTO: A Clustering Toolkit. Technical Report 02-017, Department of Computer Science, University of Minnesota.

Landauer, T. and Dumais, S. (1997). A solution to Plato's problem: the latent semantic analysis theory of acquisition, induction, and representation of knowledge. *Psychological Review*, 104(2):211–240.

Lehoucq, R. B., Sorensen, D. C., and Yang, C. (1998). *Arpack users' guide: Solution of large scale eigenvalue problems with implicitly restarted Arnoldi methods*. SIAM.

Levy, J. P. and Bullinaria, J. A. (2012). Using Enriched Semantic Representations in Predictions of Human Brain Activity. *Proceedings of the 12th Neural Computation and Psychology Workshop*, pages 292–308.

Lin, D. (1998). Automatic Retrieval and Clustering of Similar Words. In *Proceedings of COLING-ACL*, pages 768–774.

Lund, K. and Burgess, C. (1996). Producing high-dimensional semantic spaces from lexical co-occurrence. *Behavior Research Methods, Instruments, and Computers*, 28:203–208.

Lund, K., Burgess, C., and Atchley, R. (1995). Semantic and associative priming in high dimensional semantic space. In *Proceedings of the 17th Cognitive Science Society Meeting*, pages 660–665.

Mairal, J., Bach, F., Ponce, J., and Sapiro, G. (2010). Online learning for matrix factorization and sparse coding. *The Journal of Machine Learning Research*, 11:19–60.

McRae, K., Cree, G. S., Seidenberg, M. S., and McNorgan, C. (2005). Semantic feature production norms for a large set of living and nonliving things. *Behavior Research Methods, Instruments, & Computers*, 37(4):547–559.

Miller, G. A., Beckwith, R., Fellbaum, C., Gross, D., and Miller, K. (1990). Introduction to WordNet: an on-line lexical database. *International Journal of Lexicography*, 3(4):235–244.

Mitchell, T. M., Shinkareva, S. V., Carlson, A., Chang, K.-M., Malave, V. L., Mason, R. A., and Just, M. A. (2008). Predicting Human Brain Activity Associated with the Meanings of Nouns. *Science*, 320:1191–1195.

Murphy, B., Korhonen, A., and Chang, K. K.-M., editors (2010). *Proceedings of the 1st Workshop on Computational Neurolinguistics, NAACL-HLT*, Los Angeles. ACL.

Murphy, B., Talukdar, P., and Mitchell, T. (2012). Selecting Corpus-Semantic Models for Neurolinguistic Decoding. In *Proceedings of *SEM-2012*.

Murphy, G. (2004). *The big book of concepts*. MIT Press.

Nancy Ide and Keith Suderman (2006). The American National Corpus First Release. *Proceedings of the 5th LREC*.

Nation, P. and Waring, R. (1997). Vocabulary size, text coverage and word lists. In Schmitt, N. and McCarthy, M., editors, *Vocabulary Description acquisition and pedagogy*, pages 6–19. Cambridge University Press.

Padó, S. and Lapata, M. (2007). Dependency-based construction of semantic space models. *Computational Linguistics*, 33(2):161–199.

Palatucci, M., Hinton, G., Pomerleau, D., and Mitchell, T. M. (2009). Zero-Shot Learning with Semantic Output Codes. *Advances in Neural Information Processing Systems*, 22:1–9.

Rapp, R. (2003). Word Sense Discovery Based on Sense Descriptor Dissimilarity. *Proceedings of the Ninth Machine Translation Summit*, pp:315–322.

Ratinov, L., Turian, J., and Bengio, Y. (2010). Word representations: a simple and general method for semi-supervised learning. *Proceedings of the 48th ACL*, 96(July):384–394.

Řehůřek, R. and Sojka, P. (2010). Software Framework for Topic Modelling with Large Corpora. In *Proceedings of New Challenges Workshop, LREC 2010*, pages 45–50. ELRA.

Rubenstein, H. and Goodenough, J. B. (1965). Contextual correlates of synonymy. *Communications of the ACM*, 8(10):627–633.

Sahlgren, M. (2006). *The Word-Space Model: Using distributional analysis to represent syntagmatic and paradigmatic relations between words in high-dimensional vector spaces*. Dissertation, Stockholm University.

Schunn, C. D. (1999). The Presence and Absence of Category Knowledge in LSA. In *Proceedings of the 21st Annual Conference of the Cognitive Science Society.*, Mahwah. Erlbaum.

Schütze, H. and Pedersen, J. (1993). A Vector Model for syntagmatic and paradigmatic relatedness. In *Making Sense of Words Proceedings of the 9th Annual Conference of the University of Waterloo Centre for the New OED and Text Research*, pages 104–113.

Socher, R., Huang, E., Pennington, J., Ng, A., and Manning, C. (2011). Dynamic pooling and unfolding recursive autoencoders for paraphrase detection. *Advances in Neural Information Processing Systems*, 24.

Turney, P. D. and Pantel, P. (2010). From Frequency to Meaning: Vector Space Models of Semantics. *Artificial Intelligence*, 37(1):141–188.

Vinson, D. and Vigliocco, G. (2008). Semantic feature production norms for a large set of objects and events. *Behavior Research Methods*, 40(1):183–190.

Combining Wordnet and Morphosyntactic Information in Terminology Clustering

Agnieszka Mykowiecka Małgorzata Marciniak
Institute of Computer Science, PAS, Jana Kazimierza 5, 01-248 Warsaw, Poland
agn@ipipan.waw.pl, mm@ipipan.waw.pl

ABSTRACT
The paper presents results of clustering terms extracted from economic articles in Polish Wikipedia. First, we describe the method of automatic term extraction supported by linguistic knowledge. Then, we define different types of term similarities used in the clustering experiment. Term similarities are based on Polish Wordnet and morphosyntactic analysis of data. The latter takes into account: term contexts, coordinated sequences of terms, syntactic patterns in which terms appear and words that are parts of terms (such as their heads and modifiers). Then we performed several experiments with hierarchical clustering of the 400 most frequent terms. We present the results of clustering when different groups of similarity coefficients are applied. Finally, we present an evaluation that compares the results with manually obtained groups. Our results prove that morphosyntactic information can help or even serve themselves for initial clustering of terms in semantically coherent groups.

KEYWORDS: terminology extraction, terminology clustering, Polish.

Proceedings of COLING 2012: Technical Papers, pages 1951–1962,
COLING 2012, Mumbai, December 2012.

1 Introduction

Many NLP applications, like text indexing, information extraction or question answering, rely on sets of predefined concepts which can be identified within texts, on a given subject. Such concepts form either hierarchical ontologies or flat terminology lists. Although there are already very many works aimed at making these kinds of resources available, still, the existing data is very limited with regard to the chosen application/knowledge domain as well as natural language addressed. An overview of existing approaches to automatic terminology extraction is included in (Pazienza et al., 2005) while problems with ontology creation are described, among others, in (Cimiano, 2006).

For some NLP purposes flat terminology lists are sufficient, but for others, like IE, ontologies representing relations between particular terms are more adequate. Unfortunately, domain ontologies' availability is very limited and what is even more crucial – they are rarely adequate for the purpose at hand. They are either too specific or too general, or do not cover the appropriate domain, or they are outdated. Ontology reuse and projection is still a very difficult and unresolved issue. As a result, when general ontologies like SUMO (Pease and Niles, 2001) are insufficient, or are not available in a particular language, a new dedicated ontology has to be created. In such a case, a list of concepts to be included in the ontology can be prepared manually by a domain expert, or can be (at least initially) extracted from texts similar to those which are to be processed.

In this paper we address the problem of finding interesting concepts in Polish economic texts and organizing them in coherent groups which can be further analyzed more easily than a long unstructured term list. The identified sets of terms will be used for developing a domain model which will constitute the base of an information extraction system. The basic idea is to facilitate the construction of IE systems by automation of the initial stage of domain model creation, i.e. defining concepts which can be addressed in the selected type of texts, and relate them to particular language expressions. The same method may be used later on to gather new concepts which appear in time and relate them to the already existing ones on the basis of the contexts of their occurrences.

As a test domain we have chosen economy, in particular, economic articles of Polish Wikipedia. In the paper we present the process of selecting term candidates, their ordering according to the defined importance measure and clustering. To make the approach usable for many domains we assume that only general language resources like a morphological tagger and Wordnet are used.

2 Data

The experiment was conducted on the economic articles taken from the Polish Wikipedia. Only textual content of these articles was taken into account. The data was collected in 2011 and contains 1219 articles that have economics related headings and articles linked to them. The data contains about 456,000 tokens. An initial linguistic analysis of the plain texts was performed. It consisted of the following steps:

- Segmentation into tokens. We distinguish words (365,042), numbers (14,906) and punctuation marks (76,239).
- Morphological annotation. To each word we assign: its base form, part of speech and complete morphological characterization. The annotation is based on the results obtained

by the publicly available Polish POS tagger Pantera (Acedański, 2010) that cooperates with a general-purpose morphological analyzer of Polish Morfeusz SGJP (Woliński, 2006). The Pantera tagger allows us to define a separate dictionary in which we can describe unknown tokens. So we defined an additional domain dictionary containing 741 entries of word-forms (not recognized by the Morfeusz analyzer). Many of them have more than one morphological characterization, e.g. *podsektor* 'subsector' that represents a noun in the nominative or accusative case. In case of adjectives there are usually more possible interpretations, for example the word-form *proekologiczne* 'proecological' has 7 different characterizations in the additional dictionary. Our dictionary does not describe all unknown tokens as we decided not to analyze foreign words and proper names that are not present in Morfeusz. In the data, many notions are translated into foreign languages, e.g. *Spółdzielnia europejska* 'European Cooperative Society' has in the data its Latin equivalent 'Societas Cooperativa Europaea', and all three tokens are annotated with *ign* tag that indicates an unknown word. So still 10,796 tokens have no morphological characterization.

- Improving tagger results. We defined 84 rules in Spejd (Przepiórkowski, 2008) (a cascade of regular grammars) in order to correct Pantera decisions, and to extend some descriptions. The advantage of using this method is the possibility of taking contexts into account. Spejd rules are particularly helpful in correcting some regular tagging errors in frequently occurring phrases. They corrected or extended over 4,000 token descriptions. The changes in the tagset consisted in the introduction of the *number* tag assigned to Arabic as well as Roman numerals, and extending descriptions of abbreviations (*brev* POS). In Morfeusz, an abbreviation is characterised only by a specification whether it has to be followed by a full stop. We extended its description with information about the type of word or phrase it abbreviates to allow for constructing correct grammatical phrases containing the abbreviation.

- Removing improperly recognized sentence endings after abbreviations.

3 Terms identification

The very first problem while doing terminology extraction is to define what a domain term really is. Unfortunately, there exists no strict definition of this concept and usually only pragmatical approaches are taken. For the purpose of this work we defined a term as a noun phrase which occurs more frequently in domain specific texts than in the general language. Thus, we decided not to follow the approach in which linguistic information is neglected (like (Wermter and Hahn, 2005)), but to use morphological information at the stage of candidate selection. We also use this information while defining similarity coefficients.

For term candidates we choose noun phrases of a limited internal complexity, i.e. we assume that they are built according to one of the following syntactic schemata of which the first four are very frequent and the last one is much less common. In particular, we do not allow for prepositional phrases to occur within the terms (compare the results of the terminology extraction task described in (Marciniak and Mykowiecka, 2012)).

- a single noun or an abbreviation, e.g. *bank* 'bank', *EC* 'European Commission';
- a noun followed (or, more rarely, preceded or surrounded) by an adjective, e.g. $administracja_{n}\ publiczna_{adj}$ 'public administration', $wysoki_{adj}\ dochód_{n}$ 'high income', $ogólna_{adj}\ sytuacja_{n}\ gospodarcza_{adj}$ 'general economic situation';

- a noun followed by another noun in genitive, e.g. $kurs_{n,nom}$ $walut_{n,gen}$ 'exchange rate';
- a combination of the last two structures, e.g. $walne_{adj,nom}$ $zgromadzenie_{n,nom}$ $akcjonariuszy_{n,gen}$ 'general meeting of shareholders',
- a noun preceded by an adjectival phrase, e.g. *wschodnia i południowa Afryka* 'East and South Africa'.

For recognizing the selected types of nominal phrases, a cascade of six simple shallow grammars was created. Its rules operate on the results of morphological analysis described in section 2. The gradual phrase creation starts with adjective modifiers and then genitive modifiers are added.

To make our data a little more coherent (domain related) we eliminated time related expressions (names of months, nouns like 'hour', 'minute', adjectives like 'late') which are studied separately. We also excluded selected sets of nouns and adjectives which can be thought of as a kind of 'stop words' for the terminology extraction task, that is words which can be used in very many contexts and which themselves do not constitute terms elements. These are adjectives like *dany* 'given' or pronouns. The list was built up in our previous terminology extraction experiment (Mykowiecka and Marciniak, 2012) and supplemented with new elements in the current one. The list which contains 65 words is used only additionally, many such phrases can be eliminated at the later stage of ordering term candidates.

Applying the adopted set of rules to the data resulted in obtaining 80,212 types of phrases in which there are 45,144 top level types occurring 104,576 times. The longest (non overlapping) phrases which can be built starting from subsequent text positions were extracted. Their internal structure was annotated by markers showing subphrase boundaries. For the resulting set of phrases, we performed an analysis similar to that proposed in (Frantzi et al., 2000). In this approach both the entire high-level phrases and the internal nominal subphrases are taken into account, e.g. in the phrase *rzecznik dyscypliny finansów publicznych* 'advocate for public finance discipline' we also encounter the subphrases *dyscyplina finansów publicznych* 'public finance discipline' and *finanse publiczne* 'public finance'. Taking subphrases into account is important, as for example, the following phrases: *kapitał obrotowy* 'working capital' *system emerytalny* 'pension system' and *akt notarialny* 'notarial deed' did not occur in isolation in the data.

As Polish is an inflectional language, phrases which are identified within the text are of different forms (e.g. $\text{kurs}_{n,acc}$ $\text{walut}_{n,gen}$, $\text{kursie}_{n,loc}$ $\text{walut}_{n,gen}$ 'exchange rate') so the usual processing stages like counting phrase frequencies and preparing a list of phrase types became difficult. To overcome this problem we produce an artificial base form of every identified phrase occurrence, by taking base forms assigned by the tagger to its elements, i.e. $\text{kurs}_{n,nom}$ $\text{waluta}_{n,nom}$.

All extracted phrases are ranked according to the value of a specially defined coefficient (C-value) which is calculated basing on the occurrences of the phrase in the text as a stand alone phrase and its occurrences within other phrases from the list. This allows us to identify terms which never (or very rarely) occur in isolation, and to some extent, to filter out erroneous phrases (those which are recognized by a shallow grammar as one phrase but in fact are incomplete although grammatically sound) or are built up of more than one phrase (like *zamieszkania właściwość różnych organów* 'living jurisdiction of different authorities' which resulted from *podlegają z uwagi na swe miejsce zamieszkania właściwości różnych organów* 'fall under jurisdiction of different authorities depending on their place of living').

We used a slightly modified definition of C-value which is given below. p – is a phrase under consideration, LP – is a set of phrases containing p, and P(LP) – the number of types of phrases

differing in elements which are adjacent to p, that is the sum of different direct left and right one-word contexts counted separately (e.g. if the phrase *angielski bank* 'English bank' occurs in three types of longer phrases: *angielski bank inwestycyjny* 'English investment bank', *najstarszy angielski bank inwestycyjny* 'the oldest English investment bank' and *bankructwo najstarszego angielskiego banku inwestycyjnego* 'bankruptcy of the oldest English investment bank' , P(LP) is set to 2).

$$C-value(p) = \begin{cases} lc(p) * freq(p) - \frac{1}{P(LP)} \sum_{lp \in LP} freq(lp), & if\ P(LP) > 0, \\ lc(p) * freq(p), & if\ P(LP) = 0 \end{cases}$$

where $lc(p) = log_2(length(p))$ if $length(p) > 1$ and 0.1 otherwise;

To eliminate phrases which are not from the economy domain, but occur in all types of texts similarly often, we compared the list of phrases obtained for Wikipedia economic texts with phrases obtained form the balanced one million word subcorpus of NKJP (the corpus of general Polish (Przepiórkowski et al., 2012)) using the same processing schema. Table 1 shows how many terms are recognized in both corpora and how many of them have a grater C-value in each data set. Less than 10% of terms recognized in economic texts are also recognized in NKJP data — the longest common phrases have 5 words.

Table 1: Comparison with general corpus

Terms	common	C-value greater in econom.	C-value greater in NKJP
1-word	4089	767	3322
2-words	2558	1133	1425
3-5-words	201	98	103
Total	7848	1998	4850

There are a number of phrases with a greater C-value for NKJP subcorpus and relevant to the economic domain, e.g. *skarb państwa* 'state treasury', *urząd skarbowy* 'treasury office', *ustawa budżetowa* 'budget act'. So we decided that phrases which have a greater C-value counted in the context of general texts than that counted for economic data, should be manually inspected.

For the clustering experiment, the first 400 terms from the list, ranked according to the C-value coefficient, were chosen. On the bases of the comparison with NKJP terms, from the original list we removed one-word terms like: *grupa* 'group', *przyklad* 'example', *funkcja* 'function'; and a few multi-word terms, e.g: *wszcząć postępowanie* 'initiation of proceedings', *różny rodzaj* 'different types'. Removed terms were substituted with the subsequent terms from the list. Choosing a relatively small number of terms was motivated by the need for manual checking of the results.

4 Defining similarity features

At the next stage of domain model creation, a list of terms is organized into clusters which group elements addressing similar concepts. This is most frequently done manually by domain experts, but manual processing of a long list of names is time consuming and prone to errors. To perform this task automatically it is necessary to decide how to represent term similarity. In our approach we decided to use morphosyntactic information (in this case we follow the ideas presented in (Nenadić et al., 2004)), as well as information included in Polish Wordnet.

4.1 Contextual similarity

Contextual similarity is based on the contexts in which terms appear. We consider left and right contexts of terms separately. Contexts are not allowed to cross sentence or paragraph boundaries. We decided to consider the following types of context patterns:

- POS contexts. In this case patterns are strings of part of speech tags. We took into account patterns of 2 to 4 elements. If sentence boundaries are encountered, the context is shorter.
- The base form of the token preceding and following the term (separately).
- The base form of the nearest verb. If there are no verbs encountered within the sentence boundaries, the context is set to the null context.
- The base form of the nearest noun type token (e.g. nouns, gerunds).
- The nearest preposition.

In the case of the last two contexts, if there are no prepositions or nouns between the term and a verb, the context is set to the null context.

4.2 Coordination

Co-occurrence of terms in coordinated sequences is the next type of information we take into account when finding similar terms. We find sequences of terms connected by conjunctions or commas. All terms should be in the same grammatical case, and can be preceded by a preposition. The following example of a coordinated sequence: *<akcje>, <obligacje> i <instrumenty pochodne>* '<shares>, <bonds> and <derivatives>' joins terms denoting various financial instruments. We also consider coordination of prepositional phrases that consist of a preposition and a term, where terms are in the same grammatical case. See an example of such a phrase: *dla <osoby prywatnej> i dla <jednostki organizacyjnej>* 'for <a private person> or for an <organization unit>'. We do not check the wordforms of prepositions in coordinated sequences, but the grammatical case of terms has to be the same. This is a rough method of coordinated terms recognition and needs further refinements. For example, it will not recognize the following coordination: *eksportowane do Niemiec*$_{gen}$ *i na Litwę*$_{acc}$ 'exported to Germany and to Lithuania'. However, it excludes the majority of cases where two terms preceded by prepositions are separated by a comma and they belong to two different parts of a sentence, e.g: *W <Polsce>*$_{loc}$*, mimo <wpisania pojęcia konsumenta>*$_{gen}$ *do konstytucji ...* 'In [Poland], despite of [entering the notion of a consumer] into the constitution ...'.

In our data we detected 5,885 coordinated sequences of terms, which join 9,807 different pairs of terms. The vast majority of them occurred only a few times, only 9 pairs of terms occurred in coordinated sequences more than 10 times. The most frequent pair of terms *<towar>* 'product' and *<usługa>* 'service' occurred 74 times. For the selected 400 terms, 157 coordination pairs were found within the texts.

4.3 Syntactic patterns

Besides the coordination sequences we recognize several syntactic patterns that indicate similarity between terms. These patterns contain the following Polish phrases/words: *taki jak* 'such as', *czyli* 'or, that is', *na przykład* 'for example', *to jest* 'that is' and *zarówno...jak i* 'both...and also' and their equivalents. The first four patterns have the following construction:

<term1> [key phrase] <list of terms>

while the last one has slightly different form:

[key phrase 1] <term1> [key phrase 2] <list of terms>.

In the above patterns <list of terms> is the coordination of terms with limitations and internal similarity measures described in 4.2. These constructions recognize similarity between pairs built up from <term1> and all terms in the <list of terms>. Let us consider the following example:

wiele cech <oferty rynkowej> takich jak <cena>, <jakość> i <forma płatności>

'many features of <market offer> such as <price>, <quality> and <form of payment>'

The above phrase indicates that following 3 pairs of terms are similar:

- *<oferty rynkowej>* 'market offer' and *<cena>* 'price'
- *<oferty rynkowej>* 'market offer' and *<jakość>* 'quality'
- *<oferty rynkowej>* 'market offer' and *<forma płatności>* 'form of payment'

In the data we detected 545 pairs of similar terms recognized by the above lexical patterns from which 85 are used in the clustering experiment of 400 terms.

4.4 Lexical Similarity

Terms that have the same head element usually describe related concepts, for example *kurs obcej waluty* 'foreign currency exchange rate' and *kurs dolara* 'dollar exchange rate' have the same head element *kurs* 'exchange rate', and describe similar notions. In the task we promote terms with the same head. If the heads of two terms are the same, then the head similarity coefficient for these phrases is set to 1. We do not consider different meanings of heads so the following phrases: *klasa robotnicza* 'working class' and *klasa szkolna* 'classroom' are set to 1.

Terms that have common words are also more related than those without any common words. Counting them we exclude common heads. For example the common adjective *budżetowy* 'budgetary' indicates that the following terms are to a certain degree similar: *dotacja budżetowa* 'budget subsidy' *wydatek budżetowy* 'budget expenditure' and *założenia budżetowe* 'budget assumption' To establish these types of similarities for all term pairs we counted how many common words they have (except common head elements). The similarity between two terms is equal to the number of common modifiers divided by the number of modifiers of the longer term.

4.5 Wordnet similarity

Polish Wordnet (PlWordNet, (Piasecki et al., 2009)) is one of the biggest resources of the type introduced by Princeton Wordnet (Miller, 1995), but it mainly describes general language and it contains mostly one word items. Domain terminology usually contains a prevalence of multiword expressions. On our list, among 400 terms, 130 are one word expressions and 270 are longer. All one word terms have at least one sense defined in PlWordNet. For multiword expressions the situation is drastically different: 52 phrases are defined in PlWordNet while 218 phrases are not. For 3 of them their head elements are also not described: *Brytania* 'Britain, *środki* 'resources', *Adam* 'Adam' (two of them are proper names, for the word 'resources' only the singular form is defined which has different meanings).

The above statistics show that in order to utilize information given in PlWordnet, operating only on information in the phrases which appear within the data, is insufficient. Thus, we decided

to calculate the similarity between terms on the basis of information given both on the terms themselves and on their head elements. The schema of calculating similarity between terms A and B was defined in two steps. In the first one an initial similarity value is set to:

- if both terms appear in PlWordNet and share at least one synset — 1/minimum of synsets defined for A and B;
- otherwise, if A appears in PlwordNet and belongs to the same synset as one of the B hiper- or hiponims – 0.5/number of synsets of A;
- otherwise, if B appears in PlwordNet and belongs to the same synset as one of the B hiper- or hiponims – 0.5/number of synsets of B;

In the second step, when at least one of the terms is longer than one word, the similarity value is assigned to a minimum from the number resulting from the following additions and 1:

- if A is a multiword term:
  - if A's head belongs to at least one synset to which B also belongs: +0.25;
  - otherwise, if A's head belongs to at least one synset to which the head of a multiword A also belongs: +0.1;
  - if B belongs to the same synset as a hiper- or hiponim of A's head: +0.15
- if B is a multiword term
  - if B's head belongs to at least one synset to which any hiper- or hiponim of B also belongs: +0.05
  - if (oneword) A belongs to the same synset as the head of B: +0.2
  - if (oneword) A belongs to the same synset as a hiper- or hiponim of B: +0.1

This process resulted in 298 nonzero coefficients. 19 pairs were judged to be equivalent (similarity 1), e.g. *<dochód>-<zysk>* 'income-gain', *<prawo>-<zasada>* 'law-rule. One example was incorrect: *<model>-<klient>* 'model-customer'.

4.6 Overall Similarity

All similarity tables were rescaled in such a way that the highest coefficient for each measure is equal 1. In all experiments described below, an overall similarity of a pair of terms was calculated as a weighted sum of up to 19 coefficients:

- neighboring left/right form (lf, rf),
- left/right POS contexts of length 2/3/4 (c2l, c3l, c4l, c2r,c3r,c4r),
- first left/right verb, noun, preposition (l_v, l_n, l_p, r_v, r_v, r_p),
- coordination coefficient (crd),
- syntactic patterns coefficient (syn)
- common head coeff. (head),
- common modifiers coeff. (mod),
- wordnet similarity (wdnet).

5 Clustering

Automatic clustering was done using MultiDendrograms (Fernández and Gómez, 2008) performing hierarchical clustering. From several options, the unweighted average of similarity coefficient values was selected on the basis of the results of the preliminary tests.

As no resource which can be used as a reference set exists, to enable the evaluation of the results, a manually prepared version of the partition of 400 terms was created. The only instruction given to a person doing this task was to group similar elements even if they cannot be treated as subtypes of one concept. The result, which was verified by the second annotator, comprises 127 group, of which 30 contain only one element. The maximal group size is 14.

In the experiments, different weighting schemata of the coefficients used to calculate the overall similarity measure were tested. The exact values of the weights assigned for some selected models are given in Table 2. To check the impact of morphosyntactic features on the result obtained while using only PlWordNet data, automatic clustering was done for the models belonging to the three groups described below.

- only Wordnet similarity (as defined above) has nonzero (i.e. 1) weight – model W1,
- all weights are non zero — models W4, W5 and W6,
- Wordnet similarity is assigned 0, its weight is distributed among other weights which are initially set as in W4 – W2.

Table 2: The selected models characterization

	fl	fr	syn	c2l	c2r	c3l	c3r	c4l	c4r	crd	mod	head	r_v	r_n	r_p	l_v	l_p	l_n	wdnet
w2	.11	.11	.058	.058	.058	.033	.033	.013	.013	.058	.058	.108	.058	.058	.018	.058	.043	.053	.00
w4	.10	.10	.050	.050	.050	.025	.025	.005	.005	.050	.050	.100	.050	.050	.010	.050	.035	.045	.15
w5	.13	.13	.100	.015	.020	.010	.010	.003	.002	.050	.050	.080	.040	.050	.050	.100	.030	.030	.10
w6	.10	.10	.050	.050	.050	.025	.020	.001	.001	.050	.050	.100	.020	.050	.010	.123	.040	.040	.13

The results of clustering were compared using the B-cubed measure (Bagga and Baldwin, 1998) positively evaluated in different experiments, e.g. (Amigó et al., 2009). This measure counts precision for every group element so it is sensitive to both – presence and absence of the elements of groups. The results presented in Tab. 3 show some of the tested system configurations. Rows correspond to the combinations of weights given in Tab. 2. Each cell of the table contains precision, recall and F-measure results obtained when comparing the models to the manual clustering (rescaled into the 0-100 range).

Table 3: Model comparisons with manual grouping.

	127 groups	best F-value	nb of groups
W1	64.2/74.1/68,8	84.9/71/77.4	175
W2	62.4/60.9/61.7	82,5/56,9/68,2	205
W4	74.3/73.9/74.1	94.9/69.6/80.3	190
W5	76.8/73.6/75.1	90.6/69.6/78,7	175
W6	76.4/73.0/73.8	94.4/68.8/79.6	191

By adjusting the weights used in the definition of the similarity measure, we obtained an enhancement of the clusters matching which did not vary much (for reasonable weight distribution). For all these models the results were about 5% better than those obtained using only Wordnet data. Using morphosyntactic information alone also gave usable results at the level of about 62%.

Conclusions

In the paper we presented the results of the process of detecting coherent groups within terminological phrases extracted from real texts from the economy domian. The obtained results show that in the case where semantic information is lacking, morphosyntactic description of the contexts of term occurrences can help in a terminology clustering task. Even when only morphosyntactic features are available, the results achieved can make further manual clustering much easier. However, adding other sources of information, like Wordnet relation for phrase head elements, improves the results.

The F-measure of about 75% achieved when comparing the automatically obtained clusters to manually obtained groups is not high, but in the case of this task, which also proved difficult for well trained annotators, can be seen as good enough to be utilized in further domain ontology development. The presented method can be used for texts in any domain or language but the quality of the results highly depends on the quality of lexical tools used for preprocessing. Our results of the lexical preprocessing stages showed that the quality and coverage of Polish taggers are not very good when dealing with more specific texts. In such cases even a small additional dictionary might be necessary to obtain good results. On the other hand, a big common part which economic texts have with general language used in newspapers make the terminology selection stage less precise.

References

Acedański, S. (2010). A morphosyntactic Brill tagger for inflectional languages. In Loftsson, H., Rögnvaldsson, E., and Helgadóttir, S., editors, *Advances in Natural Language Processing*, volume 6233, pages 3–14. Springer Berlin / Heidelberg.

Amigó, E., Gonzalo, J., Artiles, J., and Verdejo, F. (2009). A comparison of extrinsic clustering evaluation metrics based on formal constraints. *Information Retrieval*, 12(5):613.

Bagga, A. and Baldwin, B. (1998). Algorithms for scoring coreference chains. In *The First International Conference on Language Resources and Evaluation Workshop on Linguistics Coreference*, pages 563–566.

Cimiano, P. (2006). *Ontology learning and population from text. Algorithms, evaluation and applications*. Springer.

Fernández, A. and Gómez, S. (2008). Solving non-uniqueness in agglomerative hierarchical clustering using multidendrograms. *Journal of Classification*, 25:43–65.

Frantzi, K., Ananiadou, S., and Mima, H. (2000). Automatic recognition of multi-word terms: the C-value/NC-value method. *Int. Journal on Digital Libraries*, 3:115–130.

Marciniak, M. and Mykowiecka, A. (2012). Terminology extraction from domain texts in Polish. In Bembenik, R., Skonieczny, Ł., Rybiński, H., Kryszkiewicz, M., Niezgódka, M., editors, Intelligent Tools for Building a Scientific Information Platform: Advanced Architectures and Solutions, Springer (in press)

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

Mykowiecka, A. and Marciniak, M. (2012). Clustering of medical terms based on morpho-syntactic features. In Proc. of International Conference on Knowledge Engineering and Ontology Development KEOD 2012, Barcelona. SciTePress Digital Library..

Nenadić, G., Spasić, I., and Ananiadou, S. (2004). Automatic discovery of term similarities using pattern mining. *International Journal of Terminology*, 10(1):55–80.

Pazienza, M., Pennacchiotti, M., and Zanzotto, F. (2005). Terminology extraction: an analysis of linguistic and statistical approaches. In Sirmakessis, S., editor, *Knowledge Mining Series: Studies in Fuzziness and Soft Computing*. Springer Verlag.

Pease, A. and Niles, I. (2001). Towards a standard upper ontology. In *Proceedings of the 2nd International Conference on Formal Ontology in Information Systems (FOIS-2001)*.

Piasecki, M., Szpakowicz, S., and Broda, B. (2009). *A Wordnet from the Ground Up*. Oficyna Wydawnicza Politechniki Wrocławskiej, Wrocław.

Przepiórkowski, A. (2008). *Powierzchniowe przetwarzanie języka polskiego*. Akademicka Oficyna Wydawnicza EXIT, Warsaw.

Przepiórkowski, A., Bańko, M., Górski, R. L., and Lewandowska-Tomaszczyk, B., editors (2012). *Narodowy Korpus Języka Polskiego*. Wydawnictwo Naukowe PWN, Warsaw.

Wermter, J. and Hahn, U. (2005). Massive biomedical term discovery. In *Discovery Science, LNCS 3735*, pages 281–293.

Woliński, M. (2006). Morfeusz — a practical tool for the morphological analysis of Polish. In Kłopotek, M., Wierzchoń, S., and Trojanowski, K., editors, *Intelligent Information Processing and Web Mining, IIS:IIPWM'06 Proceedings*, pages 503–512. Springer.

Alignment by Bilingual Generation and Monolingual Derivation

Toshiaki Nakazawa *Sadao Kurohashi*
Graduate School of Informatics, Kyoto University
Yoshida-honmachi, Sakyo-ku
Kyoto, 606-8501, Japan
nakazawa@nlp.ist.i.kyoto-u.ac.jp, kuro@i.kyoto-u.ac.jp

ABSTRACT
One of the main issues in a word alignment task is the difficulty of handling function words that do not have direct translations which we call unique function words. They are often aligned to some words in the other language incorrectly. This is prominent in language pairs with very different sentence structures. In this paper, we propose a novel approach for handling unique function words. The proposed model *monolingually derives* unique function words from bilingually generated treelet pairs. The monolingual derivation prevents incorrect alignments for unique function words. The derivation probabilities are estimated from a large monolingual corpus, which is much easier to acquire than a parallel corpus. Also, the proposed alignment model uses semantic-head dependency trees where dependency relations between words become similar in each language. Experimental results on an English-Japanese corpus show that the proposed model achieves better alignment and translation quality compared with the baseline models.

TITLE AND ABSTRACT IN JAPANESE

二言語の生成と単言語の派生によるアライメント

単語アライメントタスクにおける主な問題の一つは、機能語の中でも相手言語に対応する語が存在しない機能語の扱いの困難さである。我々はこのような語を孤立機能語と呼ぶ。孤立機能語は、相手言語の何らかの単語に不適切に対応付けられることが多く、これは特に文構造が大きく異なる言語対において顕著である。本論文では、孤立機能語を扱うための新しい手法を提案する。提案モデルは、二言語で生成された部分木ペアから、孤立機能語をそれぞれ単言語で派生することにより、孤立機能語が誤って対応付けられることを防ぐ。派生確率は、対訳コーパスに比べて入手が容易である大規模単言語コーパスから推定する。また提案モデルは、単語同士の依存関係が各言語で近くなるように、意味主辞依存構造木を用いる。英日コーパスでの実験結果から、提案モデルはベースラインモデルと比べてより良いアライメントおよび翻訳精度を実現した。

KEYWORDS: monolingual derivation, semantic-head dependency tree, treelet alignment.

KEYWORDS IN JAPANESE: 単言語の派生, 意味主辞依存構造木, 木構造アライメント.

Proceedings of COLING 2012: Technical Papers, pages 1963–1978,
COLING 2012, Mumbai, December 2012.

1 Introduction

Alignment accuracy is crucial for providing high quality corpus-based machine translation systems because translation knowledge is acquired from an aligned training corpus. For similar language pairs, alignment accuracy is high. Less than 10% alignment error rate (AER) for French-English has been achieved by the conventional word alignment tool GIZA++, an implementation of the alignment models called the IBM models (Brown et al., 1993), with some heuristic symmetrization rules. However, for distant language pairs such as English-Japanese, the conventional alignment method is quite inadequate (achieving an AER of about 20%).

There are two main issues in a word alignment task for distant language pairs: one is the word order difference, while the other relates to function words. The word order issue has to some extent been solved by using word dependency trees in the alignment model (Nakazawa and Kurohashi, 2011). Most of the remaining alignment errors are related to function words such as English articles and Japanese case markers (Wu et al., 2011) because they do not have counterparts in the other language. As an example, most of the errors in Figure 1 are related to function words: "has" and "は (*topic-marker*)" in example (A), and "although", "は (*topic-marker*)", "を (*ACC*)" and "が (*but*)" in example (B).

Several previous works focused on alignment errors of function words. Isozaki et al. (2010) inserted pseudo nodes in English sentences for Japanese function words. Wu et al. (2011) removed the alignment of some function words to effectively acquire translation rules using the underlying word alignment by GIZA++. Nevertheless, these methods for dealing with function words are ad-hoc and based on hand-crafted rules.

In this paper, we propose a novel approach for handling function words. If there is a direct translation for a function word, these words should be aligned with each other. For function words that do not have any counterparts, the conventional model is supposed to align them to NULL, but it does not always work well. They are often aligned to some words incorrectly. In contrast with the conventional model, our model *derives* such function words from content words in their own language. The derivation probabilities used in our proposed model are estimated from a large monolingual corpus for each language. Thus, we do not require a large parallel corpus. With this derivation model, we can reduce alignment errors for function words, which leads to a better translation resources such as a phrase table, which is acquired from a word-aligned parallel corpus. In the remainder of this paper, we use English-Japanese language pairs for explanation. However, it should be noted that the proposed model is completely language independent.

2 Semantic-head dependency tree

The proposed model utilizes word dependency trees on both the source and target sides. Dependency trees are effective for language pairs with very different word orders, such as English-Japanese, to achieve high quality alignment by absorbing the difference (Nakazawa and Kurohashi, 2011). There are two types of word dependency trees: syntactic-head and semantic-head. Our model adopts the latter. This section discusses the difference between syntactic-head and semantic-head, and the reason why we choose the semantic-head dependency tree.

The syntactic-head dependency tree has two main drawbacks. One is that distances between content words are excessively large for agglutinative languages such as Japanese. The other is that dependency relations differ because of the difference in head word definitions in each

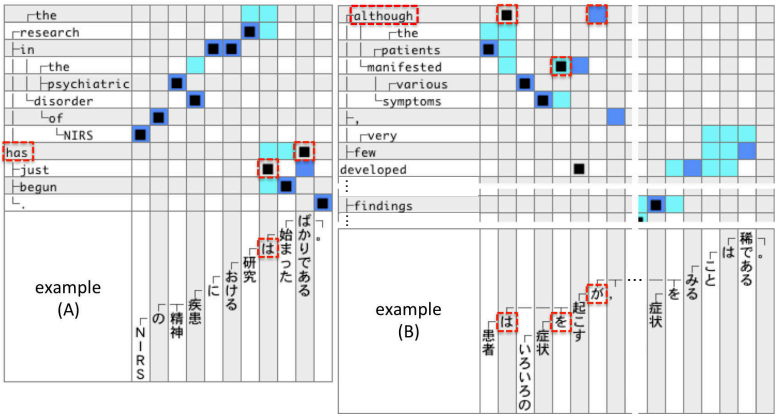


Figure 1: Alignment results of Nakazawa and Kurohashi (2011). Black boxes depict the system output, while dark blue (Sure) and light blue (Possible) cells denote gold-standard alignments. Cells demarcated by dotted lines are alignment errors related to function words.

language. On the other hand, in semantic-head dependency trees, function words giving additional information to content words are placed as children of the content words, thus it preserves the dependency relations between words over languages. In the semantic-head dependency tree (on the right of Figure 2), "medical treatment ↔ 治療", "may ↔ かも しれない", "not ↔ ない" and "weight ↔ 体重" are all children of "change ↔ 変化", while the relations are not preserved in the syntactic-head dependency tree (on the left of Figure 2). Because of these advantages, our model uses semantic-head dependency trees.

In this paper, English sentences are first parsed by nlparser (Charniak and Johnson, 2005) which outputs phrase structures that are then converted into word dependency trees by defining the head word for phrases. The conversion rules follow Collins' head percolation table (Collins, 1999) with some modifications for acquiring semantic-head dependency trees. The following head-specifying rules are examples in which the syntactic head (underlined) and the semantic head (double underlined) is different.

- VP → MD VB (ex. "may change" in Figure 2)
- VP → VBZ JJ (ex. "is large")

Japanese sentences are usually parsed based on a unit called a *basic phrase*, which consists of one content word followed by zero or more function words. In syntactic-head word dependency trees, on the left of Figure 2, the head word is the last function word (or the content word if there is no function word in a basic phrase), and other words depend on their following words (Hajič et al., 2009). In semantic-head dependency trees, while function words showing a relationship between content words such as case markers are placed as parents of content words, other function words are placed as children. We obtain semantic-head dependency trees by modifying the rule file of the Japanese dependency analyzer KNP (Kawahara and Kurohashi, 2006b).

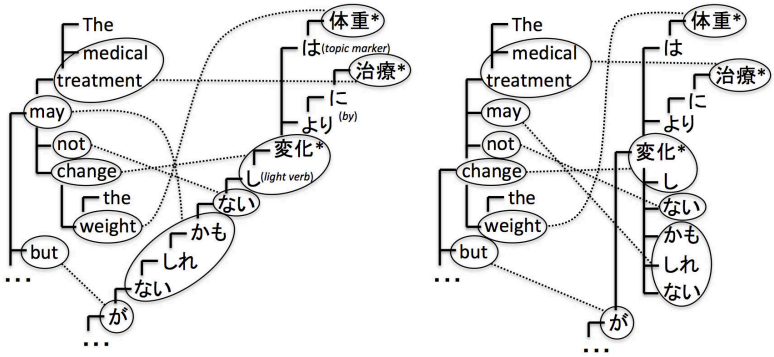


Figure 2: Examples of syntactic-head (left) and semantic-head (right) dependency trees. The root of the tree is at the extreme left and words are placed from top to bottom. Japanese content words are indicated by "*".

Several related studies use the semantic-head dependency trees in machine translation (Hong et al., 2009; Xu et al., 2009). They use English-side semantic-head dependency trees for pre-reordering of English sentences in order to make the word order closer to that of Subject-Object-Verb language sentences. The closer the word order is, the easier it is to train the model. However, certain hand-crafted rules are needed for reordering, and pseudo words are used to take care of function words. Compared with these studies, our proposed model uses semantic-head dependency trees on both sides and does not reorder sentences, and therefore it does not require any hand-crafted rules or pseudo words.

3 Handling function words

Although each language has a closed set of function words, the variety of function words differs between languages. For example, in Figure 2, some function words have direct translations in the other language ("may ↔ かも しれ ない", "not ↔ ない" and "but ↔ が"), while others do not ("the", "は (*topic marker*)" and "に より (*by*)"). The first case is less problematic, and any alignment model can correctly detect the link. The second case represents the issue addressed in this paper. We call function words that do not have direct translations *unique function words*.

One solution, adopted by almost all the existing alignment models, is to align unique function words to NULL. However, it is difficult to judge whether a unique function word has to be NULL-aligned or not, and often causes alignment errors as shown in Figure 1. Also, the NULL-aligned words behave as gaps between aligned words, making it harder to capture relations between aligned words. In Figure 2, "に より (*by*)" is a gap between "治療 (*medical treatment*)" and "変化 し (*change*)" whereas counterparts have a direct parent-child relation.

Another solution is to enhance the alignment model so as to handle a larger unit than a word, and include the unique function words in neighboring alignments. For example, "weight ↔ 体重" can include the Japanese "は (*topic marker*)" and become "weight ↔ 体重 は". However, this solution can lead to a less appropriate parameter estimation for an alignment model because "weight ↔ 体重" and "weight ↔ 体重 は" are treated as different alignment patterns

while the two patterns have essentially no difference.

The novel solution to this problem proposed in this paper is to *derive* unique function words from neighboring words monolingually. In the monolingual derivation model, "は (*topic marker*)" can be derived from "体重 (*weight*)" without changing the original treelet pair "weight ⟷ 体重", and therefore the model achieves good estimation of parameters. Note that it is also possible to derive "は (*topic marker*)" from "変化 し (*change*)" because they are contiguous in the tree structure. Another advantage of the monolingual derivation model is that it can reduce the gaps between alignments, and preserve the dependency relations between treelets over languages. For example, the model can derive "に より (*by*)" from "変化 し (*change*)" or "治療 (*medical treatment*)" and let the dependency relation between "変化 し (*change*)" and "治療 (*medical treatment*)" be direct parent-child like their English counterparts. This is effective for estimating the dependency relation probability described in Section 4.3.

4 Model overview

The proposed model is an extension of that proposed by Nakazawa and Kurohashi (2011). This earlier model was overcoming the long-distance reordering issue by incorporating dependency trees. However, it was suffering from alignment errors for function words, which our new model solves by incorporating a *monolingual derivation model*. First we describe the generative story for the joint alignment model in the same manner as in previous work (Marcu and Wong, 2002; DeNero et al., 2008; Nakazawa and Kurohashi, 2011).

1. Generate ℓ concepts from which bilingual treelet pairs are generated independently.
2. For each treelet pair, derive zero or more treelets monolingually from each treelet in the treelet pair.
3. Combine the treelets in each language so as to create parallel sentences.

The number of concepts ℓ (> 0) is parameterized using a geometric distribution

$$P(\ell) = p_\$ \cdot (1 - p_\$)^{\ell - 1} \tag{1}$$

where $p_\$$ is a constant. Each concept generates a bilingual treelet pair from an unknown distribution θ_T. We call the treelet pair the *core alignment* denoting it as $\langle e_C, f_C \rangle$. Either one of the treelets in a treelet pair can be NULL, which represents an unaligned treelet. Unaligned treelets must be composed of exactly one word (NULL-alignment restriction).

Each treelet e_C and f_C derives sets of treelets $\{d_{e_C}\}$ and $\{d_{f_C}\}$ monolingually which basically consist of unique function words. The numbers of monolingual derivations $|\{d_{e_C}\}|$ and $|\{d_{f_C}\}|$ (≥ 0) are parameterized using a geometric distribution

$$P(|\{d_{e_C}\}|) = p_d \cdot (1 - p_d)^{|\{d_{e_C}\}|}, \quad P(|\{d_{f_C}\}|) = p_d \cdot (1 - p_d)^{|\{d_{f_C}\}|} \tag{2}$$

where p_d is a constant. Each derivation is drawn from a known multinomial distribution ϕ_{e_C} and ϕ_{f_C}, as explained in Section 4.2. We use the notation e to represent the combination of e_C and $\{d_{e_C}\}$, and f for f_C and $\{d_{f_C}\}$. Thus $\langle e, f \rangle$ contains $\langle e_C, f_C \rangle$, $\{d_{e_C}\}$ and $\{d_{f_C}\}$.

Finally, the treelet pairs are combined in each language. We denote the relation of treelets on the e-side as $D_E = \{(j \rightarrow k)\}$, where $(j \rightarrow k)$ denotes that treelet e_j depends on treelet e_k, and on the f-side as D_F. D refers to D_E and D_F as a whole. With these notations, the joint

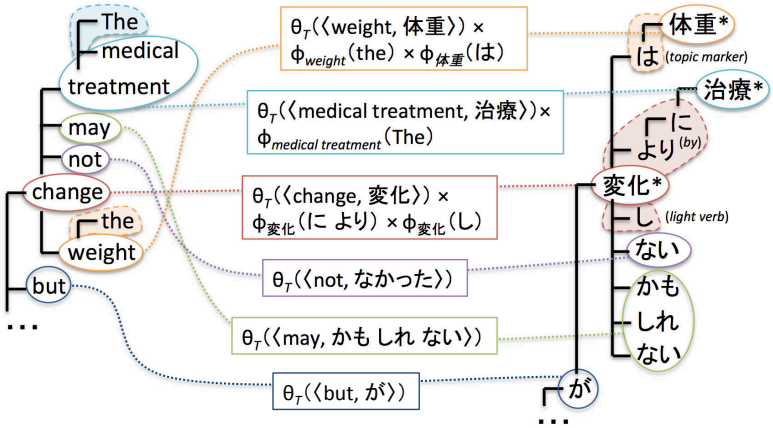


Figure 3: Example showing the calculation of the bilingual generation probability and monolingual derivation probability.

probability for an aligned sentence pair is defined as:

$$P(\ell, \{\langle e, f\rangle\}, D) = P(\ell) \cdot P(D|\{\langle e, f\rangle\}) \cdot \prod_{\langle e,f\rangle} \left(\theta_T(\langle e_C, f_C\rangle) \cdot P(|\{d_{e_C}\}|) \cdot \prod_{d_{e_C}} \phi_{e_C}(d_{e_C}) \cdot P(|\{d_{f_C}\}|) \cdot \prod_{d_{f_C}} \phi_f(d_{f_C}) \right). \tag{3}$$

In Figure 3, we show an example of the calculation of bilingual generation probability and monolingual derivation probability for each treelet pair (ignoring $P(|\{d_{e_C}\}|)$ and $P(|\{d_{f_C}\}|)$ for ease of understanding). For the core alignment ⟨not, ない⟩, there is no derivation, and only bilingual generation probability $\theta_T(\langle not, \text{なかった}\rangle)$ is used. For ⟨change, 変化⟩, there are two derivations from "変化 (*change*)"; "に より (*by*)" and "し (*light verb*)". Therefore, we need to calculate two monolingual derivation probabilities in addition to the bilingual generation probability.

The remainder of this section gives the details of the bilingual generation probability θ_T, monolingual derivation probability ϕ and dependency relation probability $P(D)$.

4.1 Bilingual generation probability

When generating bilingual treelets, we first need to decide whether to generate an unaligned treelet (with probability p_N) or an aligned treelet pair (with probability $1-p_N$). Aligned treelet pairs are generated from an unknown probability distribution θ_A, which obeys the Dirichlet process (DP):

$$\theta_A(\langle e_C, f_C\rangle) \quad \sim \quad DP(M_A, \alpha_A), \tag{4}$$

where M_A is the base distribution and α_A is a concentration parameter. The base distribution is defined as:

$$M_A(\langle e_C, f_C\rangle) = [P_e(e_C)P_{WA}(f_C|e_C) \cdot P_f(f_C)P_{WA}(e_C|f_C)]^{\frac{1}{2}}$$
$$P_e(e_C) = p_t \cdot (1-p_t)^{|e|-1} \cdot \left(\frac{1}{n_e}\right)^{|e_C|} \quad P_f(f_C) = p_t \cdot (1-p_t)^{|f|-1} \cdot \left(\frac{1}{n_f}\right)^{|f_C|}, \tag{5}$$

where P_{WA} is the translation probability computed by IBM model1 (Brown et al., 1993), and n_e and n_f are the numbers of word types in each language. θ_A does not give a weight to an unaligned treelet.

Unaligned treelets are generated from another unknown probability distribution θ_N:

$$\begin{aligned} \theta_N(\langle e_C, f_C\rangle) &\sim DP(M_N, \alpha_N) \\ M_N(\langle e_C, f_C\rangle) &= \begin{cases} P_{WA}(e_C|\text{NULL}) & \text{if } f_C = \text{NULL} \\ P_{WA}(f_C|\text{NULL}) & \text{if } e_C = \text{NULL} \end{cases}. \end{aligned} \tag{6}$$

θ_N does not give a weight to an aligned treelet pair. Note that an unaligned treelet is always composed of only one word in our model. Finally, θ_T can be decomposed as:

$$\theta_T(\langle e_C, f_C\rangle) = p_N\theta_N(\langle e_C, f_C\rangle) + (1-p_N)\theta_A(\langle e_C, f_C\rangle). \tag{7}$$

The earlier study (Nakazawa and Kurohashi, 2011) only considered treelets as alignment units. However, this is inadequate for semantic-head dependency trees, since a set of sibling function words is often considered as an alignment unit. In Figure 2, for example, sibling "かも しれ ない (*may*)" in Japanese should be aligned to the English "may". Therefore, our model allows siblings to be a core alignment unit when the siblings are contiguous in the word sequence. We suppose the term "treelet" includes siblings in this paper.

4.2 Monolingual derivation probability

We only explain the *e*-side derivations in this section, since the *f*-side is the same. We calculate the monolingual derivation probability using a large monolingual corpus. Derivations d_{e_C} are conditioned on the treelet e_C from which they were derived:

$$\phi_e(d_{e_C}) = p(d_{e_C}|e_C). \tag{8}$$

For example, $\phi_{medical\ treatment}(\text{the}) = p(\text{the}|\text{medical treatment})$. However, using a treelet as a condition is vulnerable to the data sparseness problem. We use an *anchor word* in e_C as the condition instead of e_C. The derivation is connected to the anchor word in the word dependency tree:

$$p(d_{e_C}|e_C) \approx p(d_{e_C}|A(e_C, d_{e_C})). \tag{9}$$

The function $A(e_C, d_{e_C})$ returns the anchor word in e_C for d_{e_C}. For example, $A(\text{medical treatment, the})$ is "treatment".

$p(d_{e_C}|A(e_C, d_{e_C}))$ are calculated as follows:

$$p(d_{e_C}|A(e_C, d_{e_C})) = \frac{Count(d_{e_C}, A(e_C, d_{e_C}))}{\sum_d Count(d, A(e_C, d_{e_C}))}. \tag{10}$$

Anchor	Derivations
the	P:treatment, P:treatment change
medical	P:treatment, P:treatment change
treatment	L:the, L:medical, P:change
may	P:change
not	P:change
change	L:treatment, L:the treatment, L:medical treatment, L:the medical treatment, L:may, L:not, R:weight, R: the weight
the	P:weight, P: change weight
weight	P:change, L: the

Anchor	Derivations
体重	P:は, P:は 変化
は	L:体重, P:変化
治療	P:に, P:に より, P:に より 変化
に	L:治療, P:より, P:より 変化
より	L:に, L:治療 に, P:変化
変化	L:は, L:体重 は, L:より, L:に より, L:治療 に より, R:し, R:ない, R:かも, R:しれ, R:ない
し	P:変化
ない	P:変化

Table 1: The English derivations (left) and Japanese derivations (right) acquired from the sentences in Figure 3. 'P', 'L' and 'R' denote, respectively, Parent, pre-child (dependent from the Left), and post-child (dependent from the Right).

$Count(d_{e_C}, A(e_C, d_{e_C}))$ denotes the frequency with which d_{e_C} is connected to $A(e_C, d_{e_C})$ in the monolingual corpus. Taking each sentence in Figure 3 as an example sentence in the monolingual corpus, we can enumerate the derivations shown in Table 1 from the sentences. A derivation must be contiguous as a tree, and we do not consider sibling derivations. We distinguish three types of derivations: parent, pre-child (dependent from the left) and post-child (dependent from the right).

This lexicalized derivation is excessively specific. For example, the highest probability derivations from "Ph.D." acquired from the English Web corpus (Kawahara and Kurohashi, 2006a) are "a", "student", "thesis" in order. Consequently, using only lexicalized derivation can cause many derivation errors. We consider not only the lexicalized derivation probability, but also another probability using part-of-speech (POS) is used as the condition. Using the notation $A_{pos}(e_C, d_{e_C})$ for the POS of the anchor word, the monolingual derivation probability is defined as:

$$p(d_{e_C}|e_C) = [p(d_{e_C}|A(e_C, d_{e_C})) \cdot p(d_{e_C}|A_{pos}(e_C, d_{e_C}))]^{\frac{1}{2}}. \quad (11)$$

$p(d_{e_C}|A_{pos}(e_C, d_{e_C}))$ is also acquired from the large monolingual corpus in the same manner as $p(d_{e_C}|A(e_C, d_{e_C}))$. We take the geometric mean of the two probabilities because this eliminates noisy derivations of lexicalized probabilities while keeping the derivation preferences for each word.

Note that we do not need to discriminate between content words and function words in the enumeration of derivations and the calculation of derivation probabilities. Generally, the neighboring words of function words are content words. The vocabulary size of content words is much larger than that of function words. Therefore, the number of derivation patterns from function words is quite large, causing their probabilities to be very small. The probabilities naturally prefer deriving function words from content words than deriving content words from function words.

4.3 Dependency Relation Probability

Our model considers dependency relations between treelets and assigns a weight to each relation following the previous work (Nakazawa and Kurohashi, 2011). Here, each treelet includes both core and derivation treelets, and treelets in a treelet pair have the same index, for example, the counterpart of e_j is f_j.

The dependency relations are considered on each e and f side in the same manner, thus we only explain the e-side. First, we find the nearest aligned parent treelet, which we call relational parent, for each treelet in e-side. The relational parent is searched by ascending the dependency tree to the root node until an aligned treelet is found. The number of unaligned treelets on the path to relational parent from e_j is denoted as $N(e_j)$. $N(e_j) = 0$ if the relational parent is the direct parent of e_j. We consider an imaginary root as the relational parent for the root treelet of a sentence.

Suppose the relational parent treelet of e_j is e_k. Then, we consider where their counterparts, f_j and f_k respectively, are on the dependency tree of the other side. We can assume that f_j tends to depend on f_k because the dependencies between concepts hold across languages. The dependency relation probability reflects this tendency. We define the function $rel(e_j, e_k)$ which returns a dependency relation between the counterparts of the two arguments, in other words, dependency relation between f_j and f_k. We express a dependency relation as the shortest path from f_j to f_k. For simplicity, we indicate the path with a pair of non-negative integers, where the first is the number of steps going up (Up) the dependency tree and the other is the number going down ($Down$). For example, in Figure 3, traveling from "medical treatment" to "weight" requires 1 step going up (to reach "change") and 1 step going down, so the dependency relation is $(Up, Down) = (1, 1)$.

Finally, we assign the dependency relation probability to a triplet of non-negative integers $R_f = (N, Up, Down)$. The dependency relation probabilities for the e-side are drawn from an unknown probability distribution θ_{ef} and for the f-side from θ_{fe}, with both obeying the DP:

$$\begin{array}{rclcrcl} \theta_{ef}(R_e) & \sim & DP(M_{ef}, \alpha_{ef}) & & M_{ef}(R_e) & = & p_{ef} \cdot (1-p_{ef})^{N+Up+Down-1} \\ \theta_{fe}(R_f) & \sim & DP(M_{fe}, \alpha_{fe}) & & M_{fe}(R_f) & = & p_{fe} \cdot (1-p_{fe})^{N+Up+Down-1}. \end{array} \tag{12}$$

Using the notations and definitions above, the dependency tree-based reordering model $P(D|\{\langle e, f\rangle\})$ is decomposed as:

$$P(D|\{\langle e, f\rangle\}) = \prod_{\langle e,f\rangle} \theta_{ef}(R_e) \cdot \theta_{fe}(R_f). \tag{13}$$

5 Model training

We train the model by means of a collapsed Gibbs sampling, which has been used in some recent NLP works (Nakazawa and Kurohashi, 2011; DeNero et al., 2008). In a Gibbs sampling, we first need to initialize the states of the training data, such as the boundaries between treelets and their alignments, and also initialize the latent variables according to the initial states of the data. Starting with the initial state, we generate many samples sequentially from the last state by changing a small local point. Normalizing the counts in the samples yields the parameter estimations.

5.1 Initialization

We initialize the states of the training data by heuristically merging bi-directional alignment results of the standard word alignment tool GIZA++. Many machine translation studies use heuristics to combine the two alignment results, one of which is called grow-diag-final-and (Koehn et al., 2007). Our heuristic is similar to this, but the difference is that we combine the two results based on dependency trees, and not on word sequences. The initialization is carried out by the following steps:

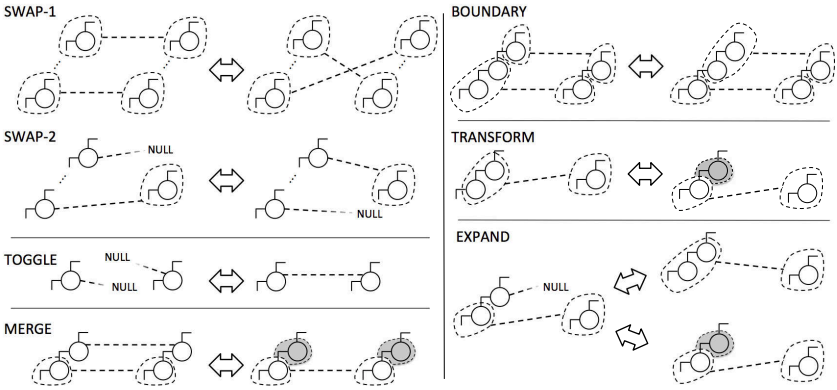


Figure 4: Illustration of the sampling operators. A solid circle represents a single word, while a treelet is depicted surrounded by a broken line. A gray treelet represents a derivation. A link directly connected to a word denotes that the treelet must consist of exactly one word, whereas other treelets can consist of one or more words including derivations.

1. Take the intersection of the two results.
2. In the union of the two results, accept alignment points connected to at least one accepted point in terms of the dependency tree (corresponds to grow-diag).
3. In the union of the two results, accept alignment points between two unaligned words (corresponds to final-and).

Initial boundaries of treelets and their alignments, and also the counts of treelet pairs and dependency relations are thus acquired. Note that there is no derivations after the initialization step.

5.2 Sampling operators

Our sampler repeatedly uses the six operators illustrated in Figure 4, to generate samples. Each application of an operator generates one new sample. We could, of course, use all the generated samples. However, since successive samples are almost the same, except for one local part, it is futile keeping all the samples. Thus, for each iteration, we keep only one sample, which is the final outcome after applying all the operators to all the possible points in all the sentence pairs in the training corpus.

SWAP
The SWAP operator exchanges the counterparts of two treelets, which may have derivations. There are two cases: [SWAP-1] both treelets are aligned, and [SWAP-2] one of the two treelets is unaligned and the other consists of exactly one word.

TOGGLE
The TOGGLE operator adds or removes an alignment. If f_j and e_k are both unaligned treelets, TOGGLE links the two treelets. Alternatively, if f_j and e_j are aligned, TOGGLE cuts the link

and makes each of the treelets unaligned. Because of the NULL-alignment restriction, f_j and e_j must consist of exactly one word.

MERGE
The MERGE operator combines a one-to-one alignment with the neighboring alignment as derivations from each treelet, or separates derivations from each treelet as an independent alignment.

BOUNDARY
The BOUNDARY operator moves the boundary between two treelets by one word. This operator does not change the type (core or derivation) of the boundary word.

TRANSFORM
The TRANSFORM operator changes the type of a word from core to derivation or vice versa.

EXPAND
The EXPAND operator expands or contracts an aligned treelet. If an unaligned treelet is next to an aligned one, EXPAND merges the unaligned and aligned treelets, either as a part of core treelet or derivation treelet. As the opposite direction, it excludes a marginal node from a treelet, and to make the excluded node unaligned.

6 Alignment experiments

We conducted alignment experiments on the English-Japanese corpus to show the effectiveness of the proposed model.

6.1 Settings

For the experiments, we used the JST[1] paper abstract corpus. This corpus was created by NICT[2] from JST's 2M English-Japanese paper abstract corpus using the method of Utiyama and Isahara (2007). This corpus consists of 996K parallel sentences: 24.7M words in English and 27.5M words in Japanese. Unfortunately, this corpus is not publicly available now, but they will become available in the near future.

As gold-standard data, 500 sentence pairs were annotated by hand using two types of annotations: sure (S) alignments and possible (P) alignments (Och and Ney, 2003). The unit of evaluation was the word. We used precision, recall, and alignment error rate (AER) as evaluation criteria. All the experiments were run on the original forms of words. The hyper parameters for our model used in the experiments are as follows: $p_\$ = 0.1$, $p_d = 0.9$, $p_N = 0.1$, $p_t = 0.8$, $\alpha_A = 100$, $\alpha_N = 100$, $\alpha_{fe} = 100$, $\alpha_{ef} = 100$, $p_{fe} = 0.5$, $p_{ef} = 0.5$. They are borrowed from the previous work (DeNero et al., 2008; Nakazawa and Kurohashi, 2011) and changed a little. The training time was about 1 day using 200 CPU cores. It is much slower than the word-sequence-based models because considering tree structures is computationally more complex.

The derivation probabilities were calculated from English and Japanese Web corpora each consisting of 550M sentences (Kawahara and Kurohashi, 2006a). We limited the maximum size of a derivation treelet to three words. We only consider top-20 frequent derivations for each word and POS.

[1] http://www.jst.go.jp/
[2] http://www.nict.go.jp/

English sentences were converted into phrase structures using Charniak's nlparser (Charniak and Johnson, 2005), and then they were transformed into dependency structures by rules defining head words for phrases (Collins, 1999). Japanese sentences were converted into dependency structures using the morphological analyzer JUMAN (Kurohashi et al., 1994) and the dependency analyzer KNP (Kawahara and Kurohashi, 2006b).

For comparison, we used GIZA++ (Och and Ney, 2003), which implements the well-known word-based statistical alignment model of the IBM Models. We conducted word alignment bidirectionally with the default parameters and merged them using the grow-diag-final-and heuristics (Koehn et al., 2003). We also tested the BerkeleyAligner[3] (DeNero and Klein, 2007) in the unsupervised training mode with default settings.

6.2 Experimental result and discussion

The experimental results are given in Table 2. "Syntactic-head" is the alignment accuracy of the baseline system by Nakazawa and Kurohashi (2011), while "Semantic-head w/o derivation" is the result of using the baseline model on semantic-head dependency trees. The results of incorporating the monolingual derivation are given in the bottom two rows, where "all" means that we evaluated all the alignments including derivations, while "core" means that we only evaluated the core alignments.

As mentioned in Section 1, the baseline model has already shown much better alignment accuracy than the conventional models, GIZA++ and BerkeleyAligner. There was a slight improvement using semantic-head dependency trees (0.36% absolute AER reduction).

The proposed model further improved the alignment accuracy. Compared with the baseline model, we achieved 0.6% and 1.34% improvement in absolute AER by evaluating all the alignments and only the core alignments respectively. The relative error reduction in AER is about 10% for the core alignment, which can be considered as a significant improvement. The reason of the further AER decrease when using only the core alignments is as follows: although the monolingual derivation can prevent from incorrect alignments for unique function words, it sometimes causes over derivations. This is discussed in detail later.

Figure 5 shows the alignment results by the proposed model for sentences in Figure 1. The proposed model reduced the alignment errors for unique function words by deriving them monolingually, and found correct alignments which the baseline system failed to find.

There are two main causes of alignment errors in the proposed model. One is the granularity of the derivation probability. We used the product of the two probabilities, lexicalized and POS-based, to take advantage of them, but this is insufficient. For example, in the sentence fragment "the possibility that ...", the proposed model failed to derive the unique word "that" from "possibility". In another fragment "the patient who ...", the proposed model failed to derive "who" from "patient". The reason for these failures is the low probability in the POS-based derivation. The possible solution for the granularity problem is to use word classes where each word class contains the words which have similar derivation distribution. For example, some abstract nouns such as "possibility" and "fact" tend to derive appositive "that", and person-category nouns tend to derive a relative "who".

The other cause of alignment errors is the over derivation typically created by the noise in a parallel sentence and parsing error. On the left of Figure 6, the fragment of the Japanese

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

Alignment model	Pre.	Rec.	AER
GIZA++ & grow-diag-final-and	83.00	83.01	16.99
BerkeleyAligner	85.69	74.13	20.32
Syntactic-head (baseline)	88.59	83.78	13.66
Semantic-head w/o derivation	88.82	84.28	13.30
Semantic-head w/ derivation (all)	87.83	**85.81**	13.06
Semantic-head w/ derivation (core)	**90.51**	84.49	**12.32**

Table 2: Results of English-Japanese alignment experiments.

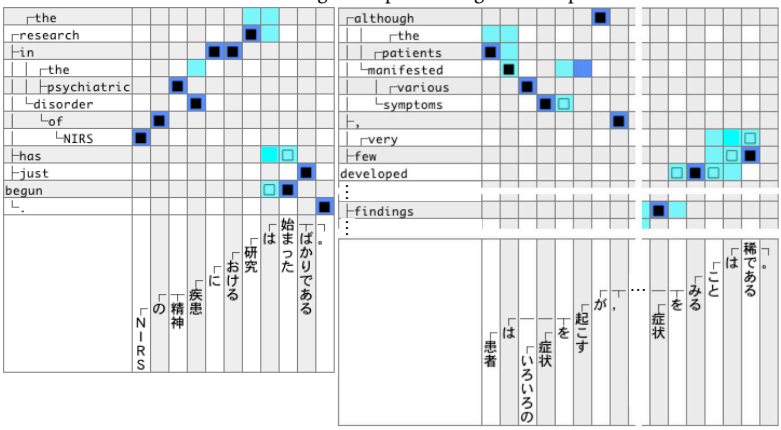


Figure 5: Alignment results by the proposed model. The solid and open squares depict core alignments and derivations, respectively.

sentence “音 列 の (*of tone sequence*)” has no corresponding part in the English sentence. “音 列 (*tone sequence*)” is correctly aligned to NULL, but the function word “の (*of*)” is incorrectly derived from “法 (*method*)”. On the right of Figure 6, the Japanese “HDMI は” should not depend on “異なる (*different*)”, but on “使用 (*use*)”. Because of this parsing error, the topic marker “は” is incorrectly derived from “異なる (*different*)”. One possible short-term solution for the parsing problem is to use the n-best parsing results in the model. An alternative solution was proposed by Burkett et al. (2010), who described a joint parsing and alignment model that exchanges useful information between the parser and aligner.

7 Translation experiments

We conducted English-to-Japanese and Japanese-to-English translation experiments on the same corpus used in the alignment experiments. We translated 500 paper abstract sentences from the JST corpus. Note that these sentences were not included in the training corpus. We use Joshua[4], a Java-based opensource implementation of the hierarchical decoder, version 4.0 (Ganitkevitch et al., 2012) with default settings. It was tuned using another 500 development sentence pairs.

Table 3 shows the BLEU (Papineni et al., 2002) scores for the translations. The proposed

[4]http://joshua-decoder.org

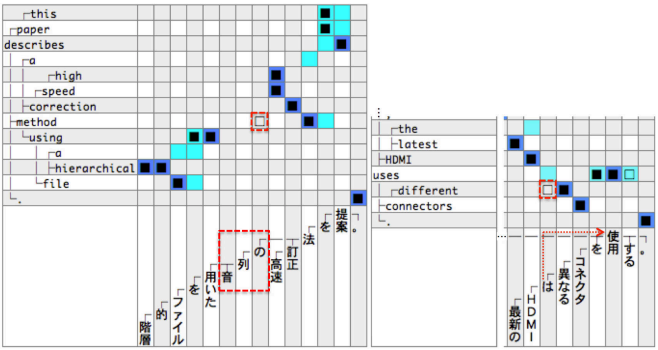


Figure 6: Alignment errors of the proposed model caused by a NULL part (left) and a parsing error (right).

Alignment model	En → Ja	Ja → En
GIZA++ & grow-diag-final-and	23.84	17.75
Syntactic-head (baseline)	24.16	17.83
Semantic-head w/o derivation	24.11	18.06
Semantic-head w/ derivation (all)	**24.55**†	**18.46**†‡
Semantic-head w/ derivation (core)	24.45	17.76

Table 3: BLEU scores for English-to-Japanese and Japanese-to-English translation experiments. † and ‡ marks indicate significant difference by bootstrap resampling (Koehn, 2004) from the decoder using GIZA++ & grow-diag-final-and alignment and baseline alignment respectively ($p < 0.05$).

model using all the alignments including derivations achieved the best translation quality. We believe this improvement is due to the reduction in function word alignment errors. The BLEU score decreased when only core alignments were used. This is because the exclusion of the derivations increased the ambiguity of the translation rules.

Conclusion and future work

In this paper, we proposed a novel approach for handling unique function words based on semantic-head dependency trees. The proposed model monolingually derives unique function words from bilingually generated treelet pairs. The derivation probabilities are acquired from a large monolingual corpus for each language. We showed that semantic-head dependency trees are more effective than syntactic-head dependency trees for high quality alignment, and that the treelet derivation model can reduce alignment errors for function words resulting in better translation quality.

To further validate the effectiveness of the proposed model, we need to apply our model to other language pairs, including the Korean language, which is also an agglutinative language. In addition, we need to resolve the issues discussed in Section 6.2.

Acknowledgments

This work was partially supported by Yahoo Japan Corporation.

References

Brown, P. F., Pietra, S. A. D., Pietra, V. J. D., and Mercer, R. L. (1993). The mathematics of statistical machine translation: Parameter estimation. *Association for Computational Linguistics*, 19(2):263–312.

Burkett, D., Blitzer, J., and Klein, D. (2010). Joint parsing and alignment with weakly synchronized grammars. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 127–135, Los Angeles, California. Association for Computational Linguistics.

Charniak, E. and Johnson, M. (2005). Coarse-to-fine n-best parsing and maxent discriminative reranking. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 173–180.

Collins, M. (1999). *Head-Driven Statistical Models for Natural Language Parsing*. PhD thesis, University of Pennsylvania.

DeNero, J., Bouchard-Côté, A., and Klein, D. (2008). Sampling alignment structure under a Bayesian translation model. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 314–323, Honolulu, Hawaii. Association for Computational Linguistics.

DeNero, J. and Klein, D. (2007). Tailoring word alignments to syntactic machine translation. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 17–24, Prague, Czech Republic. Association for Computational Linguistics.

Ganitkevitch, J., Cao, Y., Weese, J., Post, M., and Callison-Burch, C. (2012). Joshua 4.0: Packing, pro, and paraphrases. In *Proceedings of the Seventh Workshop on Statistical Machine Translation*, pages 283–291, Montréal, Canada. Association for Computational Linguistics.

Hajič, J., Ciaramita, M., Johansson, R., Kawahara, D., Martí, M. A., Màrquez, L., Meyers, A., Nivre, J., Padó, S., Štěpánek, J., Straňák, P., Surdeanu, M., Xue, N., and Zhang, Y. (2009). The conll-2009 shared task: Syntactic and semantic dependencies in multiple languages. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL 2009): Shared Task*, pages 1–18, Boulder, Colorado. Association for Computational Linguistics.

Hong, G., Lee, S.-W., and Rim, H.-C. (2009). Bridging morpho-syntactic gap between source and target sentences for english-korean statistical machine translation. In *Proceedings of the ACL-IJCNLP 2009 Conference Short Papers*, pages 233–236, Suntec, Singapore. Association for Computational Linguistics.

Isozaki, H., Sudoh, K., Tsukada, H., and Duh, K. (2010). Head finalization: A simple reordering rule for sov languages. In *Proceedings of the Joint Fifth Workshop on Statistical Machine Translation and MetricsMATR*, pages 244–251, Uppsala, Sweden. Association for Computational Linguistics.

Kawahara, D. and Kurohashi, S. (2006a). Case frame compilation from the web using high-performance computing. In *the 5th International Conference on Language Resources and Evaluation (LREC2006)*.

Kawahara, D. and Kurohashi, S. (2006b). A fully-lexicalized probabilistic model for japanese syntactic and case structure analysis. In *Proceedings of the Human Language Technology Conference of the NAACL, Main Conference*, pages 176–183, New York City, USA. Association for Computational Linguistics.

Koehn, P. (2004). Statistical significance tests for machine translation evaluation. In Lin, D. and Wu, D., editors, *Proceedings of EMNLP 2004*, pages 388–395, Barcelona, Spain. Association for Computational Linguistics.

Koehn, P., Hoang, H., Birch, A., Callison-Burch, C., Federico, M., Bertoldi, N., Cowan, B., Shen, W., Moran, C., Zens, R., Dyer, C., Bojar, O., Constantin, A., and Herbst, E. (2007). Moses: Open source toolkit for statistical machine translation. In *Annual Meeting of the Association for Computational Linguistics (ACL), demonstration session*.

Koehn, P., Och, F. J., and Marcu, D. (2003). Statistical phrase-based translation. In *HLT-NAACL 2003: Main Proceedings*, pages 127–133.

Kurohashi, S., Nakamura, T., Matsumoto, Y., and Nagao, M. (1994). Improvements of Japanese morphological analyzer JUMAN. In *Proceedings of The International Workshop on Sharable Natural Language*, pages 22–28.

Marcu, D. and Wong, D. (2002). A phrase-based,joint probability model for statistical machine translation. In *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing*, pages 133–139. Association for Computational Linguistics.

Nakazawa, T. and Kurohashi, S. (2011). Bayesian subtree alignment model based on dependency trees. In *Proceedings of the 5th International Joint Conference on Natural Language Processing (IJCNLP2011)*, pages 794–802, Chiang Mai, Thailand. Asian Federation of Natural Language Processing.

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

Papineni, K., Roukos, S., Ward, T., and Zhu, W.-J. (2002). Bleu: a method for automatic evaluation of machine translation. In *ACL*, pages 311–318.

Utiyama, M. and Isahara, H. (2007). A japanese-english patent parallel corpus. In *MT summit XI*, pages 475–482.

Wu, X., Matsuzaki, T., and Tsujii, J. (2011). Effective use of function words for rule generalization in forest-based translation. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 22–31, Portland, Oregon, USA. Association for Computational Linguistics.

Xu, P., Kang, J., Ringgaard, M., and Och, F. (2009). Using a dependency parser to improve smt for subject-object-verb languages. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 245–253, Boulder, Colorado. Association for Computational Linguistics.

Optimizing for Sentence-Level BLEU+1 Yields Short Translations

Preslav NAKOV Francisco GUZMÁN Stephan VOGEL
Qatar Computing Research Institute, Qatar Foundation
Tornado Tower, floor 10, P.O. Box 5825, Doha, Qatar
{pnakov,fherrera,svogel}@qf.org.qa

ABSTRACT
We study a problem with *pairwise ranking optimization* (*PRO*): that it tends to yield too short translations. We find that this is partially due to the inadequate smoothing in PRO's BLEU+1, which boosts the precision component of BLEU but leaves the brevity penalty unchanged, thus destroying the balance between the two, compared to BLEU. It is also partially due to PRO optimizing for a sentence-level score without a global view on the overall length, which introducing a bias towards short translations; we show that letting PRO optimize a *corpus-level* BLEU yields a perfect length. Finally, we find some residual bias due to the interaction of PRO with BLEU+1: such a bias does not exist for a version of MIRA with sentence-level BLEU+1. We propose several ways to fix the length problem of PRO, including smoothing the brevity penalty, scaling the effective reference length, grounding the precision component, and unclipping the brevity penalty, which yield sizable improvements in test BLEU on two Arabic-English datasets: IWSLT (+0.65) and NIST (+0.37).

KEYWORDS: Statistical machine translation, parameter optimization, MERT, PRO, MIRA.

Proceedings of COLING 2012: Technical Papers, pages 1979–1994,
COLING 2012, Mumbai, December 2012.

1 Introduction

Early work on statistical machine translation (SMT) has relied on generative training using maximum likelihood parameter estimation. This was inspired by the noisy channel model (Brown et al., 1993), which asked for calculating the product of two components, a language model and a translation model, giving them equal weights. As mainstream research has moved towards combining multiple scores, the field has switched to discriminative tuning in a log-linear fashion. The standard approach has been to maximize BLEU (Papineni et al., 2002) on a tuning dataset using a coordinate descent optimization algorithm known as *minimum error rate training* (*MERT*), as proposed by Och (2003).

MERT has dominated the SMT field for years, until the number of parameters in the log-linear model has gradually increased, in some cases to hundreds and even to hundreds of thousands of scores, which has called for new tuning algorithms since MERT was unable to scale beyond just a handful of parameters. Many alternatives to MERT have been proposed over the years, but it is only recently that some of them have gained popularity in the community, most notably, the *margin infused relaxed algorithm* (MIRA) (Chiang et al., 2008) and *pairwise ranking optimization* (*PRO*) (Hopkins and May, 2011).

While the number of parameters that an optimizer can handle has become a major concern recently, there are many other important aspects that researchers have paid attention to, e.g., the performance of parameters when translating unseen test data, the speed of convergence, the stability across multiple reruns, the objective function being optimized (e.g., BLEU vs. an approximation of BLEU), the mode of learning (e.g., online vs. batch).

Here we study a different, and so far neglected, aspect: the characteristics of the translations generated using weights found by different optimizers. More specifically, we focus on the length with respect to the reference translations. It has been observed that while optimizers like MERT and MIRA typically yield translation hypotheses with the right level of verbosity, other optimizers such as PRO tend to generate translations that are too short.[1] This phenomenon has not been analyzed in sufficient detail so far; yet, it is important since generating short translation hypotheses is penalized by BLEU, which could mean missed opportunities for better BLEU scores. Our contributions can be summarized as follows:

- We analyze the length issue with PRO in detail, we explore possible reasons, and we propose several fixes. We achieve small but consistent improvements in BLEU for fixes that yield longer hypotheses: up to 0.65 BLEU points absolute. We also find that the better a fix approximates the target length, the higher the testing BLEU score.
- We find that the core of the problem is the function being optimized: sentence-level BLEU+1 yields short translations, while corpus-level BLEU yields the right verbosity.
- We demonstrate that what matters is the objective function, not the optimization algorithm. We show that similar translations – in terms of BLEU score and length – can be generated using weights from different optimizers, e.g., PRO vs. MIRA, when they are given the same objective function, e.g., sentence-level vs. corpus-level BLEU.

The remainder of this paper is organized as follows: Section 2 describes related work, Section 3 explains the length issue with PRO and several methods we propose to fix it, Section 4 describes our experiments and evaluation results, and Section 5 offers a more general analysis and discussion. Finally, Section 6 concludes with directions for future work.

[1] That is why the Moses toolkit has an option to run a few iterations of MERT after PRO – to get the length right.

2 Related Work

The dominant approach for parameters optimization in SMT is to use MERT (Och, 2003), a batch tuning algorithm that iterates between two modes: (i) generating a k-best list of translation hypotheses using the current parameters values, and (ii) parameter optimization using the k-best lists from all previous iterations. MERT optimizes expected BLEU. It works well for a small number of parameters, but suffers from scalability and stability issues (Foster and Kuhn, 2009). Most importantly for our discussion, it tends not to have length biases; this is also confirmed by our own experiments (see Table 4).

Various alternatives to MERT have been proposed, motivated primarily by scalability considerations. One popular alternative is MIRA (Watanabe et al., 2007; Chiang et al., 2008, 2009), which is a perceptron-like online tuning algorithm with passive-aggressive updates. It uses an approximation to BLEU, where a sentence is scored in the context of a pseudo-document formed from the n-gram statistics for the last few updates. MIRA can scale to thousands of parameters and generally has no length bias (see Table 4).

Another recent, but already popular alternative to MERT is PRO (Hopkins and May, 2011), which models parameter tuning as pairwise ranking optimization. This is a batch tuning algorithm, which iterates between translation and optimization, just like MERT, but scales to thousands of parameters. It uses an add-one smoothed sentence-level version of BLEU, known as BLEU+1 (Lin and Och, 2004), and suffers from a length bias: the parameters it finds yield translations that are too short compared to the references (see Table 4). Exploring the reasons for this bias and proposing ways to solve it is the main focus of this paper.

There are many other tuning strategies, which fall outside of the scope of the current study, but to many of which some of our general finding and conclusions should apply. This includes improved versions of some of the above-mentioned algorithms, e.g., a batch version of MIRA (Cherry and Foster, 2012), or a linear regression version of PRO (Bazrafshan et al., 2012), but also many original algorithms that use a variety of machine learning methods and loss functions. We refer the interested reader to some excellent recent overviews: (McAllester and Keshet, 2011; Cherry and Foster, 2012; Gimpel and Smith, 2012).

To the best of our knowledge, no prior work has tried to study the reasons for the length bias of optimizers like PRO. However, researchers have previously expressed concerns about sentence-level BLEU+1, and some have proposed improvements, e.g., He and Deng (2012) used different smoothing for higher-order n-grams, unclipped brevity penalty, and scaled reference length. However, this was not done for the purpose of studying the length bias of PRO; moreover, as we will see below, the use of BLEU+1 is not the only reason for this bias.

3 The Length Bias with PRO

We explore the following hypotheses about the length bias with PRO:

- **PRO's optimization:** The bias could be due to the optimization mechanism of PRO.
- **BLEU+1:** PRO uses BLEU+1, where the add-one smoothing is applied to the precision component but does not touch the brevity penalty, which introduces an imbalance.
- **Sentence-level optimization:** PRO uses a (smoothed) sentence-level BLEU instead of corpus-level BLEU, which could make it hard to get the global corpus length right.

3.1 Possible Reason 1: The Optimization Mechanism of PRO

PRO is a batch optimizer that iterates between (i) *translation*: using the current parameter values, generate k-best translations, and (ii) *optimization*: using the translations from all previous iterations, find new parameter values. The optimization step has four substeps:

1. **Sampling:** For each sentence, sample uniformly at random $\Gamma = 5000$ pairs from the set of all candidate translations for that sentence from all previous iterations.
2. **Selection:** From these sampled pairs, select those for which the absolute difference in their BLEU+1 scores is higher than $\alpha = 0.05$.
3. **Acceptance:** For each sentence, accept the $\Xi = 50$ selected pairs with the highest absolute difference in their BLEU+1 scores.
4. **Learning:** Assemble the accepted pairs for all sentences into a single set and use it to train a ranker to prefer the higher-scoring sentence in each pair.

While sampling is unlikely to introduce a bias, selection and acceptance could do so.

Selection could filter too many pairs for some of the sentences, leaving for them less than Ξ pairs for the acceptance step; such sentences would be under-represented compared to those for which there are Ξ pairs accepted. If these under-represented sentences are generally longer then their references, this could yield a bias towards shorter translations.

Similarly, by focusing on the pairs with the highest differentials, the acceptance step would over-represent pairs with big differences in sentence lengths, e.g., a sentence that is extremely long/short compared to the reference would have a very low BLEU+1 score, and thus it will have a high differential when paired with any other sentence. If at some iteration, the k-best list is populated with overly long sentences, they would keep getting accepted in subsequent iterations of PRO, and thus the classifier will keep learning that being shorter is better.

Given the above discussion, we propose the following experiments with PRO:

- **PRO, no threshold.** At the selection step, keep everything, i.e., no selection step.
- **PRO, random accept.** At the acceptance step, accept the pairs at random, as opposed to accepting those with the highest differentials in BLEU+1, i.e., no acceptance step.
- **PRO, no threshold, random accept.** Combination of the previous two.

3.2 Possible Reason 2: BLEU+1

The other possible reasons are related to BLEU; thus, we should have a look at its definition:

$$\text{BLEU} = BP \cdot \left(\prod_{n=1}^{N} p_n \right)^{\frac{1}{N}} \tag{1}$$

BLEU has two components: (1) brevity penalty (BP), and (2) precision component (PC), the geometric mean of n-gram precisions p_n, $1 \leq n \leq N$. The BP is defined as follows:

$$BP = \begin{cases} 1 & \text{if } c > r \\ \exp\left(1 - \frac{r}{c}\right) & \text{if } c \leq r \end{cases} \tag{2}$$

where c is the length of the candidate, and r is the effective reference corpus length.

BLEU is defined at the corpus-level, and p_n, r and c are sums over all corpus sentences:

- $p_n = \frac{\sum_i m_{in}}{\sum_i h_{in}}$, where m_{in} is the number of n-gram matches between a translation and the references for sentence i, and h_{in} is the number of n-grams in the hypothesis;[2]
- $c = \sum_i c_i$, where c_i is the length of the candidate translation for sentence i;
- $r = \sum_i r_i$, where r_i is the reference length for sentence i; in case of multiple reference translations, this is the closest reference sentence length.

While BLEU is defined at the corpus level, PRO works at the sentence level and thus needs to optimize sentence-level BLEU for each sentence i. Such a version of BLEU can be obtained by redefining p_n, r and c to look at sentence i only: $p_n = \frac{m_{in}}{h_{in}}$, $c = c_i$, and $r = r_i$.

Using such a sentence-level version of BLEU is problematic though since it can easily become zero. This is because of the product of n-gram precisions in the geometric mean of the precision component of BLEU: if some p_n is zero, the whole product will be zero. In particular, it is easy to see that BLEU will be zero for any hypothesis without 4-gram matches. This is undesirable for optimization purposes since it does not allow to distinguish (a) a hypothesis translation that has no matches at all from (b) one that has unigram, bigram and trigram matches but no 4-gram matches; intuitively, the latter should be preferred over the former.

A popular strategy to solve the problem, which is also adopted by PRO, is to use an add-one smoothed version of BLEU, called BLEU+1, where p_n is redefined as follows: $p_n = \frac{m_{in}+1}{h_{in}+1}$.

There are two observations we can make about BLEU+1: (i) while it alters the precision component, it leaves the brevity penalty unchanged, and (ii) it is strictly positive.[3]

Let us start with the first observation. Adding one to m_{in} implies the need to add an extra n-gram to the reference translation – an n-gram that matches the candidate translation. This can be seen in the extreme case of a perfect match between the hypothesis and the reference: then, the additional match between the hypothesis and the reference, which is accounted for in m_{in}, would also require an additional word in the reference. However, the brevity penalty is not altered to account for this additional n-gram. Consider the special case of unigrams: if we assume that the reference contains an extra unigram, then it should be longer by one word. And vice versa, adding one extra word to the reference would increase the number of n-grams it contains by one for each n, $1 \leq n \leq N$, which is exactly what BLEU+1 assumes in its precision smoothing.

Let us try to analyze the impact of BLEU+1 on the hypothesis length. We have seen that BLEU+1 sees the precision component of a longer reference, but it pays the brevity penalty with respect to a shorter one. Because BLEU+1 boosts the precision component while leaving the BP intact, the relative weight of BP decreases compared to the original BLEU. This means that there could be potential gain from staying shorter if this can boost precision even further: the BP price to be paid for this would be relatively lower than it was in BLEU. Thus, high-scoring shorter hypotheses are more likely with BLEU+1 than with BLEU.

[2] More precisely, let g_n be an arbitrary n-gram, and let $\#_c(i, g_n)$ be the number of times g_n occurs in the candidate translation for sentence i. Let also $\#_r(i, g_n)$ be the maximum number of occurrences of g_n in any reference translation for sentence i. Then, we can define $m_{in} = \sum_{g_n} \min\left(\#_c(i, g_n), \#_r(i, g_n)\right)$ and $h_{in} = \sum_{g_n} \#_c(i, g_n)$.

[3] While BLEU+1 in PRO is strictly positive, there are definitions in the literature that allow it to become zero, e.g., Lin and Och (2004) do not smooth the unigram counts, which makes BLEU+1 zero in case of no matches at all. Experimenting with their version of BLEU+1 is left for future work; it would still need a fix for BP though.

Let us now consider the second observation: that BLEU+1 is strictly positive. While there is nothing wrong with this per se, it suggests a different way to restore the balance between BP and PC: by "grounding" BLEU+1 so that it is zero when there are no matches at all. This can be achieved by subtracting from the precision component of BLEU+1 the score for that component when there are no matches. Thus, the "grounded" precision component of BLEU+1 changes from $PC = \left(\prod_{n=1}^{N} \frac{m_{in}+1}{h_{in}+1}\right)^{\frac{1}{N}}$ to $PC = \left(\prod_{n=1}^{N} \frac{m_{in}+1}{h_{in}+1}\right)^{\frac{1}{N}} - \left(\prod_{n=1}^{N} \frac{1}{h_{in}+1}\right)^{\frac{1}{N}}$.

Given the above discussion, we propose the following fixes for the sentence-level BLEU+1:

- **BLEU, PC-unsmoothed**. Just use BLEU. The idea is that if add-one smoothing in BLEU+1 is causing length problems, they should go away if unsmoothed BLEU is used instead. While unsmoothed BLEU will make many pairs of hypotheses indistinguishable, both having a score of zero, many other pairs with at least one non-zero-score hypothesis will still remain, and they should be enough to train the classifier of PRO.
- **BLEU+1, grounded.** "Ground" the PC of BLEU+1 by subtracting from it its value when there are no matches. Given the asymmetry of add-one smoothing in BLEU+1, which boosts the PC for shorter sentences more than for longer ones for the same number of matches, "grounded" BLEU is slightly more biased towards longer sentences.
- **BLEU+1, BP-smoothed.** Use add-one smoothing not only for the precision component but also for the length of the reference translation, i.e., use $r = r_i + 1$, in addition to $p_n = \frac{m_{in}+1}{h_{in}+1}$. The idea here is to smooth the PC and the BP consistently, thus maintaining the balance between them: if we assume an extra n-gram in the reference, then we should also assume that the reference contains an extra word.
- **BLEU+1, BP-smoothed, grounded.** Combination of the previous two: smooth the reference length in BP and also ground the precision component of BLEU+1 .

3.3 Possible Reason 3: Sentence-Level Optimization

Another possible reason for the length issue with PRO could be a bias due to a sentence-level version of BLEU being optimized as opposed to using a corpus-level BLEU. If, for whatever reason, being a bit shorter than the reference is preferable to being a bit longer, e.g., because sentence-level BLEU+1 penalizes longer sentences more than shorter ones, then the candidate translation for each sentence will try to stay on the short side as opposed to getting longer than the reference translation. These length differences might be minimal at the sentence level, but they would accumulate and would eventually yield a larger difference at the corpus level. The crucial observation is that if each and every sentence looks at its length in isolation, it would be reluctant to getting longer than its reference (since this would harm its sentence-level BLEU), even if this could improve corpus-level BLEU.

One possible way to address the problem is to introduce in the sentence-level brevity penalty information about the ratio of the corpus-level length and the effective reference corpus length from the previous iteration of PRO. For example, if the corpus length was shorter than the effective reference corpus length, then we could scale the sentence-level reference lengths r_i on the current iteration accordingly, so that proportionally higher brevity penalty be paid for being too short; and vice versa, if on the previous iteration the corpus length was too long, we could scale the reference length so that no penalty be paid for staying proportionally shorter; this idea was previously explored by He and Deng (2012).

Another possibility is to unclip the brevity penalty, as proposed by He and Deng (2012), by allowing it to get bigger than 1, thus effectively becoming a "bonus" for longer sentences. If there is a bias in sentence-level BLEU+1 towards shorter translations, i.e., longer hypotheses are penalized by the precision component more heavily than shorter hypotheses are penalized by BP, then turning the brevity penalty into a bonus for longer hypothesis should encourage them; note, however, that they will still be discouraged by the decrease in the score for the precision component, which should act as a counter-balance.

Finally, if optimizing sentence-level BLEU causes problems, we could just get rid of it and use some kind of corpus-level BLEU instead. This is what MIRA does (Chiang et al., 2008): even though it is an online algorithm and makes updates on a per-sentence basis, it does not use a sentence-level BLEU, but instead it calculates the BLEU score for the current sentence in the context of a pseudo-corpus that tracks the n-gram statistics of the model-best derivations from the last few updates. We could adopt a similar strategy here as well.

Given the above, we propose the following fixes/substitutes for the sentence-level BLEU+1:

- **BLEU+1, unclipped.** Use an unclipped brevity penalty: $BP = \exp\left(1 - \frac{r}{c}\right)$.
- **BLEU+1, scaled.** Scale the reference length with the inverse length ratio (ILR) from the previous iteration of PRO, i.e., define $r = r_i \cdot ILR$, where $ILR = c'/r' = \sum_i c'_i / \sum_i r'_i$, where c' and r' are the corpus-level candidate and reference lengths from the previous iteration, respectively; we define $LR = 1$ for the first iteration of PRO.
- **BLEU+1, unclipped, scaled.** Combination of the previous two: BLEU+1 with unclipped brevity penalty and also with scaled reference length. This combination is advocated by He and Deng (2012).[4]
- **Corpus-level BLEU, MIRA-style.** This is BLEU calculated using a background pseudo-corpus, similar to the way this is done in MIRA. In our case, this pseudo-corpus is initialized with an exponentially decaying sum over the sufficient statistics for the one-best hypotheses from all previous iterations of PRO: taken with a weight of 0.9 for the last iteration,[5] with a weight of 0.9^2 for the iteration before it, etc. Then, the BLEU score for a candidate sentence is calculated by adding its sufficient statistics to the sufficient statistics for the background pseudo-corpus. Each time PRO samples, selects, and accepts a pair of hypotheses to use for training its classifier, we immediately update the pseudo-corpus by adding to it the sufficient statistics for the higher-scoring sentence, i.e., the positive example, in the pair. Thus, while the pseudo-corpus is initialized with the statistics from previous iterations, it soon gets dominated by statistics for the positive examples that are being accepted as training sentence pairs at the current iteration, since there are more of them than there are one-best hypotheses from previous iterations.[6] As in MIRA, our pseudo-document is time-dependent and thus can adjust dynamically to changing sentence lengths by encouraging or discouraging longer translations depending on what kinds of sentences have been accepted already.

[4] He and Deng (2012) further add to the combination a sophisticated smoothing for the precision component; however, in our experiments, the combination of all three changes to BLEU of theirs did not work well with PRO, yielding extremely long translations and an absolute loss of seven BLEU points on the NIST datasets.

[5] The value of 0.9 was found to work well in general, but many other values yielded a similar result since the BLEU score gets dominated by examples from the current iteration very quickly, making this value irrelevant.

[6] We only allow up to 25 iterations, which means there could be up to 24 accumulated one-best hypotheses per sentence, while we accept up to $\Xi = 50$ pairs per sentence.

4 Experiments and Evaluation

We compare variations of three parameter optimization algorithms: MERT, PRO, and MIRA. In all experiments, we use the phrase-based SMT model (Koehn et al., 2003) as implemented in the Moses toolkit (Koehn et al., 2007), and we report evaluation results over two datasets: NIST, which has four reference translations, and IWSLT, with a single reference translation.

In order to be able to directly compare the candidate/reference length ratios on the development and on the testing datasets, we need to make sure that we use the same tokenization when calculating BLEU on tuning and on testing. Such differences can arise because many standard scoring tools, e.g., those of NIST, work on detokenized text, which they retokenize again internally; this retokenization typically differs from the one used by the SMT system. As a result, the optimizer and the final scorer would most certainly see different tokenizations, and more crucially, this could affect the length ratios and thus the brevity penalty. Thus, we report BLEU scores calculated using the *multi-bleu* scoring tool, which uses the supplied tokenization and does no retokenization; we supplied it with references that were tokenized and truecased using the same models that were used for training and tuning.[7]

Finally, in order to avoid stability issues, we report results averaged over three runs.

4.1 Experimental Setup

Preprocessing: We tokenized the English side of all bi-texts and the monolingual data for language modeling using the standard tokenizer of Moses. We further truecased this data by changing the casing of each sentence-initial word to its most frequent casing in the training corpus; for lines containing ALL CAPS, we did this for each word. We segmented the words on the Arabic side using the ATB segmentation scheme: we used MADA (Roth et al., 2008) for NIST, and the Stanford word segmenter (Green and DeNero, 2012) for IWSLT.

Training. We built separate directed word alignments using IBM model 4 (Brown et al., 1993), we symmetrized them with the *grow-diag-final-and* heuristic of Moses, and we extracted phrase pairs of length up to seven. We scored these pairs using maximum likelihood with Kneser-Ney smoothing, to build a phrase table with five standard scores: forward and reverse phrase translation probabilities, forward and reverse lexical translation probabilities, and phrase penalty. We also built a lexicalized reordering model (Koehn et al., 2005): *msd-bidirectional-fe*. For language modeling, we trained a separate 5-gram Kneser-Ney smoothed model on each corpus (target side of a training bi-text or monolingual dataset); we then interpolated these models minimizing the perplexity on the target side of the tuning dataset. Finally, we built a log-linear model including the language model probability, the word penalty, and the parameters from the phrase and the reordering tables.

Tuning. We tuned the weights in the log-linear model by optimizing BLEU (Papineni et al., 2002) on the tuning dataset, using MERT, PRO, or MIRA. We allowed optimizers to run for up to 25 iterations, and we allowed them to extract 1000-best lists for each iteration.

Decoding. On tuning and testing, we dropped unknown words (this has yielded slightly shorter translations) and we used monotone-at-punctuation decoding (this had no impact on the translation length). On testing, we further used cube pruning and minimum Bayes Risk decoding (the latter yielded slightly longer translations).

[7] Still, for comparison purposes, we also report BLEU calculated with respect of the original references using NIST v13a, after detokenization and recasing of the system's output (shown in small script in the tables).

4.2 Datasets

We experimented with the Arabic-English datasets from two machine translation evaluation campaigns: (1) the NIST 2012 Open Machine Translation Evaluation[8], and (2) the IWSLT 2011 Evaluation Campaign on Automatic Talk Translation (Federico et al., 2012).

1. NIST: We trained the phrase and the reordering tables on all training datasets from NIST 2012 (except for UN), we tuned on MT06 and tested on MT09. For language modeling, we built a separate LM from the English side of each training dataset, and from each year of the English GigaWord; we then interpolated them into a single LM.
2. IWSLT: We trained the phrase and the reordering tables on the TED training dataset, we tuned on dev2010, and we tested on tst2010. Since there was a small mismatch in the source/reference length ratios between dev2010 and tst2010, we also experimented with reversed tuning/testing, tuning on tst2010 and testing on dev2010; this is to see the impact of the test set length ratio being a bit longer and also being a bit shorter than the tuning dataset ratio. We used two LMs: one trained on the English side of TED, and another one that is an interpolation of Europarl and WMT11 News.

4.3 Evaluation Results

We experimented with MERT, MIRA, PRO, and the various fixes thereof that were introduced in Section 3. The results of these experiments are shown in Tables 1, 2, 3, and 4. In these tables, the first column describes the method, followed by (1) the BLEU scores and (2) the length ratio of the candidate corpus-level translation to the effective reference corpus length, which are calculated (i) for the tuning dataset using multi-bleu,[9] (ii) for the test dataset using multi-bleu, and (iii) for the test dataset using the NIST scoring tool v13a.

The most important column in these tables is column three, which reports the candidate/reference length ratio for the tuning dataset as calculated using multi-bleu. We will be observing the impact of the various fixes we propose on this ratio: the closer it gets to 1.0, the better the fix should be; this will be verified by the value in the following column four, which shows the BLEU score on the test dataset calculated using multi-bleu.

4.3.1 Testing Possible Reason 1: The Optimization Mechanism of PRO

Table 1 explores possible reason 1 from Section 3, i.e., that there may be a bias in the optimization mechanism of PRO. It compares the original PRO to (a) PRO with no threshold for the selection step (default: filter pairs whose difference in BLEU+1 is less than a threshold: $\alpha = 0.05$), (b) PRO with random acceptance for the acceptance step (default: accept the pairs with the highest absolute difference in BLEU+1), and (c) combination of (a) and (b).

We can see in Table 1 that the length ratio in column three stays relatively stable for the different versions of PRO for all testing datasets. Note, however, the notable exception of IWSLT for (c), which exhibits a large increase in the length ratio: from 0.958 to 0.967.

[8] `http://www.nist.gov/itl/iad/mig/openmt12.cfm`

[9] Note that the results for the tuning dataset are obtained not using the BLEU score for the last iteration of the respective optimizer, but by translating the development set one more time – using the weights that the tuning algorithm has found. This is because, for some of the optimizers, the final tuning weight calculation might use weights from previous iterations, e.g., PRO accumulates and interpolates weights, while our version of MIRA, Batch-MIRA (Cherry and Foster, 2012), returns the weights from the iteration that yielded the highest BLEU score (using this option was suggested by the authors of Batch-MIRA; we found that it also worked best for our datasets).

Method	Tune: multi-bleu BLEU	Tune: multi-bleu len ratio	Test: multi-bleu BLEU	Test: multi-bleu len ratio	Test: NIST v13a BLEU	Test: NIST v13a len ratio
NIST dataset (tune: MT06, eval: MT09)						
PRO	45.65	**0.981**	48.45	0.976	47.80	0.978
PRO, no threshold	45.62	**0.981**	$48.39_{(-0.06)}$	0.975	47.74	0.977
PRO, accept random	45.55	**0.979**	$48.11_{(-0.33)}$	0.973	47.48	0.976
PRO, no threshold, accept random	45.58	**0.981**	$48.23_{(-0.22)}$	0.970	47.59	0.973
IWSLT dataset (tune: dev2010, eval: tst2010)						
PRO	26.96	**0.958**	26.34	0.965	25.30	0.973
PRO, no threshold	26.94	**0.959**	$26.24_{(-0.10)}$	0.966	25.18	0.975
PRO, accept random	26.97	**0.958**	$26.29_{(-0.05)}$	0.965	25.26	0.973
PRO, no threshold, accept random	26.87	**0.967**	$26.29_{(-0.05)}$	0.975	25.24	0.983
IWSLT dataset – reversed (tune: tst2010, eval: dev2010)						
PRO	26.08	**0.952**	26.45	0.945	25.43	0.946
PRO, no threshold	26.05	**0.952**	$26.51_{(+0.06)}$	0.945	25.49	0.947
PRO, accept random	26.11	**0.953**	$26.48_{(+0.03)}$	0.946	25.46	0.949
PRO, no threshold, accept random	26.06	**0.954**	$26.36_{(-0.09)}$	0.946	25.35	0.948

Table 1: Testing possible reason 1: bias in the optimization mechanism of PRO.

These results are inconclusive but they show that, at least for one dataset, the length bias in PRO was influenced by its selection and acceptance steps.[10] Overall, multi-bleu in column 4 stays stable, with a slight degradation in some cases, which is not surprising and can be attributed to PRO training its classifier on less reliable examples in these cases – ones with smaller BLEU+1 differentials, which risks focusing on tiny, unimportant distinctions.

4.3.2 Testing possible Reason 2: Bias in the Smoothing of BLEU+1

Table 2 explores possible reason 2 from Section 3, i.e., that the length issue may be caused by the smoothing in BLEU+1; see also Section 3.2 for more detail. It compares the results for the original PRO to a version (a) without smoothing of the precision component, (b) with grounding of the precision component, (c) with smoothing of the brevity penalty, and (d) with both BP-smoothing and PC-grounding.

Several interesting observations can be made about this table. First, we can see that using unsmoothed BLEU instead of BLEU+1 yields consistent improvements of the length ratio for all datasets; it also improves multi-bleu and NIST v13a scores. This suggests that the smoothing in BLEU+1 could indeed be one of the causes for the length problem. A similar trend can be seen for grounded BLEU+1, which performs slightly better, both in terms of length ratio and in terms of multi-bleu and NIST v13a scores. This suggests that part of the problem could be the balance between the precision component and the brevity penalty in BLEU+1: grounding reduces the absolute value of the precision component, which was inflated by BLEU+1, and thus helps restore a balance between the two components that is closer to that in BLEU. Moreover, grounded BLEU+1 has the advantage of assigning a non-zero value to any sentence with at least one unigram match, while unsmoothed BLEU assigns a zero score to many sentences, which could leave PRO with less training examples.

[10] However, there could be other reasons, e.g., the kind of classifier PRO uses, the fact that it samples from all previous steps, etc.; exploring these possibilities is left for future work.

Method	Tune: multi-bleu BLEU	len ratio	Test: multi-bleu BLEU	len ratio	Test: NIST v13a BLEU	len ratio
NIST dataset (tune: MT06, eval: MT09)						
PRO	45.65	**0.981**	48.45	0.976	47.80	0.978
PRO, PC-unsmoothed	45.83	**0.987**	$48.58_{(+0.13)}$	0.985	47.87	0.986
PRO, PC-grounded	45.82	**0.986**	$48.62_{(+0.17)}$	0.985	47.96	0.986
PRO, BP-smoothed	45.76	**0.988**	$48.65_{(+0.20)}$	0.985	48.03	0.988
PRO, BP-smoothed, PC-grounded	45.70	**0.992**	$48.79_{(+0.34)}$	0.991	48.16	0.993
IWSLT dataset (tune: dev2010, eval: tst2010)						
PRO	26.96	**0.958**	26.34	0.965	25.30	0.973
PRO, PC-unsmoothed	27.05	**0.970**	$26.46_{(+0.12)}$	0.979	25.38	0.987
PRO, PC-grounded	27.13	**0.975**	$26.48_{(+0.14)}$	0.984	25.39	0.992
PRO, BP-smoothed	27.10	**0.987**	$26.38_{(+0.04)}$	0.995	25.21	1.002
PRO, BP-smoothed, PC-grounded	27.16	**0.996**	$26.33_{(-0.01)}$	1.004	25.04	1.011
IWSLT dataset – reversed (tune: tst2010, eval: dev2010)						
PRO	26.08	**0.952**	26.45	0.945	25.43	0.946
PRO, PC-unsmoothed	26.09	**0.953**	$26.49_{(+0.04)}$	0.946	25.47	0.947
PRO, PC-grounded	26.24	**0.967**	$26.75_{(+0.30)}$	0.959	25.71	0.960
PRO, BP-smoothed	26.38	**0.991**	$27.10_{(+0.65)}$	0.982	26.07	0.983
PRO, BP-smoothed, PC-grounded	26.41	**1.003**	$27.07_{(+0.62)}$	0.995	26.04	0.994

Table 2: Testing possible reason 2: bias in the smoothing of BLEU+1.

Using add-one smoothing for the brevity penalty of BLEU+1 yields even better results on NIST and IWSLT-reversed: for IWSLT-reversed, the length ratio jumps from 0.952 to 0.991, and multi-bleu on the test dataset gains +0.65 absolute. The improvements in the length ratio are comparable for IWSLT-forward (from 0.958 to 0.987), and are more modest for NIST (from 0.981 to 0.988), where there is less room for improvement because of the higher baseline and because of the multiple reference translations. Overall, BP-smoothing works better than grounding, which suggests that smoothing with add-one both the precision component and the brevity penalty is a good way to restore the balance between them.

Finally, the combination of BP-smoothing and grounding yields even further improvements in the length ratios: 0.992 for NIST, 0.996 for IWSLT, and 1.003 for IWSLT-reversed. This adds +0.14 additional multi-bleu points to NIST, but slightly hurts IWSLT in both directions. This suggests that while grounding helps improve the balance between the precision component and ultimately the brevity penalty even further, it does this at the cost of deteriorating the estimates for the precision component, and thus the result in terms of multi-bleu is mixed.

Overall, as the length ratio gets closer to 1 on tuning, it does so on testing as well; this typically also yields an improvement in both multi-bleu and NIST v13a – we have achieved absolute multi-bleu improvements of up to +0.34 for NIST and +0.65 for IWSLT-reversed.

4.3.3 Testing Possible Reason 3: Sentence-Level Optimization

Table 3 explores possible reason 3 from Section 3, i.e., that the length issue may be due to sentence-level optimization. It compares the results for the original PRO to a version that (a) scales the reference length with the inverse document-level length ratio from the previous iteration of PRO, (b) unclips the brevity penalty, and (c) a combination thereof.

Method	Tune: multi-bleu BLEU	len ratio	Test: multi-bleu BLEU	len ratio	Test: NIST v13a BLEU	len ratio
NIST dataset (tune: MT06, eval: MT09)						
PRO	45.65	**0.981**	48.45	0.976	47.80	0.978
PRO, scaled	45.72	**0.984**	$48.50_{(+0.05)}$	0.979	47.87	0.981
PRO, unclipped	45.78	**0.992**	$48.81_{(+0.36)}$	0.990	48.18	0.992
PRO, unclipped, scaled	45.83	**0.993**	$48.82_{(+0.37)}$	0.991	48.19	0.993
IWSLT dataset (tune: dev2010, eval: tst2010)						
PRO	26.96	**0.958**	26.34	0.965	25.30	0.973
PRO, scaled	27.07	**0.975**	$26.39_{(+0.05)}$	0.984	25.31	0.991
PRO, unclipped	26.54	**1.024**	$25.63_{(-0.71)}$	1.035	24.34	1.040
PRO, unclipped, scaled	26.38	**1.030**	$25.43_{(-0.92)}$	1.042	24.14	1.046
IWSLT dataset – reversed (tune: tst2010, eval: dev2010)						
PRO	26.08	**0.952**	26.45	0.945	25.43	0.946
PRO, scaled	26.29	**0.972**	$26.75_{(+0.30)}$	0.962	25.73	0.963
PRO, unclipped	25.65	**1.033**	$26.40_{(-0.05)}$	1.021	25.34	1.021
PRO, unclipped, scaled	25.53	**1.037**	$26.09_{(-0.36)}$	1.028	25.02	1.028

Table 3: Testing possible reason 3: sentence-level optimization.

We can see that scaling does a great job at improving the length ratio, especially for IWSLT (in both directions), where it improves from 0.958/0.952 to 0.975/0.972; the effect is more limited for NIST – from 0.981 to 0.984, probably because of the higher baseline and due to the multiple reference translations. Scaling also yields small but consistent improvements in the multi-bleu/NIST-v13a scores on the tuning and the test datasets: up to +0.30 multi-bleu for IWSLT-reversed (but only +0.05 on the other two datasets).

We can further see that unclipping the brevity penalty in BLEU+1 yields a much larger increase in the length ratio, but this has mixed effect on the multi-bleu score: for NIST, the length ratio improves from 0.981 to 0.992, which yields a gain of +0.36 multi-bleu points absolute, but for IWSLT, the ratio jumps over 1, which yields a drop in the multi-bleu score.

The combination of unclipping and scaling yields an even higher length ratio, which helps NIST a bit, but hurts IWSLT even further as its length increases even more over 1.

Next, we experimented with sentence-level vs. corpus-level BLEU for PRO. The results are shown in Table 4. We can see that switching to corpus-level BLEU (calculated with respect to a pseudo-document, similarly to MIRA; see Section 3) yields a perfect length ratio of 1 on all tuning datasets, which is on par with the length ratios of MIRA and MERT, which also optimize corpus-level BLEU. This also improves the tuning multi-bleu for all datasets, and the testing multi-bleu by +0.14 for NIST and by +0.29 for IWSLT-reverse; however, there is a drop of -0.14 for IWSLT-forward because of a test length ratio that goes over 1 – due to a difference in the source/target length ratios between the tuning and the test sets for IWSLT.

We further tested the impact of doing the reverse: plugging a sentence-level BLEU+1 score as an objective in a corpus-level optimizer that generally has no length bias – MIRA. The results are shown in Table 4. We can see that MIRA with sentence-level BLEU+1 yields short reference translations just like PRO. This suggests that optimizing for sentence-level BLEU+1 yields short translations, regardless of the optimizer that is being used.

Method	Tune: multi-bleu BLEU	len ratio	Test: multi-bleu BLEU	len ratio	Test: NIST v13a BLEU	len ratio	*Avg. sent-BLEU tune*	*test*
NIST dataset (tune: MT06, eval: MT09)								
PRO	45.65	**0.981**	48.45	0.976	47.80	0.978	*46.20*	*47.75*
MIRA, sent-BLEU+1	45.51	**0.986**	48.18	0.983	47.52	0.985	*46.07*	*47.40*
PRO, corpus-BLEU	45.83	**1.000**	48.59	0.999	47.89	1.001	*45.83*	*47.59*
MIRA	45.26	**1.006**	48.26	1.006	47.48	1.009	*45.52*	*47.36*
MERT	45.58	**0.997**	48.48	0.997	47.72	0.998	*45.56*	*47.44*
IWSLT dataset (tune: dev2010, eval: tst2010)								
PRO	26.96	**0.958**	26.34	0.965	25.30	0.973	*32.03*	*31.43*
MIRA, sent-BLEU+1	26.93	**0.983**	25.81	0.994	24.62	0.999	*31.64*	*30.69*
PRO, corpus-BLEU	27.06	**1.000**	26.20	1.011	24.90	1.017	*31.50*	*30.89*
MIRA	27.28	**1.002**	26.09	1.012	24.78	1.017	*31.62*	*30.85*
MERT	27.24	**1.005**	25.84	1.017	24.53	1.020	*31.63*	*30.68*
IWSLT dataset – reversed (tune: tst2010, eval: dev2010)								
PRO	26.08	**0.952**	26.45	0.945	25.43	0.946	*31.35*	*31.70*
MIRA, sent-BLEU+1	26.25	**0.974**	26.75	0.965	25.69	0.966	*31.37*	*31.97*
PRO, corpus-BLEU	26.15	**1.000**	26.74	0.989	25.72	0.990	*30.73*	*31.12*
MIRA	26.54	**0.998**	27.09	0.988	26.08	0.989	*31.16*	*31.71*
MERT	26.33	**1.003**	26.88	0.992	25.86	0.993	*30.94*	*31.43*

Table 4: Sentence-level vs. corpus-level BLEU: experiments with PRO, MIRA, and MERT.

Note, however, that the length ratio with sentence-level MIRA does not drop all the way down to that of PRO, which also uses sentence-level optimization; this suggests once again that part of the bias may be coming from the optimization mechanism of PRO (which we have explored above as possible reason 1). This difference is larger for IWSLT than for NIST, which means that this bias is less pronounced with multiple reference translations.

The last two columns in Table 4 show the average sentence-level BLEU score, calculated for the tuning and the testing datasets. We can see that, on the tuning dataset, sentence-level optimizers score consistently higher than corpus-based ones on sent-BLEU, but they are not very competitive on corpus-BLEU; this suggests a mismatch of objectives. Note, also the relatively big difference between corpus-BLEU (i.e., multi-bleu – columns 2 and 4) and sent-BLEU (which uses no smoothing – columns 8 and 9) for IWSLT (about 4-5 BLEU points absolute), and the much smaller difference for NIST (0-1 BLEU points). One possible explanation is that with multiple references, average sent-BLEU approximates corpus-BLEU better; this is yet another possible reason for the length issue with PRO being less pronounced with NIST compared to IWSLT.

5 Discussion

Our experiments suggest that the length issue of PRO is due primarily to optimizing for sentence-level BLEU+1, which (i) cannot "see" the global length ratio, and (ii) uses a biased smoothing in BLEU+1, which boosts the precision part of the score, but leaves the brevity penalty intact, thus destroying the balance between the two. This is confirmed by an experiment where we plugged sentence-level BLEU+1 in MIRA. We have also found that part of the bias may be coming from the optimization mechanism of PRO.

We have proposed a number of ways to fix the length ratio of PRO by fixing BLEU+1, some of which have worked fairly well, e.g., smoothing the brevity penalty and scaling the effective reference length. Other fixes such as grounding the precision component and unclipping the brevity penalty were less helpful. Moreover, some combinations thereof have yielded too long length ratios, which had a negative impact on multi-bleu for IWSLT, but not for NIST.

Given our experimental results, we cannot make a universal recommendation about which fixes would work best for all datasets, but, in practical terms, we would recommend trying different fixes and choosing the one for which the length ratio is closest to 1 on the tuning dataset. Following this advice for Tables 2 and 3, we would select BLEU+1 with unclipped BP and scaled reference length for NIST (tuning length ratio of 0.993), and BLEU+1 with BP-smoothing and PC-grounding for IWSLT (tuning length ratio of 1.003), yielding +0.37 multi-bleu for NIST and +0.62 for IWSLT-reversed on the test dataset.

Coming back to our title: optimizing for sentence-level BLEU+1 yields short translations. This is true for the tuning dataset that the optimizer sees, but we can make no claims about the lengths of the test-time translations. This is because they can differ from the tuning set a lot in terms of verbosity, and the optimizer has no control over this.[11] However, if the tuning and the testing datasets have similar source/reference ratios (which is the best guess in the absence of other information), we expect that the claim would also hold for the test set.[12]

6 Conclusion and Future Work

We hope that the present study will be useful for those who want to use optimizers such as PRO and MIRA, which can handle many features, but feel frustrated that certain idiosyncrasies cause MERT to still yield the best test BLEU scores in certain cases. Here we have pointed out that one cause for this are poor length ratios in the resulting translations, which we have attributed to the use of sentence-level BLEU+1 as an objective function. We have thus suggested a number of simple modifications, which do improve the length ratio in practice, ultimately yielding better BLEU scores, while also preserving the sentence-level nature of BLEU+1, which makes optimizers simpler conceptually and implementation-wise.

In future work, we plan a more systematic study of the relationship between optimizers and objective functions with respect to the target/reference length ratio, which would be extended with other optimizers such as TER (Snover et al., 2006) and METEOR (Lavie and Denkowski, 2009). Overall, we see two promising general directions in which the present study can be extended. First, explore the relationship between sentence-level and corpus-level optimization and the possibility to combine them. Second, study the characteristics of translations generated using weights from different optimizers: while here we have only touched length, we believe there are many other important aspects that are worth exploring.

Acknowledgments

We thank the anonymous reviewers for their comments, which helped us improve the paper.

[11] For example, the average source/reference length ratio for our NIST tuning dataset MT06 is 1.248, which is very close to that for MT09, which is 1.252, and thus translation/reference length ratios are very close as well; however, this is not so for MT05, where the source/reference ratio is only 1.183; thus, tuning on MT06 and testing on MT05 yields translations that are too long even for standard PRO. There is also some imbalance in the source/reference length ratios of dev2010 vs. tst2010 for IWSLT, and thus we experimented with reversing them.

[12] The internal scoring tool segmentation and the recasing have an influence as well; though, we have seen in columns 6-7 of all tables that (1) the length ratios are quite close, and (2) the relative improvements in terms of BLEU are quite correlated for multi-bleu and for the NIST scoring tool v13a.

References

Bazrafshan, M., Chung, T., and Gildea, D. (2012). Tuning as linear regression. In *Proceedings of the 2012 Annual Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, NAACL-HLT '12, pages 543–547, Montréal, Canada.

Brown, P. F., Pietra, V. J. D., Pietra, S. A. D., and Mercer, R. L. (1993). The mathematics of statistical machine translation: parameter estimation. *Computational Linguistics*, 19(2):263–311.

Cherry, C. and Foster, G. (2012). Batch tuning strategies for statistical machine translation. In *Proceedings of the 2012 Annual Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, NAACL-HLT '12, pages 427–436, Montréal, Canada.

Chiang, D., Knight, K., and Wang, W. (2009). 11,001 new features for statistical machine translation. In *Proceedings of the 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, NAACL-HLT '09, pages 218–226, Boulder, CO, USA.

Chiang, D., Marton, Y., and Resnik, P. (2008). Online large-margin training of syntactic and structural translation features. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, EMNLP '08, pages 224–233, Honolulu, Hawaii, USA.

Federico, M., Stüker, S., Bentivogli, L., Paul, M., Cettolo, M., Herrmann, T., Niehues, J., and Moretti, G. (2012). The IWSLT 2011 evaluation campaign on automatic talk translation. In Calzolari, N., Choukri, K., Declerck, T., Doğan, M. U., Maegaard, B., Mariani, J., Odijk, J., and Piperidis, S., editors, *Proceedings of the Eight International Conference on Language Resources and Evaluation*, LREC '12, pages 3543–3550, Istanbul, Turkey.

Foster, G. and Kuhn, R. (2009). Stabilizing minimum error rate training. In *Proceedings of the Fourth Workshop on Statistical Machine Translation*, StatMT '09, pages 242–249, Athens, Greece.

Gimpel, K. and Smith, N. A. (2012). Structured ramp loss minimization for machine translation. In *Proceedings of the 2012 Annual Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, NAACL-HLT '12, pages 221–231, Montréal, Canada.

Green, S. and DeNero, J. (2012). A class-based agreement model for generating accurately inflected translations. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, ACL '12, pages 146–155, Jeju Island, Korea.

He, X. and Deng, L. (2012). Maximum expected BLEU training of phrase and lexicon translation models. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, ACL '12, pages 292–301, Jeju Island, Korea.

Hopkins, M. and May, J. (2011). Tuning as ranking. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, EMNLP '11, pages 1352–1362, Edinburgh, Scotland, United Kingdom.

Koehn, P., Axelrod, A., Mayne, A. B., Callison-Burch, C., Osborne, M., and Talbot, D. (2005). Edinburgh system description for the 2005 IWSLT speech translation evaluation. In *Proceedings of the International Workshop on Spoken Language Translation*, IWSLT '05, Pittsburgh, PA, USA.

Koehn, P., Hoang, H., Birch, A., Callison-Burch, C., Federico, M., Bertoldi, N., Cowan, B., Shen, W., Moran, C., Zens, R., Dyer, C., Bojar, O., Constantin, A., and Herbst, E. (2007). Moses: Open source toolkit for statistical machine translation. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics (Demonstration session)*., ACL '07, pages 177–180, Prague, Czech Republic.

Koehn, P., Och, F. J., and Marcu, D. (2003). Statistical phrase-based translation. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology (Volume 1)*, HLT-NAACL '03, pages 48–54, Edmonton, Canada.

Lavie, A. and Denkowski, M. J. (2009). The Meteor metric for automatic evaluation of machine translation. *Machine Translation*, 23:105–115.

Lin, C.-Y. and Och, F. J. (2004). ORANGE: a method for evaluating automatic evaluation metrics for machine translation. In *Proceedings of the 20th International Conference on Computational Linguistics*, COLING '04, pages 501–507, Geneva, Switzerland.

McAllester, D. and Keshet, J. (2011). Generalization bounds and consistency for latent structural probit and ramp loss. In Shawe-Taylor, J., Zemel, R., Bartlett, P., Pereira, F., and Weinberger, K., editors, *Advances in Neural Information Processing Systems 24*, NIPS '11, pages 2205–2212, Granada, Spain.

Och, F. J. (2003). Minimum error rate training in statistical machine translation. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, ACL '03, pages 160–167, Sapporo, Japan.

Papineni, K., Roukos, S., Ward, T., and Zhu, W.-J. (2002). BLEU: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, ACL '02, pages 311–318, Philadelphia, PA, USA.

Roth, R., Rambow, O., Habash, N., Diab, M., and Rudin, C. (2008). Arabic morphological tagging, diacritization, and lemmatization using lexeme models and feature ranking. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, ACL '08, pages 117–120, Columbus, OH, USA.

Snover, M., Dorr, B., Schwartz, R., Micciulla, L., and Makhoul, J. (2006). A study of translation edit rate with targeted human annotation. In *Proceedings of the 7th Biennial Conference of the Association for Machine Translation in the Americas*, AMTA '06, pages 223–231, Cambridge, MA, USA.

Watanabe, T., Suzuki, J., Tsukada, H., and Isozaki, H. (2007). Online large-margin training for statistical machine translation. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, EMNLP-CoNLL '07, pages 764–773, Prague, Czech Republic.

Grammarless Parsing for Joint Inference

Jason Naradowsky[1,2] *Tim Vieira*[3] *David A. Smith*[4]
(1) University of Massachusetts Amherst, Amherst, Massachusetts
(2) Macquarie University, Sydney, Australia
(3) Johns Hopkins University, Baltimore, Maryland
(4) Northeastern University, Boston, Massachusetts
narad@cs.umass.edu, tim@cs.jhu.edu, dasmith@ccs.neu.edu

Abstract

Many NLP tasks interact with syntax. The presence of a named entity span, for example, is often a clear indicator of a noun phrase in the parse tree, while a span in the syntax can help indicate the lack of a named entity in the spans that cross it. For these types of problems joint inference offers a better solution than a pipelined approach, and yet large joint models are rarely pursued. In this paper we argue this is due in part to the absence of a general framework for joint inference which can efficiently represent syntactic structure.

We propose an alternative and novel method in which constituency parse constraints are imposed on the model via combinatorial factors in a Markov random field, guaranteeing that a variable configuration forms a valid tree. We apply this approach to jointly predicting parse and named entity structure, for which we introduce a zero-order semi-CRF named entity recognizer which also relies on a combinatorial factor. At the junction between these two models, soft constraints coordinate between syntactic constituents and named entity spans, providing an additional layer of flexibility on how these models interact. With this architecture we achieve the best-reported results on both CRF-based parsing and named entity recognition on sections of the OntoNotes corpus, and outperform state-of-the-art parsers on an NP-identification task, while remaining asymptotically faster than traditional grammar-based parsers.

1 Introduction

Research in statistical parsing has made significant progress on recovering the kinds of annotations found in treebank-annotated corpora (Collins, 2003; Petrov et al., 2006), but in practice parsing is rarely an end in itself. Instead, parsing must be integrated with other forms of markup (part-of-speech tagging, named-entity recognition, coreference resolution) in order to perform complete end-to-end tasks like question answering. Historically, independent pursuit of these tasks has often been accompanied by the belief that improvements on a measure intrinsic to parsing (for our domain, often F1 on the test section of the Wall Street Journal) will accurately reflect improvements on an end task when incorporated into a complete system.

Nevertheless, errors propagate in NLP pipelines, and the need for a consistent analysis across several layers of linguistic structure has motivated the development of joint models in which uncertainty is shared between individual model components. Previous work has applied joint inference to parsing and named-entity recognition (NER), the task pursued in this paper, by augmenting the grammar with specialized non-terminals which couple syntactic and NER labels to represent an analysis over both domains (Finkel and Manning, 2009). This coupling proved to be beneficial to both tasks, increasing parsing performance by 0.47 F1 and NER performance by 3.27 F1 on average over the independently-trained models.

Proceedings of COLING 2012: Technical Papers, pages 1995–2010,
COLING 2012, Mumbai, December 2012.

Yet there are many reasons one might be concerned with this approach. First, it is a problem-specific formulation of a joint problem, and not necessarily extensible to problems that require a looser or more flexible coupling between the two problem domains. For instance, the presence of an NER span indicates the presence of a noun phrase in the corresponding syntax. NER spans are therefore a subset of syntactic constituent spans, and the two problems can be represented with a single tree-structured derivation. However, a joint model of prosody and parsing would be difficult to capture with this approach, as boundaries of prosodic and syntactic spans are not required to overlap (Selkirk, 1984; Jackendoff, 2002, Ch. 5, p. 118-123).

Second, formulating joint inference as context-free parsing also imposes a computational burden at a particularly susceptible point in the algorithm. Consider the CKY algorithm's complexity of $O(|G|n^3)$, where $|G|$ is the size of the grammar and n the length of the sentence. While algorithms are generally compared in terms of their asymptotic complexity (here the property of being a cubic-time algorithm), in many natural language problems the grammar constant is by far the most computationally expensive component. In the grammar augmentation approach to joint inference, the grammar constant increases as the product of the different domains' label sets.

Perhaps the most limiting aspect of this approach is that it makes assumptions over what types of structures are permissible, which may force the model structure of one problem into less intuitive designs based on the structure of another. For instance, named entity recognizers have traditionally been developed using sequence models, but some of the sequential model structure is lost when conforming to a grammar-based notion of joint inference.[1]

All of the aforementioned criticisms will only become more pertinent when joint inference in NLP begins to scale toward coupling the three or more problems in larger end-to-end systems. We therefore present in this paper a more generalized approach to joint inference via combinatorial factor constraints in factor graphs. In particular we describe a special-purpose combinatorial factor for constraining syntactic variables to a valid constituent bracketing, providing an efficient and flexible method for representing constituent syntax in graphical models. We first demonstrate that on its own this simple parser setup is competitive in accuracy with more complex models at the standard task of recovering treebank annotation. Then, we demonstrate the composability of the factor graph approach by coupling the syntactic representation to a semi-Markov NER model and performing inference jointly.

2 Parsing without a Grammar

While the dominant approaches to constituency parsing almost always depend on an explicit grammar, efficiently representing grammatical parsing in frameworks as general as factor graphs is very difficult. In this section we will discuss methods for representing constituency-style parsing constraints in a factor graph. This general-purpose syntactic representation can then be flexibly coupled to other tasks, and trained based upon common inference and learning methods.

A factor graph is a type of graphical model represented as a bipartite graph composed of variables, whose values represent the state of the world, and factors, which specify potentials over the belief of a particular state. For learning methods that require the marginal beliefs to compute a gradient, factor graphs provide an efficient way of computing such marginals via message passing inference algorithms.

[1] In this particular case, one could compose the context-free grammar with the finite-state model—causing the grammar to grow even more. Similar problems arise in combining syntactic machine translation models with n-gram language models (Vaswani et al., 2011).

The difficulty in representing grammars with factor graphs is that the complexity of inference in such a graph scales exponentially in the size of the largest clique. Naively representing a weighted grammar in Chomsky normal form requires variables to be densely connected with ternary factors [2] to represent the application of a weighted grammar rule. This is not only computationally problematic (on a per-iteration of inference basis), but could also further complicate convergence when performing belief propagation, an efficient message passing inference algorithm (Pearl, 1988), in the now exceedingly loopy graph.

If we were to back off from a desire to represent grammatical parsing in a factor graph, we could impose a constraint which prohibits crossing brackets by examining variables in a pairwise manner and assigning zero probability to configurations in which two conflicting variables were both on, and this would be possible with a quartic number of constraining factors. But while both approaches would capture the gist of a useful constraint, they are both computationally inefficient and will still fail to prohibit all structures that are not valid trees.

Instead we introduce a special-purpose combinatorial factor, CKYTree. It was observed previously (Smith and Eisner, 2008) that the outgoing messages from such combinatorial factors to a variable could be computed from the factor's posterior beliefs about that variable, thus defining an interface for inserting special purpose logic within the standard inference algorithm. Instead of representing a tree constraint externally in the structure of the model, we can encapsulate similar logic in this factor where the computation can be done more efficiently with variants of standard dynamic programming algorithms (Figure 1). Previous work has applied this technique to non-projective and projective dependency parsing (Smith and Eisner, 2008; Naradowsky et al., 2012), and a similar belief propagation approach has been used to enforce well-formed productions in CCG parsing (Auli and Lopez, 2011).

More specifically, inputs to this algorithm are the span weights $u(i,j)$. As in earlier dependency parsing work, these weights are derived from the ratio of messages coming in from the $Span$ variables:

$$u(i,j) = \frac{m_{Span(i,j)\rightarrow \text{CKYTree}}(\text{true})}{m_{Span(i,j)\rightarrow \text{CKYTree}}(\text{false})}$$

After running this inside-outside algorithm in $O(n^3)$ time, we calculate the $O(n^2)$ outgoing messages from CKYTree:

$$m_{\text{CKYTree}\rightarrow Span(i,j)}(\text{true}) = g(i,j)$$
$$m_{\text{CKYTree}\rightarrow Span(i,j)}(\text{false}) = 1 - u(i,j) \cdot g(i,j)$$

Here $g(i,j)$ is the gradient of the sum of the weights of all trees with respect to the input weight $u(i,j)$. Using the familiar inside β and outside α quantities, we can write this as:

$$g(i,j) \stackrel{\text{def}}{=} \frac{\partial \beta(0,n)}{\partial u(i,j)} = \frac{\alpha(i,j)}{\beta(0,n)}$$

[2] Ternary factors connect three variables, in this case a parent span variable and two contained and adjacent child span variables.

Algorithm 1 Bracket inside algorithm

function Inside(u, n)
  for $w \leftarrow 2..n$ **do**
    for $i \leftarrow 0..(n-w)$ **do**
      $k \leftarrow i + w$
      $s \leftarrow \mathbf{0}$
      for $j \leftarrow (i+1)..(k-1)$ **do**
        $s \overset{\oplus}{\leftarrow} \beta(i,j) \otimes \beta(j,k)$
      end for
      $\beta(i,k) \leftarrow s \oplus u(i,k)$
    end for
  end for
  return β
end function

Algorithm 2 Bracket outside algorithm

function Outside(u, β, n)
  for $w \leftarrow n..2$ **do**
    for $i \leftarrow 0..(n-w)$ **do**
      $k \leftarrow i + w$
      for $j \leftarrow (i+1)..(k-1)$ **do**
        $\alpha(i,j) \overset{\oplus}{\leftarrow} \alpha(i,k) \otimes \beta(j,k) \otimes u(i,k)$
        $\alpha(j,k) \overset{\oplus}{\leftarrow} \alpha(i,k) \otimes \beta(i,j) \otimes u(i,k)$
        $g(i,k) \overset{\oplus}{\leftarrow} \alpha(i,k) \otimes \beta(i,j) \otimes \beta(j,k)$
      end for
    end for
  end for
  return g
end function

Figure 1: Bracket inference algorithms are special cases of the familiar inside and outside algorithms for PCFGs (Baker, 1979) with a different "grammar-rule" weight $u(i,j)$ for each span.

2.1 Bracket Model

Having introduced the necessary computational framework, we can now present our most basic model of syntax: a factor graph for predicting unlabeled, projective binary constituency trees without any representation of a grammar. We model the possible parses of an n-word sentence with the following factor graph:[3]

- Let $\{Span(i,j) : 0 \leq i < j \leq n\}$ be $O(n^2)$ boolean variables such that $Span(i,j) = \textsf{true}$ iff there is a bracket spanning i to j.[4]
- Let $\{\text{Brack}(i,j) : 0 \leq i < j \leq n\}$ be $O(n^2)$ unary factors, each attached to the corresponding variable $Span(i,j)$. These factors score the independent suitability of each span to appear in an unlabeled constituency tree.
- Let CKYTree be a global combinatorial factor attached to all the $Span(i,j)$ variables. This factor contributes a factor of 1 to the model's score if the span variables collectively form a legal, binary bracketing and a factor of 0 otherwise. It enforces, therefore, a hard constraint on the variables. All outgoing messages from this factor are computed simultaneously by the outside algorithm described above in §2.

This comprises the core of our parsing model, which couples local span-oriented factors with rich features to a combinatorial factor that guarantees the global structure is a valid constituent tree. Though it is lightweight, with a complexity of $O(n^3 + n^2)$ in comparison to $O(|G|n^3)$ of a traditional CKY parser, it is not possible to identify which predicted spans are introduced through binarization (because constituents are not labeled), and thus limits the validity of parser comparison. Instead of evaluating this model in isolation, we immediately augment it with factors and variables to model *labeled* constituency trees.

[3] Variables are denoted by italicized names, factors by small capitals.

[4] In practice, we do not need to include variables for spans of width 1 or n, since they will always be true.

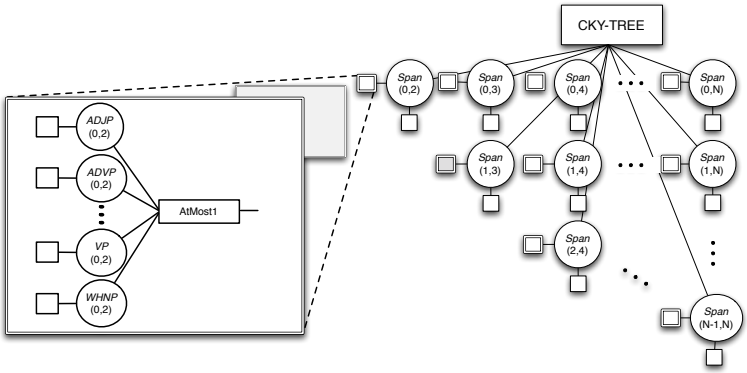


Figure 2: A graphical depiction of Bracket Model and Labeled Bracket Model. The components of the Bracket Model – *Span(i,j)* variables, corresponding Brack(i,j) factors (shown as single-bound rectangles), and the constraining tree factor – are represented on the right. The Labeled Bracket Model extensions include the AtMost1 factors on each span variable (shown as double-bound rectangles), which coordinate to select just one label from the set of nonterminals (illustrated left).

2.2 Labeled Bracket Model (LBM)

Extending the bracket model to incorporate a set of labels L is as simple as connecting each $Span(i, j)$ variable to a set of $Label$ variables via a factor that ensures at most one label is present:

- Let $\{Label(i, j, \lambda) : \lambda \in L$, and $0 \leq i < j \leq n\}$ be $O(|L|n^2)$ boolean variables such that $Label(i, j, \lambda) = \text{true}$ iff there is a bracket spanning i to j with constituent label λ.

- Let $\{\text{AtMost1}(i, j) : 0 \leq i < j \leq n\}$ be $O(n^2)$ factors, each coordinating the set L of possible nonterminal variables to the $Span$ variable at each i, j tuple, allowing a label variable to be true iff all other label variables are false and $Span(i, j) = \text{true}$.

One contribution of this paper is the exploitation of logical factors to proxy multinomial distributions. In this case the AtMost1 factors coordinate sets of label variables with a boolean indicator variable, capturing the same semantics as the multinomial while, most importantly, partitioning the values themselves into separate boolean variables. This is a necessary design choice for easily extending the model with grammatical rules as discussed in Section §6.

Asymptotically this is still an advantageous decomposition, yielding a complexity of $O(n^3 + |L|n^2)$, though the label set must also include a separate nonterminal for labels introduced through binarization of the original grammar, so predicted spans labeled with these binarized symbols can be removed when constructing the final parse tree. While there is some freedom in this choice and a traditional CNF transformation often introduces a binarized variant of each constituent, we find no appreciable benefit from having more than one binary symbol.

2.3 Inference

With the exception of combinatorial factors, which have their own specialized propagators as described above, inference is performed using the standard sum-product algorithm (Pearl, 1988), also known as *belief propagation* (BP). Each node in the graph sends and receives messages of two types:

A message from a variable node v to a factor node u is the product of all the messages from the factor nodes connected to v excluding u itself.

$$m_{v \to u}(x_v) = \prod_{u* \in N(v)/u} m_{u* \to v}(x_v)$$

Similarly, a message from a factor node u to a variable node v is the product of the factor and all of the messages from neighboring variables, marginalizing over all neighboring variables with the exception of x_v.

$$m_{u \to v}(x_v) = \sum_{x'_u : x'_v = x_v} f_u(X'_u) \prod_{v* \in N(u)/v} m_{v* \to u}(x'_{v*})$$

In cases where u is a combinatorial factor, its own combinatorial algorithm is executed instead of the standard update, but inference otherwise continues without change.

Both the bracket and label models form non-loopy graphs and, in a manner analogous to the forward-backward algorithm on chains, converge in two iterations of message passing. The robustness of this approach to joint inference with syntax is that regardless of the model structure that may be created for other tasks or couplings, the inference algorithm can remain the same. If loops arise in the graph then the convergence guarantee is lost, but in practice we find that loopy BP, which is fundamentally the same algorithm run on cyclic graphs, often converges within several iterations for joint parsing and NER tasks.

2.4 Unary Re-writes

While our motivation here is to present a syntactic representation which can be used to the benefit of a separate NLP end task, we also assess the performance of the parser independently, and must therefore address the rather tangential issue of unary rewrites. In a context-free grammar, internal unary productions—nonterminals with a single nonterminal child—fall out naturally from the recursive definition of the model, but in our span-factored formulation they must be handled separately. Similar work in conditional random field parsing (Finkel et al., 2008) collapses unary rewrite chains to single augmented rules while prohibiting multiple rewrites from occupying the same span, but this comes at the cost of additional complexity during decoding. Other work has attempted to separately predict leaf-level unary rewrites, albeit for the purpose of improving decoding speed (Bodenstab et al., 2011).

We follow the latter approach, pruning unary rewrites from the data entirely (replacing them with their parent constituent) and training a separate log-linear classifier for reinserting unary rewrites into the trees produced by the parser. Because the distribution of unary rewrites is so overwhelmingly concentrated in the nodes immediately governing the leaves of the tree, we focus our efforts solely on predicting these unary rewrites, accepting the performance hit of not predicting any which may appear in the body of the tree (comprising between 11% and 15% of all unary spans newswire sections of OntoNotes).

3 Named Entity Model

To demonstrate the usefulness of this syntactic representation in joint inference tasks, we choose to evaluate primarily on the end task of named entity extraction. In contrast to previous work, we avoid augmenting the grammar with special non-terminals and construct the model once again within the context of a factor graph. This choice allows us to emphasize the sequential information that historically has produced state-of-the-art performance.

As with parsing, however, incorporating some state-of-the-art models is not a trivial task. Consider for instance a semi-Markov conditional random field (semi-CRF) model (Sarawagi and Cohen, 2004). The context-rich nature of these models is very difficult to capture within the top-down derivations of a tree-based decoder for joint inference. Even in the general framework of a factor graph, representing these structures efficiently in a manner compatible with the rest of the joint architecture presents a challenge. However, we can once again encapsulate this logic within one combinatorial factor and connect it in a global fashion to all variables corresponding to NER spans. First, we assume a named entity variable set analogous to the LBM (denoted $NER\text{-}Span$ and $NER\text{-}Label$ and behaving similarly).

- Let Semi-CRF be a combinatorial constraint connected to all $NER\text{-}Span(i, j)$ variables. The factor implements a weighted Semi-CRF with a maximum span width μ, as described in (Sarawagi and Cohen, 2004), over the log-odds of each span variable's boolean values. Unlike the valid-bracketing constraint imposed by the CKYTree factor, this effect can be achieved with a polynomial number of binary factors. The $O(\mu^2 n^2)$ such factors would lead to inefficient inference in a very loopy factor graph. In all experiments $\mu = 10$.

4 Joint NER and Parsing

Perhaps the main advantage of casting constituency parsing in terms of a factor graph is the ease with which the model can be extended to improve separate but related task. To illustrate this, we couple our LBM constituency parser to the span-based model of named-entity recognition with an additional type of specialized factor:

- Let $\{\text{NER-Nand}(i, j) : 0 \leq i < j \leq n; 1 < j - i \leq \mu\}$ be a set of at most $O(n^2)$ factors coordinating syntactic $Span(i, j)$ and named entity span variable $NER\text{-}Span(i, j)$, multiplying in 1 unless both variables are on, in which case it multiplies a connective potential $\phi(i, j)$ derived from its features. Intuitively the joint model might learn features weights such that $\phi(i, j) > 1$, i.e., constituents and NER spans are more likely to be coterminous. The number of these coordinating factors is constrained to the number of NER span variables, subject to the maximum span-width $\mu = 10$.

These special purpose factors allow the model to learn how to best coordinate the sub-problems, adding an additional layer of flexibility to the joint architecture. For this particular domain, features that are useful for distinguishing noun phrase spans from other spans are good candidates, as essentially all named entities correspond to noun phrases in the syntax.

5 Experiments

We have presented an argument for why the combinatorial factor graph approach to joint inference can be considered a very general and principled framework for representing and reasoning over

	Template	Instantiated
General	$\{j-i, i, j\}$	LEN-3, START-13, END-16
CCM $CCM(i,j)$	$\{POS(i-1)+POS(j+1),$ $POS(i)+...+POS(j)\}$	OUTER-RB-IN, INNER-DT-VBG-NN
Unigram $U(i)$	$\{Word(i), POS(i),$ $Word(i)+POS(i), Capitalization?(i)$	WORD-shining, TAG-VBG, WORD-POS-shining-VBG, CAP-FALSE
Bigram $B(i,j,w)$	$U(i)\times U(j), U(i)\times U(j+w), U(i-w)\times U(j)$ $POS(i-w)+...+POS(i+w)\times U(j), ...\}$	SPOS0-EPOS0-DT-NN, SPOS+1-EWORD0-VBG-symbol, BG-EW-VBD-RB-DT-VBG-NN-symbol, ...
Variation $V(i,j,w)$	$\{ ContainsPOS(l), Contains(w),$ $Tagset(i,j), ConjContains(i,j,w)\}$	CONTAINS-POS-VBG, CONTAINS-WORD-shining TAGSET- DT-VBG-NN, CONTAINS-WORD-SHINING, ...

DT	JJ	NN	IN	WP	VBD	RB	|[DT	VBG	NN]|	IN	DT	NN
the	final	chapter	of	what	was	once	|[a	shining	symbol]|	of	the	future.

Table 1: Common feature templates. Instantiations of these templates on the 13-16 span of the example sentence are also provided. CCM refers to the features used by the constituent-context model (Klein and Manning, 2002).

joint models. In this section we aim to demonstrate that these benefits are not merely conceptual or aesthetic in nature, but translate to practical performance improvements in the model. In the following sections we first demonstrate that parsers based on this architecture outperform previous CRF parsers, and that they provide both an asymptotic advantage over state-of-the-art parsers in decoding speed and an attractive compromise between speed and performance. We then show that the semi-CRF NER model is a comparable baseline to previous standalone models, and that it improves upon previous results when trained jointly.

5.1 Data

We primarily evaluate all of our model configurations on the same data set: a selection of six corpora drawn from the English broadcast news section of the OntoNotes 2.0 dataset (Hovy et al., 2006). The data are partitioned to achieve an approximately 3:1 ratio between training and test sets. This is an exact reproduction of the partitioning found in (Finkel and Manning, 2009), where detailed corpus statistics may be found. As in that work, we remove empty leaf nodes, coarsen nonterminal labels (NP, not NP-PRD), and filter out sentences longer than 40 words. In supplementary parsing experiments we make us of the OntoNotes Wall Street Journal Corpus distribution, using the standard train/test split and sentences with between 3 and 40 words.

5.2 Features

For each of the models presented here, every boolean variable mentioned has a corresponding unary factor representing its likelihood of being true. This leads to a wide-variety of features, depending on the variable's semantics.

For parsing we rely on features comparable to those used in edge-factored dependency parsing (McDonald et al., 2005), consisting of combinations of unigram features (word, part-of-speech, capitalization, and presence of punctuation) between the tokens at or near the span boundaries, including tokens immediately outside and inside of the span. This allows us to capture very strong lexical indications of a span, such as the presence of a comma immediately outside the start and end of the indices. We also look for the occurrence of particular tags anywhere within the span, which might signify that a constituent should not span those indices unless it is sufficiently large. A previous generative model of span-based grammar induction (Klein and Manning, 2002) considered the probability of a span to depend on the part-of-speech tags of the two words immediately outside of it, and on the conjunction of the tags within it. For spans that are sufficiently small (here $w < 10$),

the part-of-speech tags of all words inside of the span are concatenated. Examples of these feature sets are provided in Table 1. The reliance on part-of-speech tags as features is afforded through the use of an existing maxent part-of-speech tagger (Toutanvoa and Manning, 2000) which allows us to treat part-of-speech tags as observed while maintaining a fair comparison to other systems which also do not utilize gold part-of-speech tags.

Features for the unary classifier portion of the model also consist of the same unigram features (word, tag, capitalization, etc.), but are taken over 5-word windows from the tokens of the sentence.[5]

Features for NER spans are generally identical to those used for syntactic spans, while the feature sets specifying span labels are much more lexically-oriented. In addition to a small set of contextually-directed information we look primarily at word shape; character n-grams (windows of 2 to 5); capitalization; normalization; regular expressions for initials, numerics, times, and Roman numerals; and membership in a set of lexicons for ordinals, days of the week, months, scientific units, common names, honorifics, and stop words.

5.3 Parsing Results

We present the results of the LBM on labeled parsing, largely in comparison to similar work using CRF parsing with chart decoding (F&M'09). The LBM significantly outperforms the previous standalone approach in all data sets, yielding improvements of up to over nine points in F1 over this model (Table 2). Though the models are generally quite similar, the large performance discrepancy does raise the question of whether or not this could be due solely due to the unique hybrid approach to parsing represented by the LBM over traditional methods, where local factors observe much more information than can be traditionally utilized while still having strong top-down structural constraints at play. It is possible that some of this gap is merely due to different feature sets, but the feature set used here is by no means an exhaustive or fine-tuned set, comparable in size to the features often used by graph-based dependency parsers.

In comparison to other widely available parsers, which are constructed using a vertical and horizontal Markov window of 1 where applicable, the LBM does not achieve absolute state-of-the-art, but still compares favorably to these established models on this data set, outperforming both configurations of the Stanford Parser. It is important to note that these factor graph representations of syntax are still extensible enough to incorporate non-terminal or grammar refinement, or reranking, to further improve their performance. Our purpose in this section is simply to present them as a suitable alternative to existing parsers and prove that the syntactic predictions upon which other tasks can be jointly modeled are themselves quite accurate.

5.4 Decoding Efficiency

We compare against reference implementations of the PCFG parser (Klein and Manning, 2003) and lexicalized parser included in the Stanford Parser distribution[6], and train up comparable grammars for all models from sections 2–21 of the Wall Street Journal Treebank corpus (Marcus et al., 1993).

Efficiency of the LBM is quite competitive with these models (Figure 3). When evaluated in its standard configuration, with a full nonterminal set for each span, the LBM edges out the PCFG

[5] It is also possible to derive features from the parse tree, using the constituent labels collected by traversing the tree along the root-to-leaf path. This results in an average of 4.63% F1 improvement on the unary prediction task over linear features, but requires that the tree be decoded prior to generating unary productions. For convenience we predict the tree and its unary productions simultaneously.

[6] V. 1.68, `http://nlp.stanford.edu/software/lex-parser.shtml`

		Bracket Evaluation					Labeled Evaluation		
Data	Model	Prec	Recall	F1	CB	NC	Prec	Recall	F1
ABC	LBM	82.80	79.65	81.20	1.72	50.78	79.75	76.72	78.20
	+Rules	–	–	–	–	–	80.66	77.92	**79.26**
	F&M '09	–	–	–	2.28	46.88	70.18	70.12	70.15
CNN	LBM	86.40	83.12	84.73	0.96	67.41	83.30	80.14	81.69
	+Rules	–	–	–	–	–	84.66	80.88	**82.72**
	F&M '09	–	–	–	1.11	70.06	76.92	77.14	77.03
MNB	LBM	80.77	76.18	78.41	1.40	59.26	76.98	72.61	74.73
	+Rules	–	–	–	–	–	79.09	73.05	**75.96**
	F&M '09	–	–	–	1.88	59.03	63.97	67.07	65.49
NBC	LBM	80.77	77.37	79.04	1.41	49.66	74.81	71.67	73.20
	+Rules	–	–	–	–	–	76.94	72.34	**74.57**
	F&M '09	–	–	–	2.67	48.92	59.72	63.67	61.63
PRI	LBM	85.01	82.15	83.56	1.44	57.40	82.70	79.92	**81.29**
	+Rules	–	–	–	–	–	81.90	80.44	81.17
	F&M '09	–	–	–	1.72	56.70	76.22	76.49	76.35
VOA	LBM	85.71	81.96	83.80	1.63	43.34	83.55	79.89	81.68
	+Rules	–	–	–	–	–	83.07	82.69	**82.88**
	F&M '09	–	–	–	2.44	38.89	76.56	75.74	76.15
WSJ-Mod	LBM						84.88	80.33	82.54
	+Rules								84.32
	Stanford-PCFG						80.71	79.86	80.28
	Stanford-Factored						81.64	81.65	81.64
	Berkeley						86.61	85.81	**86.21**

Table 2: Labeled model performance. The feature-rich Labeled Bracket Model (LBM) provides significant gains over previously published CRF parsing scores. Unlabeled parsing was evaluated against the true, non-binarized bracketings. The +Rules model refers to the grammar-enhanced version of the model described in Section §6. As in previous work, we find that in the joint setting we find only marginal improvements in parsing F1, and abstain from providing them for the sake of clarity.

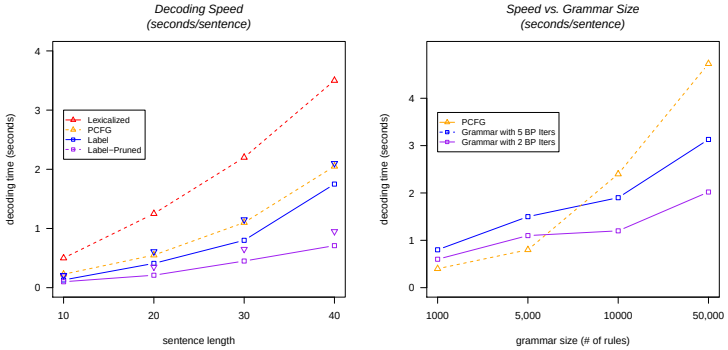


Figure 3: Decoding speeds. Comparing the performance of the labeled model, in both its standard and pruned configuration, with standard PCFG and lexicalized parsing baselines (left), the factor graph models generally decode faster than their counterparts. While featurization cost (inverted triangle annotation) hinder the standard configuration, the pruned model is quite fast. When comparing the grammar model to the PCFG model, and increasing the size of the grammar (right), the additive grammar term provides a clear and substantial benefit as the grammar size increases.

model, the faster of the two baseline systems, but only marginally so. Additionally we indicate the contribution that feature extraction time makes to overall decoding time by the carets above the factor graph parser points. In the case of the standard LBM this removes any performance benefit provided by the factor graph implementation over the PCFG. However, when combined with pruning and reducing the binarized nonterminal set to use one, the LBM becomes a very attractive choice, decoding almost twice as fast on 40-word sentences as the PCFG, and three times faster than the lexicalized model.

The bracket model cannot offer competitive accuracy when compared to the other models and has limited application given its output is only a set of projective spans. However, coupled with an appropriate task it does offer exceptional parsing speed, decoding length 40 sentences at more than 10 per second on our test system, and may be useful as a component in select joint modeling tasks.

5.5 NER Results

We again evaluate on the OntoNotes corpus, which contains both syntactic and named entity annotation, training our model with 10 iterations of stochastic gradient descent (SGD), each with 5 iterations of BP, evaluating on the more difficult full 16-entity label set. Results are presented in Table 3.

Both baseline systems, trained without any syntactic information, perform comparably on average. This is somewhat surprising given the sequential constraints of our NER model, but likely also due in part to the less difficult 4-label NER task reported in previous results. However, we see stronger than average gains by coupling the models and using joint inference. For instance, the factor graph model outperforms the previous best results on four of the six data sets, but improves over it by a much larger margin than the F&M system does on the two data sets it performs best in.

		Prec	Recall	F1	F&M '09
ABC	NER Only	76.4	67.8	72.13	74.51
	NER Joint	76.5	71.3	73.93	**74.91**
CNN	NER Only	75.2	75.0	75.07	75.78
	NER Joint	79.2	79.9	**79.56**	78.70
MNB	NER Only	68.9	70.1	69.50	62.21
	NER Joint	72.7	71.3	**72.02**	66.49
NBC	NER Only	69.5	61.8	65.69	63.90
	NER Joint	73.3	67.0	**70.18**	67.96
PRI	NER Only	80.3	82.6	81.50	83.44
	NER Joint	86.9	86.6	**86.71**	86.34
VOA	NER Only	81.4	74.8	78.11	79.23
	NER Joint	86.4	88.1	87.23	**88.18**

Table 3: NER baseline and joint model performance. Decoding jointly consistently improves the NER results across all corpora, providing the largest gains on corpora with the most data and best parser performance. In comparison to F&M '09, our joint model outperforms in all but two of the corpora.

6 Grammar Rules as Factors

Our final extension, the incorporation of grammar rules into the model, requires only the addition of rule factors that connect triples of labeled span variables:

- Let $\{\text{Rule}(i,j,k,X,Y,Z) : 0 \leq i < j < k \leq n; X,Y,Z \in L\}$, be a sparsely-applied set of ternary factors which coordinate across the $Label$ variables at spans (i,j), (j,k), and (i,k).

The difficulty in constructing a rule-based augmentation is not as much structural as it is about finding an efficient way to add rules to the model. It is easy to enumerate the $O(|L|n^2)$ potentials necessary for the LBM, but it becomes prohibitive to work with the $O(|L|^3 n^3)$ potentials that represent a complete grammar.

Figure 4 illustrates a rule factor constraining three $Label$ variables. Grammar rules don't function in the traditional way, where rules guide a set of allowable derivations. Instead, a Rule factor may be instantiated across any triplet of nonterminals to alter the local $Span$ and $Label$ beliefs. Each rule has a single primary feature: its string representation ($\mathsf{NP} \rightarrow \mathsf{NP\ VP}$), though we do also incorporate back-off features for smoother statistics ($\mathsf{X} \rightarrow \mathsf{NP\ VP}$, $\mathsf{NP} \rightarrow \mathsf{X\ VP}$). The ease with which the LBM can be augmented with rules is due to the factorization of the $Label$ variables into large sets of binary variables, instead of one high-dimensional multinomial, for each span.

To learn a sparse set of grammar rules, we used perceptron updates. After decoding a set of training sentences, we computed the difference between the rule applications in the hypothesized and true trees, and updated the weights of those productions accordingly. This creates grammars of roughly 1200 productions on both OntoNotes and WSJ and significantly outperforms the LBM results (Table 2).

Figure 3 illustrates the performance of the grammar model in comparison to a PCFG as we artificially increase the size of the grammar (sentences were fixed at length 40). Our model is unlexicalized and coarse-grained, so simply constructing a grammar by reading off the productions found in the WSJ treebank yields only 2751 rules. While the intent of developing this model is specifically for fast joint inference, lexicalizing the grammar or refining it via a split-merge procedure to bring

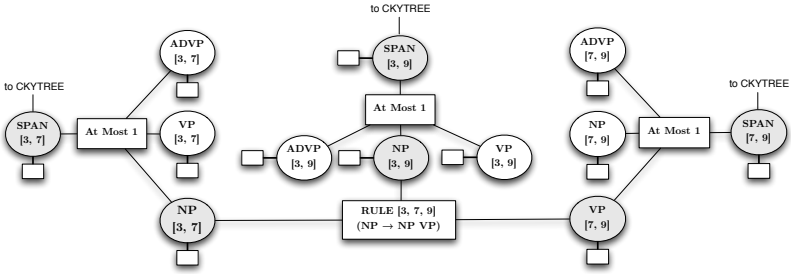


Figure 4: A Rule factor constraining three $Label$ variables. Not every span or constituent label must participate in a rule; the number of rules added on top of the labeled bracket model may be quite small. Rules act only to fix up small ungrammatical or dispreferred configurations that are more probable when looking solely at local span information.

performance to state-of-the-art levels solely on parsing increases the grammar size, often producing grammars comprising millions of rules in the latter case. In these cases our model provides clear asymptotic benefits. As depicted in Figure 3 (right), even by 50,000 rules the grammar model has begun to decode significantly faster than the PCFG, with a strong asymptotic advantage with grammars in excess of 10k rules.

All our grammar model performance experiments use five iterations of BP (less if the model converges faster), but for comparison we also present decoding times if we ran the parser as if it were exact inference, and required only two.

6.1 Comparisons to Established Parsers

How do these models handle parsing on larger, more traditionally-sized corpora? On our modified WSJ data set (Table 2, bottom) the LBM performs surprisingly well in comparison to the widely used Stanford and Berkeley parsers (Finkel et al., 2008; Petrov and Klein, 2007), achieving an F1 of 82.54% despite having no representation of a grammar. Augmenting this model with 15 iterations of rule induction yields our largest improvement over the label model, increasing F1 by 1.78% to 84.32% and brings performance in line with state-of-the-art parsers.

Another consideration is our reliance on local potentials, and predictions made primarily based on the rich sources of information at, around, and within span boundaries. While joint tasks may at some point leverage more sophisticated tree annotations, many systems, such as named-entity recognizers, rely primarily on accurately identifying NP spans. Despite having lower labeled F1 than one of the three widely-used constituency parsers we evaluated, the LBM outperforms them all on NP detection (Table 4), making it unlikely that further NER improvements will be had by leveraging these much more sophisticated, but slower, models.

7 Conclusions and Future Work

We have shown how to decompose and represent constituency parsing in terms of variables in a dynamic Markov random field, decoding with general inference algorithms: belief propagation in the exact case for both labeled and unlabeled parsing without a grammar, and loopy BP when augmented with rules or coupled with an NER model. We have demonstrated the convenience

Model	NP-P	NP-R	NP-F1
LBM	90.07	90.02	90.42
+Rules	90.15	91.07	**90.60**
Stanford-PCFG	83.88	85.74	84.80
Stanford-Factored	85.07	87.83	86.43
Berkeley	89.17	90.96	90.06

Table 4: NP prediction results. While some established parsers outperform LBM in general parsing F1 (82.54 LBM vs. 86.21 Berkeley), LBM outperforms all evaluated parsers on a measure more reflective of its potential to many joint modeling scenarios, while remaining asymptotically faster.

and effectiveness of using these combinatorial syntactic representations in joint inference tasks, generally improving performance on named entity recognition over the previous state-of-the-art.

As a standalone parser, both the LBM and rule model provide some decoding efficiency benefit over PCFG and lexicalized baselines, while providing parsing results comparable with the best coarse-grammar, unlexicalized parsers. We also showed how labeling spans via a set of boolean label variables allows ternary factors to function as grammatical rules, not specifying an entire derivation but instead fixing up trees in situations where the LBM would otherwise err.

One natural extension to the parser portion of this work is to port over useful advances from the parsing literature, lexicalizing the grammar or refining the nonterminal set. However, it is also possible to exploit other aspects of the factor graph representation. Most pertinent to parsing performance, a factor graph allows for arbitrary constraints between variables—a potentially promising avenue for incorporating additional linguistic information (headedness, lexicalization, agreement) in unique and powerful ways. This approach would be particularly interesting for languages whose dependencies are hard to capture with the traditional independence assumptions made by PCFGs. The more inherently distributed structure of this model would also make it a good choice for parallelization.

8 Acknowledgements

We thank Jenny Finkel for help in obtaining the additional performance numbers for her system, and Benjamin Börschinger, Mark Johnson, and the anonymous reviewers for their helpful comments. This work was supported in part by the Center for Intelligent Information Retrieval and in part by Army prime contract number W911NF-07-1-0216 and University of Pennsylvania subaward number 103-548106. Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect those of the sponsor.

References

Auli, M. and Lopez, A. (2011). A comparison of loopy belief propagation and dual decomposition for integrated CCG supertagging and parsing. In *ACL*, pages 470–480.

Baker, J. K. (1979). Trainable grammars for speech recognition. In *Proceedings of the Acoustical Society of America*, pages 547–550.

Bodenstab, N., Hollingshead, K., and Roark, B. (2011). Unary constraints for efficient context-free parsing. In *ACL*, pages 676–681.

Collins, M. (2003). Head-driven statistical models for natural language parsing. *Comput. Linguist.*, 29(4):589–637.

Finkel, J. R., Kleeman, A., and Manning, C. D. (2008). Efficient, feature-based, conditional random field parsing. In *ACL*, pages 959–967.

Finkel, J. R. and Manning, C. D. (2009). Joint parsing and named entity recognition. In *HLT-NAACL*, pages 326–334.

Hovy, E., Marcus, M., Palmer, M., Ramshaw, L., and Weischedel, R. (2006). OntoNotes: The 90% solution. In *HLT-NAACL*, pages 57–60.

Jackendoff, R. (2002). *Foundations of Language: Brain, Meaning, Grammar, Evolution*. Oxford University Press.

Klein, D. and Manning, C. D. (2002). A generative constituent-context model for improved grammar induction. In *ACL*, pages 128–135.

Klein, D. and Manning, C. D. (2003). Accurate unlexicalized parsing. In *ACL*, pages 423–430.

Marcus, M. P., Marcinkiewicz, M. A., and Santorini, B. (1993). Building a large annotated corpus of English: The Penn Treebank. *Computational Linguistics*, 19:313–330.

McDonald, R., Pereira, F., Ribarov, K., and Hajic, J. (2005). Non-projective dependency parsing using spanning tree algorithms. In *HLT-EMNLP*, pages 523–530.

Naradowsky, J., Riedel, S., and Smith, D. A. (2012). Improving nlp through marginalization of hidden syntactic structure. In *Proceedings of the 2012 Conference on Empirical Methods in Natural Language Processing*, Jeju, Korea. Association for Computational Linguistics.

Pearl, J. (1988). *Probabilistic reasoning in intelligent systems: networks of plausible inference*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.

Petrov, S., Barrett, L., Thibaux, R., and Klein, D. (2006). Learning accurate, compact, and interpretable tree annotation. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 433–440, Sydney, Australia. Association for Computational Linguistics.

Petrov, S. and Klein, D. (2007). Improved inference for unlexicalized parsing. In *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; Proceedings of the Main Conference*, pages 404–411, Rochester, New York. Association for Computational Linguistics.

Sarawagi, S. and Cohen, W. W. (2004). Semi-Markov conditional random fields for information extraction. In *NIPS*, pages 1185–1192.

Selkirk, E. (1984). *Phonology and Syntax: The Relation Between Sound and Structure*. Cambridge: MIT Press.

Smith, D. A. and Eisner, J. (2008). Dependency parsing by belief propagation. In *EMNLP*, pages 145–156.

Toutanvoa, K. and Manning, C. D. (2000). Enriching the knowledge sources used in a maximum entropy part-of-speech tagger. In *EMNLP*, pages 63–70.

Vaswani, A., Mi, H., Huang, L., and Chiang, D. (2011). Rule markov models for fast tree-to-string translation. In *ACL*, pages 856–864.

Error Mining with Suspicion Trees: Seeing the Forest for the Trees

Shashi Narayan[1] Claire Gardent[2]
(1) Université de Lorraine/LORIA, Nancy, France
(2) CNRS/LORIA, Nancy, France
shashi.narayan@loria.fr, claire.gardent@loria.fr

Abstract

In recent years, error mining approaches have been proposed to identify the most likely sources of errors in symbolic parsers and generators. However the techniques used generate a flat list of suspicious forms ranked by decreasing order of suspicion. We introduce a novel algorithm that structures the output of error mining into a tree (called, suspicion tree) highlighting the relationships between suspicious forms. We illustrate the impact of our approach by applying it to detect and analyse the most likely sources of failure in surface realisation; and we show how the suspicion tree built by our algorithm helps presenting the errors identified by error mining in a linguistically meaningful way thus providing better support for error analysis. The right frontier of the tree highlights the relative importance of the main error cases while the subtrees of a node indicate how a given error case divides into smaller more specific cases.

Title and Abstract in Hindi

संदेह वृक्षों के साथ त्रुटि खननः वृक्षों की खोज जंगल में

हाल के वर्षों मे, प्रतीकात्मक पद-व्याखक तथा पद-उत्पादक की गलतियों के संभावित श्रोतों की पहचान के लिये कई त्रुटि खनन तकनीकें प्रस्तावित की गयी हैं। हालांकि प्रस्तावित तकनीकें संदेहात्मक उदाहरणों को एक साधारण सूची में उनके घटते संदेह के क्रम में प्रस्तुत करते हैं। हम यहाँ एक नयी तकनीक प्रस्तुत कर रहे हैं जो त्रुटि खनन के परिणामों को एक वृक्ष (संदेह वृक्ष) के रूप में गठित कर संदेहात्मक उदाहरणों के बीच के संबंधों को दिखलाता है। हम अपने तकनीक की उपयोगिता पद-उत्पादक की संभावित असफलताओं के श्रोतों को ढूँढने तथा विश्लेषण करके प्रस्तुत करते हैं। हम यह दिखाते हैं कि कैसे हमारे तकनीक द्वारा निर्मित संदेह वृक्ष त्रुटि खनन द्वारा पहचानीत गलतियों को भाषाशास्त्र के दृष्तिकोण से काफी अर्थपुर्ण तरीके से प्रस्तुत कर त्रुटि-विश्लेषण में काफी मददगार साबित होता है। संदेह वृक्ष का दाहिना सीमांत मुख्य त्रुटिपुर्ण उदाहरणों को उनके तुलनात्मक महत्ता के साथ प्रस्तुत करता है जबकि संदेह वृक्ष की शाखायें दिखातीं हैं कि कैसे एक त्रुटिपुर्ण उदाहरण छोटे-छोटे त्रुटिपुर्ण विशिष्ट उदाहरणों में बँटता है।

Keywords: Error Mining, Generation.

Keywords in Hindi: त्रुटि खनन, पद-उत्पादक.

Proceedings of COLING 2012: Technical Papers, pages 2011–2026,
COLING 2012, Mumbai, December 2012.

1 Introduction

In recent years, error mining approaches have been proposed to identify the most likely sources of errors (called, *Suspicious Forms*) in symbolic parsers and generators. (van Noord, 2004) initiated error mining on parsing results with a very simple approach computing the parsability rate of each n-grams in a very large corpus. The parsability rate of an n-gram $w_i \dots w_n$ is the ratio $P(w_i \dots w_n) = \frac{C(w_i \dots w_n | OK)}{C(w_i \dots w_n)}$ where $C(w_i \dots w_n)$ is the number of sentences in which the n-gram $w_i \dots w_n$ occurs and $C(w_i \dots w_n \mid OK)$, the number of sentences containing $w_i \dots w_n$ which could be parsed. In other words, the parsability rate of an n-gram is the proportion of sentences in which this n-gram occurs and for which parsing succeeds. An n-gram then, is a suspicious form if it has a low parsability rate.

(van Noord, 2004)'s approach was extended and refined in (Sagot and de la Clergerie, 2006), (de Kok et al., 2009) and (Gardent and Narayan, 2012) as follows. (Sagot and de la Clergerie, 2006) defines a suspicion rate for n-grams which takes into account the number of occurrences of a given word form and iteratively defines the suspicion rate of each word form in a sentence based on the suspicion rate of this word form in the corpus. Further, (de Kok et al., 2009) extends this iterative error mining to n-grams of words and POS tags of arbitrary length. And (Gardent and Narayan, 2012) extends (van Noord, 2004)'s approach to mine for suspicious subtrees rather than n-grams.

An important limitation shared by all these error mining approaches is that their output is a flat list of suspicious forms ranked by decreasing order of suspicion. There is no clear overview of how the various suspicious forms interact and as a result, the linguist must "hop" from one error case to another instead of focusing on improving sets of related error cases. In short, the output of these error mining approaches lacks structure thereby making it difficult to handle errors in a linguistically meaningful way.

To overcome this shortcoming, we propose an algorithm which structures the output of error mining into a *suspicion tree* making explicit both the ranking of the main distinct error cases and their subcases. The suspicion tree is a binary tree structure whose internal nodes are labelled with suspicious forms and whose leaf nodes represent the clusters of error mined data grouped according to the suspicious forms characterizing their elements. Like in a decision tree, each cluster in the suspicion tree is characterized by the set of attributes (suspicious forms) labelling its ancestors; and the tree itself represents a disjunction of mutually exclusive error cases.

We illustrate the impact of our error mining algorithm on error analysis by applying it to detect and analyse the most likely sources of failure in a surface realiser developed to participate in the Surface Realisation Shared Task (Belz et al., 2011); and we show how this error mining algorithm permits improving the surface realiser.

The paper is structured as follows. We start (Section 2) by introducing our error mining algorithm. In essence, this algorithm adapts (Quinlan, 1986)'s ID3 algorithm to build a suspicion tree such that the clusters obtained group together sets of input data that share similar sources of failure (called *suspicious forms*); and the attributes labelling these clusters are the suspicious forms indicating which are these most likely causes of failure. In Section 3, we show how this error mining algorithm helps improving a surface realiser executed on the input dependency trees provided by the Surface Realisation (SR) Task challenge. Section 4 concludes with pointers for further research.

2 Building Suspicion Trees

In this section, we introduce the *suspicion tree algorithm* and discuss its complexity.

2.1 The Suspicion Tree Algorithm

As mentioned above, our error mining algorithm resembles (Quinlan, 1986)'s ID3 decision tree learning algorithm, in that it recursively partitions the data by first, selecting the attribute (here, a suspicious form) that best divides the data into more homogeneous subsets (*attribute selection*) and second, using this attribute to split the data into two subsets, a subset containing that attribute and a subset excluding that attribute (*dataset division*).

In what follows, we define the metric used to recursively select a suspicious form and partition the data, namely the *Suspicion Score* metric. We specify the termination conditions. We illustrate by means of examples how suspicion trees help structure the output of error mining. And we contrast the suspicion tree algorithm with (Quinlan, 1986)'s ID3 decision tree learning algorithm.

The Suspicion Score Metrics. Let $\mathbb{D}$ be the dataset to be error mined and $\mathbb{F}$ be the set of attributes used to partition the data. Here, $\mathbb{D}$ is a set of dependency trees provided for the Surface Realisation Task by the Generation Challenge; and $\mathbb{F}$ is the set of subtrees of $\mathbb{D}$ whose frequency is above a given threshold. Following (Gardent and Narayan, 2012), we use a complete and efficient Hybrid Tree Miner algorithm (Chi et al., 2004), to compute the set of subtrees that are present in $\mathbb{D}$.

Let $\mathbb{D}$ be divided into two disjoint sets: PASS (*P*) is the set of instances $t^P \in \mathbb{D}$ for which the processing system (e.g., a parser or a generator) succeeds; and FAIL (*F*) is the set of instances $t^F \in \mathbb{D}$ for which the system fails. Given these two sets, the **suspicion score** $S_{score}(f)$ of a form $f \in \mathbb{F}$ is then defined as follows:

$$S_{score}(f) = \frac{1}{2}(\text{Fail}(f) * \ln count(f) + \text{Pass}(\neg f) * \ln count(\neg f))$$

Intuitively, this metric captures the degree to which a form is associated with failure: it is high whenever a form f is often present in data associated with failure (high *F(ail)-Suspicion*, Fail(f)) and/or when it is often absent in data associated with success (high *P(ass)-Suspicion*, Pass($\neg f$)).

The **F-Suspicion rate of** f is defined as the proportion of cases where f occurs in an instance t for which the processing system fails:

$$\text{Fail}(f) = \frac{count(f|\text{FAIL})}{count(f)}$$

$count(f)$ is the number of instances containing f and $count(f|\text{FAIL})$ is the number of instances containing f for which processing failed.

Conversely, the **P-Suspicion rate of a form** f is defined as the proportion of cases not containing f and for which processing succeeds ($count(\neg f)$ is the number of instances where f is absent and $count(\neg f|\text{PASS})$ is the number of instances not containing f for which processing succeeds):

$$\text{Pass}(\neg f) = \frac{count(\neg f|\text{PASS})}{count(\neg f)}$$

Attribute Selection, Dataset Division and Termination. The suspicion tree algorithm selects at each step of the tree building process, the form f with highest suspicion score i.e. the form such that, in the current dataset, most instances that contain f fail to be processed; and most instances that excludes f lead to successful processing.

Based on this selected f, the current dataset is divided into two subsets: the set of instances which contain f and the set of instances which exclude f.

The form selection and dataset division process are called recursively on the new subsets until (i) the obtained set of instances is fully homogeneous (all instances in that set lead to either successful or unsuccessful processing); (ii) all forms have been processed; or (iii) the depth upper bound is reached (see below).

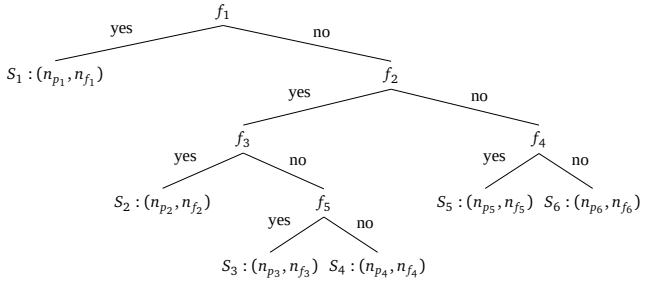


Figure 1: An example *Suspicion Tree*. Internal nodes are labeled with suspicious forms and leaves indicate the number of instances in the current data set S_i for which processing succeeds (n_{p_i}); and for which processing fails (n_{f_i}). When the sources of errors are clearly identifiable, n_{p_i} will be low, n_{f_i} will be high and the rightmost leaf (f_4) will have a low n_{f_i}.

Example. Figure 1 shows an abstract suspicion tree which illustrates how suspicion trees help structuring the output of error mining. The right frontier highlights the relative importance of the main distinct error cases while subtrees indicate how a given error case divides into smaller more specific cases. The branches of the tree also indicate the combinations of forms that frequently cooccur in failure cases.

More specifically, the root f_1 of this *suspicion tree* is the most suspicious form present in the corpus $\mathbb{D}$. Starting from the root, following the edges with label "no" (the *right-frontier* of the tree i.e., f_1, f_2 and f_4) yields the ranked list of suspicious forms present in $\mathbb{D}$ by decreasing order of suspicion. Following branches yields datasets labeled with sets (conjunctions) of suspicious forms. For example, the set S_2 with n_{p_2} of pass instances and n_{f_2} of failed instances has f_2 and f_3 as their top ranked suspicious forms. The *suspicion tree* also displays the relative ranking of the suspicious forms. For example, the set $(S_2 \cup S_3 \cup S_4)$ has f_2 as its most suspicious form, and f_3, f_5 as its next two most suspicious forms. Moreover, most of the instances in S_1, S_4 and S_5 fail because of a single form namely, f_1, f_2 and f_4 respectively.

Suspicion tree algorithm vs. ID3 algorithm. There are two main differences between (Quinlan, 1986)'s ID3 decision tree learning algorithm and the suspicion tree construction algorithm.

First, the suspicion tree construction algorithm allows for stronger pruning and termination conditions – in this way, only the most relevant suspicious forms are displayed thereby facilitating error analysis.

Second, attribute selection is determined not by the information gain (IG) but by the suspicion score

(SS) metrics. Recall that the information gain[1] metrics aims to identify the attributes which lead to more homogeneous classes. In the present case, the classes are either PASS (the inputs for which generation succeeds) or FAIL (the inputs for which generation fails). Thus the IG metrics will indifferently seek to identify attributes which predominantly associate either with a FAIL or with a PASS. There is no preference for either the FAIL or the PASS class. For error mining however, what is needed is to identify attributes which predominantly associate with the FAIL class. That is, we need a metric which permits identifying attributes which leads to classes that are homogeneous in terms of FAIL instances rather than homogeneous in terms of either FAIL or PASS instances. The example shown in Figure 2 illustrates the difference.

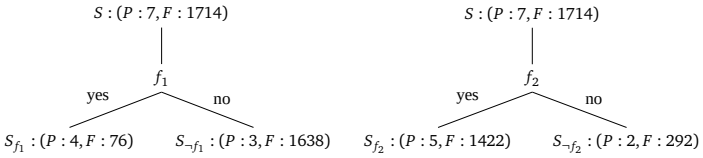


Figure 2: Attribute Selection using Information Gain (Left) and Suspicion Score (Right). While IG selects f_1, an attribute which associate 76 times with generation failure, SS selects f_2, an attribute which associates 1422 times with generation failure.

In this example, we apply the IG and the SS metrics to the same input data, a set containing 7 inputs associated with generation success and 1714 inputs associated with generation failure. While SS selects f_2, an attribute which associates 1422 times with generation failure, IG selects f_1, an attribute which associate only 76 times with generation failure. In this case, the information gain metrics incorrectly select f_1 because its absence from the input, yields a numerically very homogeneous class in terms of generation failure. Indeed, the information gain of f_1 is close to but higher than the information gain of f_2 because the resultant subsets S_{f_i} and $S_{\neg f_i}$ are treated equally while computing the information gain.

2.2 Complexity Analysis and Extensions

Let n and m be the size of the dataset $\mathbb{D}$ and of the form set $\mathbb{F}$ respectively. Then, in the worst case, the suspicion tree will be of depth $O(log\ n)$ with $O(n)$ nodes. Each node chooses a suspicious form out of $O(m)$ forms. Thus the worst computational complexity for building the suspicion tree is $O(m\ n\ log\ n)$. But on average, the algorithm described in Section 2.1 performs much faster than this. The worst case happens when the forms used to classify the corpus into PASS and FAIL are not very discriminant i.e., when all suspicious forms are equally probable.

The algorithm for building the suspicion tree is directly proportional to the size of the set $\mathbb{F}$. Since $|\mathbb{F}|$ can be very large, this can be problematic. Indeed, in the error mining on sentences for parsing systems proposed in (Sagot and de la Clergerie, 2006), the authors indicate that, in order to remain computationally tractable, the approach must be restricted to n-grams of smaller size (unigrams and bigrams). The problem is accrued of course when considering tree shaped suspicious forms (Gardent and Narayan, 2012). To abate this issue we propose two extensions to prune the suspicion tree.

[1]Information gain (IG) is defined as $IG = H(S) - ((|S_{f_i}|/|S|) * H(S_{f_i}) + (|S_{\neg f_i}|/|S|) * H(S_{\neg f_i}))$ where $H(X)$ is the entropy of set X. (Quinlan, 1986)

First, we reduce the form space $\mathbb{F}$. Following a suggestion from (de Kok et al., 2009), instead of considering all possible forms, we only consider those forms whose frequency is above a given threshold. We also account for *suspicion sharing* (i.e., the sharing of suspicion by several overlapping forms) by only considering a larger suspicious form if its suspicion rate is larger than the suspicion rate of all smaller forms it contains. These two extensions reduce the form space significantly and allow for an efficient building of the suspicion tree. To enumerate with these extensions, we use a complete and efficient algorithm described in (Gardent and Narayan, 2012).

Second, we constrain the depth of the suspicion tree. Because error mining is a cyclic process, building the complete suspicion tree is usually unnecessary. The quantity of information processed in each cycle depends on the user but in general, the linguist will focus on the top suspicious forms, use these to improve the generator and rerun error mining on the improved results. The faster the error mining step is, the better this is for this development cycle. Considering this, we added an extra constraint over the depth of the suspicion tree. This depth limit permits pruning the suspicion tree and a faster improvement cycle. In our experiments, we used a depth limit of 10.

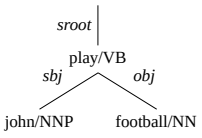
With these extensions, the enumeration process of suspicious forms takes 10-15 minutes for a dataset consisting of 123,523 trees. Building a suspicion tree for the same dataset takes about one minute.

3 Applying the Suspicion Tree Algorithm to Generation Data

We now report on an experiment we did using the suspicion tree algorithm described in the preceding section to detect and classify the most likely causes of failure when running a surface realiser on the Surface Realisation (SR) Task data. We first describe the experimental setup (Section 3.1). We then illustrate by means of examples, how suspicion trees better support error analysis than ranked lists proposed by previous error mining approaches (Section 3.2). Finally (Section 3.3), we discuss the improvements in surface realisation obtained by fixing the errors identified using error mining.

3.1 Experimental Setup

Dataset The dataset to be error mined is the set of shallow dependency trees (Figure 3) provided by the SR Task organisers and used as input for surface realisation. These trees are unordered syntactic dependency trees whose edges are labelled with dependency relations and whose nodes are labelled with lemmas and part of speech (POS) categories. In this paper, we represent these trees by an n-tuple with the root node of the tree as its first element followed by $(n-1)$ elements representing its dependent subtrees. Dependency relations are lowered to the corresponding daughter node.



word (play, (john), (football))
POS (VB, (NNP), (NN))
dep (*sroot*, (*sbj*), (*obj*))
word/**POS** (play/VB, (john/NNP), (football/NN))
***dep*-POS** (*sroot*-VB, (*sbj*-NNP), (*obj*-NN))

Figure 3: An example shallow dependency tree from the SR Task and the corresponding representations used in this paper. Our error mining algorithm considers as suspicious forms, subtrees labelled with arbitrary conjunctions of lemmas (word), part-of-speech tags (POS), dependency relations (*dep*).

To facilitate error mining, we proceed in an incremental way and examine dependency trees in the SR data that correspond to NP and Sentences of increasing size. Here we report on error mining performed on NP-type dependency trees of sizes 4 (NP-4), 6 (NP-6) and all (NP-ALL), and S-type dependency trees of sizes 6 (S-6), 8 (S-8) and all (S-ALL) (where the size refer to the number of nodes/lemmas in the tree). The data used for generation is preprocessed whereby named entities and hyphenated words are grouped into a single word and punctuation is removed so as to first focus on lexical and grammatical issues.

Attributes The attributes used to partition the SR data are suspicious trees i.e., subtrees of the SR dependency trees whose frequency is above a given threshold. Following (Gardent and Narayan, 2012), we allow for various views on errors by mining for forms labelled with lemmas only (word); with parts of speech (POS); with dependency relations (*dep*); with lemmas and parts of speech (word/POS); and with dependency relations and parts of speech (*dep*-POS) (cf. Figure 3).

Generation System The system to be tested is the symbolic Surface Realiser described in (Narayan and Gardent, 2012). We ran this surface realiser on the SR input data and separately stored the input dependency trees for which generation succeeded (PASS) and the input dependency trees for which generation failed (FAIL). We then removed from the failed data, those cases where generation failed either because a word was missing in the lexicon or because a grammar rule was missing but required by the lexicon and the input data. These cases can easily be detected using the generation system and thus do not need to be handled by error mining.

Error Mining We iterate several times between error mining and performance improvement and applied the suspicion tree algorithm to both the NP and the S data[2].

3.2 Error Analysis using Suspicion Trees

We now show by means of examples how the suspicion tree algorithm helps support error analysis. We start by showing how the overall structure of the suspicion tree (right frontier and subtrees) improves upon ranked lists when analysing the data. We then go on to show how subtrees in the suspicion tree permit differentiating between forms that are suspicious in all contexts and require a single correction; forms that are suspicious in all contexts but require several corrections; and forms that are suspicious in some but not all contexts.

3.2.1 Suspicion Trees vs. Ranked Lists

Figure 4 shows a top fragment of the suspicion tree obtained by error mining on NP-4. The node labels in this tree describe suspicious forms with part-of-speech information only.

In that tree, the right frontier indicates that the main distinct suspicious forms are, in that order:

1. Possessive NPs (POSS is the part of speech tag assigned to possessive *'s*[3])
The suspicious form (POSS) points to a mismatch between the representation of genitive NPs (e.g., *Oakland's thief*) in the SR Task data and in the grammar. While our generator expects the representation of '*Oakland's thief*' to be (thief/NN, ('s/POSS, (oakland/NNP))), the structure provided by the SR Task is (thief/NN, (oakland/NNP, ('s/POSS))). Hence whenever a possessive appears in the input data, generation fails. This is in line with (Rajkumar et al., 2011)'s finding that the logical

[2]Iteration stops either when the results are perfect (perfect coverage and perfect BLEU score) or when the trees fail to be discriminative enough (low number of FAIL instances associated with the suspicion tree leaves). So far, the latter situation did not occur and we are still using the suspicion tree to identify the main sources of errors for the remaining error cases.

[3]In fact, the part of speech tag assigned to possessive *'s* in the SR data is POS not POSS. We renamed it to avoid confusion with POS as an abbreviation for part-of-speech.

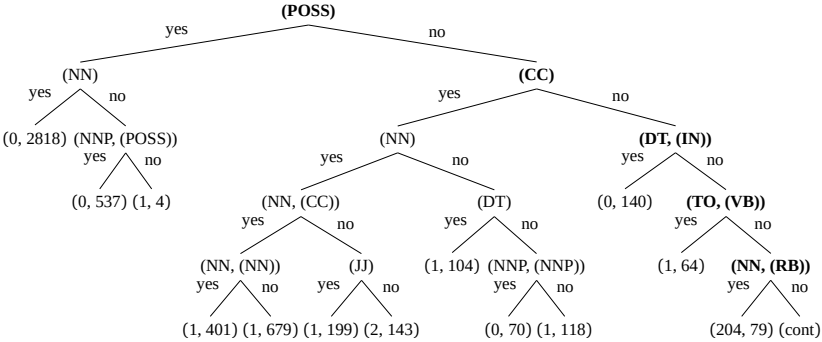


Figure 4: Suspicion Tree for Generation from the NP-4 data. Nodes are labelled with dependency subtrees with POS information. The leaves (p, f) represent the cluster with PASS (p) and FAIL (f) instances.

forms expected by their system for possessives differed from the shared task inputs. To correct these cases, we implemented a rewrite rule that converts the SR representation of possessive NPs to conform with the format expected by our realiser.

2. NPs with coordination (CC with daughter node NN)

The second top right frontier node unveils a bug (conflicting feature values) in the grammar trees associated with NP conjunction (e.g., *Europe and the U.S.*) which made all sentences containing an NP conjunction fail.

3. Determiners (DT) dominating a preposition (IN)

As we shall see below, this points to a discrepancy between the SR part of speech tag assigned to words like '*some*' in '*some of the audience*' and the part of speech tag expected by our generator. While in the SR data, such occurrences are labelled as determiners (DT), our generator expects these to be tagged as pronouns (PRP).

4. The complementizer *to* (TO) dominating a verb (VB)

As discussed below, this points to cases where the infinitival verb is a noun modifier and the input structure provided by the SR Task differs from that expected by our realiser.

5. Nouns (NN) dominating an adverb (RB)

This points to a discrepancy between the SR part of speech tag assigned to words like 'alone' in '*real estate alone*' and the part of speech tag expected by our generator. While in the SR data, such occurrences are labelled as adverbs (RB), our generator expects these to be tagged as adjectives (JJ).

In addition, for each node n on the right frontier, the subtree dominated by the yes-branch of n gives further information about the more specific forms that are subcases of the suspicious form labelling n.

The suspicion tree gives a structured view of how the various suspicious forms relate. In comparison, the ranked lists produced by previous work are flat structures which may fail to adequately display these information. For instance, applying (Gardent and Narayan, 2012)'s error mining algorithm to the data used to produce the tree shown in Figure 4 yields the list shown in Figure 5. Contrary to the suspicion tree shown in Figure 4, this list fails to highlight the main culprits and

1. **(POSS)**
2. (NNP, (POSS))
3. **(CC)**
4. (NN, (POSS))
5. (NN, (NNP, (POSS)))
6. (NN, (NN, (POSS)))
7. (NN, (CC))
8. (NNP, (NNP), (POSS))
9. (NN,(NNP,(NNP),(POSS)))
10. (NN, (NNP, (NNP)))
11. (CC, (JJ))
12. (JJ, (CC))
13. (NNP, (NNP, (POSS)))
14. (NN, (NN), (POSS))
15. **(DT, (IN))**
16. (JJ, (CC, (JJ)))
17. (NN, (CC), (NN))
18. (NN, (NNP), (POSS))
19. **(TO, (VB))**
20. (NN,(NNP,(POSS)),(NNP))
21. (NN, (NNP))
22. (NNP, (NNP))
23. (NN, (NN))
24. (NNP)
25. (NN)
26. (NN, (NNP), (NNP))
27. (VB)
28. **(NN, (RB))**
29. (PRP)
30. (DT)

Figure 5: Ranked list of suspicious forms for Generation from the NP-4 data.

the relations between the various suspicious forms. Thus the 5 main distinct suspects identified by the right frontier of the suspicion tree appears as 1st, 3rd, 15th, 19th and 28th in the ranked list. Furthermore, while subcases of the main suspects are grouped in the yes-branch of these suspects in the suspicion tree, in the ranked list, they appear freely interspersed throughout. For example, suspicious forms involving the two main suspects in the suspicion tree approach (POSS and CC part-of-speech tags) are scattered throughout the list rather than grouped under the first two right frontier nodes respectively.

Also the stronger pruning conditions used for building the suspicion tree restrict the branch exploration as soon as homogeneous clusters are achieved. For a given dataset, it only explores those suspicious forms which are good enough to identify the problems causing the failure in that dataset. For example the data containing the suspicious form (POSS) is explored with 3 suspicious forms (POSS), (NN) and (NNP, (POSS)) in the suspicion tree shown in Figure 4 whereas in the ranked list shown in Figure 5, there are 11 suspicious forms associated with (POSS). In general, the number of forms displayed by the suspicion tree algorithm is much less than that of the ranked list ones thereby giving a clearer picture of the main culprits and of their subcases at each stage in the error mining/grammar debugging cycle.

3.2.2 Reading error types off the tree structure

For each node n labelled with suspicious form f_n in a suspicion tree, the subtree dominated by n gives detailed information about the possible contexts/causes for f_n. In what follows, we show how the suspicion tree algorithm permits distinguishing between three main types of suspicious forms namely, forms that are suspicious in all contexts and require a single correction; forms that are suspicious in all contexts but require several corrections; and forms that are suspicious in some but not all contexts.

Forms that are suspicious independently of context and require a single correction. When a suspicious form always leads to failure, the node labelled with that suspicious form has no subtree thereby indicating that all configurations including that suspicious form lead to generation failure independent of context.

Such cases are illustrated in Figure 6 which show two views (one with part of speech tag only, the other with words and parts of speech tags) of the suspicion tree obtained after addressing the two main causes of errors identified in the previous section. That is, a rewrite rule was applied to convert the SR representation of possessive NPs to conform with the format expected by our

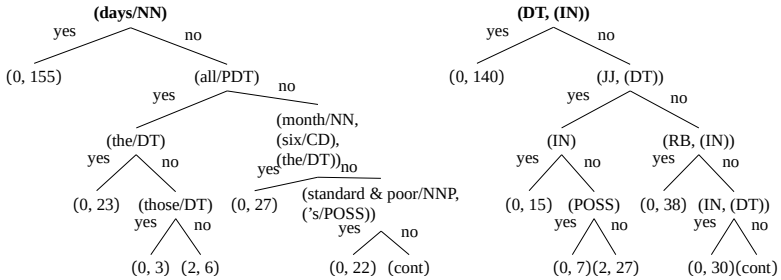


Figure 6: Suspicion Tree for (word/POS) (left) and (POS) (right) for Generation from the NP-4 data (after fixing genitive and coordination cases).

realiser ((POSS) suspicious form); and the grammar was corrected to generate for NP coordination ((CC) suspicious form).

In each of these two trees, the yes-branch of the root node has no subtree indicating that all input trees containing either the word form '*days*' with part of speech tag NN (days/NN); or a determiner dominating a preposition ((DT,(IN))) lead to generation failure.

The root node (days/NN) of the suspicion tree shown on the left of Figure 6 points to a problem in the lexicon. Although days/NN is present in the lexicon, it is not associated with the correct TAG family. We modified the entries corresponding to (days/NN) in the lexicon to solve this problem.

As mentioned above, the root node (DT, (IN)) of the suspicion tree shown on the right in Figure 6 points to a part-of-speech tagging problem in the SR Data. Words like '*some*' or '*all*' followed by a preposition (e.g., *some of the audience*, *all of fiscal 1990*, *those in other industries*) are assigned the determiner part of speech tag (DT) where our generator expects a pronoun (PRP) part-of-speech tag. To correct these cases, we implemented a rewrite rule that maps DT to PRP in the above specified context.

As these two examples illustrate, using different views (forms labelled with part of speech tags only *vs.* forms labelled with words and parts of speech) on the same data may help identifying problems at different levels. Both suspicion trees in Figure 6 are built for generation from same NP-4 dataset. The leftmost tree (suspicious forms labelled with both lemma and part of speech information) helps identifying problems in the lexicon whereas the rightmost tree (suspicious forms labelled with parts of speech only) points to problems in the input data.

Forms that are suspicious independent of context but require several corrections. It may be that a given form is almost always suspicious but that it occurs in different linguistic contexts requiring different corrections. In such cases, the suspicion tree will highlight these contexts. The root of the tree shown in Figure 7 is a case in point. The suspicious form (*im*-VB) describes subtrees whose head is related to a verb by the *im* dependency relation i.e., *infinitival verbs*. The subtrees (of the yes-branch) of that root further describe several syntactic configurations which are suspect and contain an infinitival verb. The node labelled with (*oprd*-TO) points to subcases where the infinitival verb is the complement of a control (1a[i]) or a raising verb (1a[ii]). The node labelled with (*im*-VB, (*prd*-JJ)) points to a subcase of that case namely that of an infinitival verb which is

the complement of a control or a raising verb and subcategories for an adjectival complement e.g., (1b). Finally, the node labelled with (*nmod*-TO, (*im*-VB)) points to cases where the infinitival verb is a noun modifier (1c).

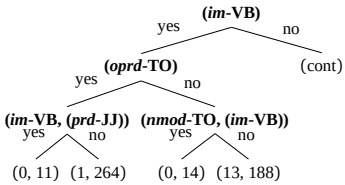


Figure 7: Suspicion Tree (*dep*-POS) for Generation from the S-6 data.

(1) a. (*oprd*-TO)

i *He will try to assuage the fears about finances.*
(try/VB, (*oprd*-to/TO, (*im*-assuage/VB))

ii *Many of the morning session winners turned out to be losers.*
(turn/VB, (*oprd*-to/TO, (*im*-be/VB, (*prd*-loser/NN)))

b. (*im*-VB, (*prd*-JJ))
Amex expects to be fully operational by tomorrow.
(expect/VB, (*oprd*-to/TO, (*im*-be/VB, (*prd*-operational/JJ)))

c. (*nmod*-TO, (*im*-VB))
The ability to trade without too much difficulty has steadily deteriorated.
(ability/NN, (*nmod*-to/TO, (*im*-trade/VB))

(2) a. (IN, (CD))
the end of 1991
(end/NN, (the/DT), (of/IN, (1991/CD)))

b. (CD, (IN))
one of the authors
(one/CD, (of/IN, (author/NN, (the/DT))))

c. (CD, (CD))
Nov. 1 , 1997
(1/CD, (1997/CD), (Nov./NNP), (,/SYS))

d. (CD, (DT))
A seasonally adjusted 332.000
(332.000/CD, (a/DT), (adjusted/JJ, (seasonally/RB)))

e. (CD, (RB))
1987 and early 1988
(1987/CD, (and/CC, (1988/CD, (early/RB))))

Although all of these cases are due to a mismatch between the SR Task dependency trees and the input expected by our realiser, they point to different input configurations requiring different modifications (rewritings) to ensure compatibility with the realiser. The structured information given by the suspicion tree provides a clear description of the main tree configurations that need to be rewritten to correct generation failures induced by infinitival verbs. We used these information to implement the rewrite rules required to resolve the identified mismatches.

Forms that are suspicious in some but not all contexts. The suspicion tree can also highlight forms that are suspicious in some but not all contexts. For instance, the right frontier of the suspicion tree in Figure 8 shows that the CD (cardinals) part of speech occurs in several suspicious forms namely, (IN, (CD)) (a preposition dominating a cardinal), (CD, (IN)) (a cardinal dominating a preposition), (CD, (CD)) (a cardinal dominating a cardinal), (CD, (DT)) (a cardinal dominating a determiner) and (CD, (RB)) (a cardinal dominating an adverb). Examples for these configurations and their subcases are given in (2).

Noticeably, the suspicious form (CD) does not appear in the suspicion tree. In other words, the tree highlights the fact that cardinals lead to generation failure in the contexts shown but not in all contexts. Indeed, in this case, all suspicious forms points to a single cause of failure namely, a mis-

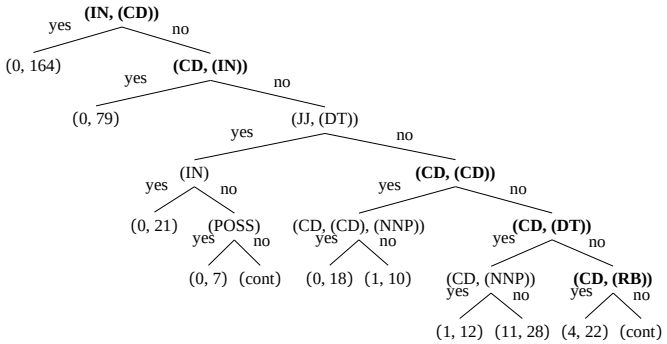


Figure 8: Suspicion Tree (POS) for Generation from the NP-6 data (after fixing genitive, coordination and determiner cases).

match between grammar and lexicon. In the TAG grammar used, the constructions illustrated in (2) all expect cardinals to be categorised as nouns. In the lexicon however, cardinals are categorised as determiners. We modified the lexicon to categorise cardinals as determiners, nouns and adjectives and rerun the generator on the input. In the newly built suspicion trees, cardinals no longer induce high generation failure rates. The fact that cardinals are not always associated with failure can be traced back to the fact that they are often used as determiners and that for this context, the lexicon contains the appropriate information.

3.3 Using Error Mining to Improve Generation Results

We now briefly report on how the suspicion tree algorithm can help improve a generation system by showing the impact of corrections on undergeneration.

Generating NPs. Table 1 summarises a run with 6 iterations between error mining and error correction on the NP data. The corrections involve rewriting the SR data to the format expected by our realiser, grammar corrections and lexicon enrichment. Each time a correction is applied, the suspicion tree is recomputed thereby highlighting the next most likely sources of errors. G(Coord) indicates a fix in the grammar for coordination (discussed in Section 3.2.1). R(Gen) involves rewriting dependency trees for genitive NPs (e.g., *Oakland 's thief*) (Section 3.2.1) and R(Dt) rewriting dependency trees with determiners to map its part-of speech from determiner (DT) to pronoun (PRP) (Section 3.2.2) and to noun (NN) (nominal positions, e.g., *That's good*). L(days) involves updating the lexicon with correct days/NN to TAG families mapping (Section 3.2.2). R(Adv) involves rewriting dependency trees with adverbs to map its part-of speech from adverb (RB) to adjective (JJ) (e.g., *real estate alone*) (Section 3.2.1).

As the table shows, error mining permits decreasing undergeneration by 22.6, 25.6 and 8.7 points for NPs of size 4, 6 and ALL respectively. This suggests that simple NPs can be generated but that bigger NPs still cause undergeneration (8.2% and 13% of the cases respectively for NP-6 and NP-ALL) presumably because of more complex modifiers such as relative clauses, PPs and multiple determiners. Since in particular, relative clauses also appear in sentences, we proceeded to error mine the sentence data so as to provide more data for the error mining algorithm and therefore get a more global picture of the most important causes of failure.

	Input Data	Init Fail	G(Coord)	R(Gen)	R(Dt)	L(days)	R(Adv)
NP-4	23468	5939 (*25.3*)	4246 (*18.1*)	999 (*4.3*)	833 (*3.6*)	678 (*2.9*)	649 (*2.7*)
NP-6	10520	3560 (*33.8*)	2166 (*20.6*)	956 (*9.1*)	881 (*8.4*)	876 (*8.3*)	865 (*8.2*)
NP-ALL	123523	26769 (*21.7*)	21525	16702	16263	16094	16028 (*13*)

Table 1: Diminishing the number of errors using information from error mining on NP data. The first column indicates the type of NP chunks to be processed, the second (*Input Data*) the number of NP chunks to be processed, the third (*Init Fail*) the number of input on which generation initially fails and the last 5 ones the decrease in errors (the number of failed cases with the *percentage failure*) after fixing error cases identified by the suspicion tree. R(X) indicates that the correction is obtained by rewriting the input for phenomena X, G(X) indicates corrections in the grammar and L(X) indicates corrections in the lexicon.

Generating Sentences. Tables 2 show the impact of corrections on generation for sentences. For this data, we start with all the improvements made during error mining on the NP data. Table 2 represents this step as *NP-Final* summarizing generation results after all improvements from Table 1. During error mining on the S data, *infinitival verbs* (discussed in Section 3.2.2) and *auxiliary verbs* appear as prominent mismatches between the SR dependency trees and the input expected by our generator. R(Inf) in Table 2 involves 3 different rewriting rules corresponding to dependency relations *im*, *oprd* and *prd* for rewriting dependency trees with infinitival verbs. R(Aux) indicates rewriting for dependency trees with Verb/Auxiliary nuclei (e.g., *the whole process might be reversed*).

	Input Data	NP-Final	R(Inf)	R(Aux)
S-6	3877	1707 (*44.0*)	753 (*19.4*)	398 (*10.3*)
S-8	3583	1749 (*48.8*)	936 (*26.1*)	576 (*16.1*)
S-ALL	26725	19280 (*72.1*)	17862 (*66.8*)	16445 (*61.5*)

Table 2: Diminishing the number of errors using information from error mining on S data. The first column indicates the type of sentences to be processed, the second (*Input Data*) the number of sentences to be processed, the third (*NP-Final*) the number of input (processed with all improvements from Table 1) on which generation fails and, the fourth and the fifth error rates after rewriting dependency trees for infinitival cases and auxiliary verb cases respectively.

Finally, Table 3 summarises results from Table 1 and Table 2 adding an extra final improvement step (*Final*) consisting of minor grammar improvement (trees for pre-determiner PDT added, e.g., *all these millions*), lexicon enrichment (mapping to TAG families corrected) and rewriting rule (mapping part-of-speech from conjunction CC to determiner DT, e.g., *neither/CC the Bush administration nor arms-control experts*). The "Final" row in this Table shows the impact of S error reduction on NP error reduction. As can be seen reducing S-errors positively impact NP errors throughout.

In total we defined 11 rewrite rules (Gen-1, Dt-4, Adv-1, Inf-3, Aux-1 and Final-1), made 2 grammar corrections and performed a few lexicon updates.

Coverage and accuracy. As the tables show, the corrections carried out after a few cycle of error mining and error correction helps achieve a large improvement in coverage for smaller dependency trees; we notice a large drop of 23.2 points (from 25.3% to 2.1%) in error rates for NP-4, 28.3 points for NP-6, 34.5 points for S-4 and 33.6 points for S-6. For larger dependency trees however, improvement is more limited and other error cases becomes visible. Thus, the failure rate is reduced

by 10.4 points for NP-ALL (NPs from minimum size 1 to maximum size 91 with the average size 4); and by 10.9 points for S-ALL (sentences from minimum size 1 to maximum size 134 with the average size 22). The suspicion tree built after the *Final* step shows that coordination cases appear as most suspicious forms. Although the corrections made for coordination in the grammar G(Coord) permit generating simple coordinations (e.g., *John and Mary likes beans.*, *We played on the roof and in the garden.*, *I cooked beans and she ate it.*), the grammar still fails to generate for more complex coordination phenomenon (e.g., verb coordination *I cooked and ate beans.*, gapping phenomenon *John eat fish and Harry chips.*, *I liked beans that Harry cooked and which Mary ate.*) (Sarkar and Joshi, 1996). Other top suspicious forms are multiword expressions (e.g., *at least*, *so far*) and foreign words (part-of-speech FW) (e.g., *the naczelnik*, *perestroika*, *product de jour*).

	NP-4	NP-6	NP-ALL	S-6	S-8	S-ALL
Input Data	23468	10520	123523	3877	3583	26725
Init Fail	5939 (*25.3*)	3560 (*33.8*)	26769 (*21.7*)	-	-	-
NP-Final	649 (*2.7*)	865 (*8.2*)	16028 (*13.0*)	1707 (*44.0*)	1749 (*48.8*)	19280 (*72.1*)
S-Final	-	-	-	398 (*10.3*)	576 (*16.1*)	16445 (*61.5*)
Final	503 (*2.1*)	584 (*5.5*)	13967 (*11.3*)	371 (*9.5*)	545 (*15.2*)	16374 (*61.2*)

Table 3: Overall impact of error mining on generation from different types of dependency trees. The first row indicates the type of dependency data to be processed and the second (*Input Data*) the number of data to be processed. The rows named (*Init Fail*), (*NP-Final*), (*S-Final*) and (*Final*) are initial error rates, errors after applying improvements from Table 1, errors after applying improvements from Table 2 and errors after final improvements respectively.

To assess the precision of the surface realiser after error mining, we computed the BLEU score for the covered sentence data and obtained a score of 0.835 for S-6, 0.80 for S-8 and 0.675 for S-ALL[4].

4 Conclusion

We introduced an error mining algorithm that takes inspiration from (Quinlan, 1986)'s ID3 algorithm to structure the output of error mining in a way that supports a linguistically meaningful error analysis. We demonstrated its workings by applying it to the analysis of undergeneration in a grammar based surface realisation algorithm. And we show that it permits quickly identifying the main sources of errors while providing a detailed description of the various subcases of these sources if any.

The approach is generic in that permits mining trees and strings for suspicious forms of arbitrary size and arbitrary conjunctions of labelling. It could be used for instance to detect and structure the n-grams that frequently induce parsing errors; or to identify subtrees that frequently occur in agrammatical output produced by a generator.

We are currently working on further improving the generator using the suspicion tree algorithm. In future work, we plan to use our error mining algorithm to detect the most likely sources of over-generation based on the output of a surface realiser; and to investigate whether the approach can be useful in automatically detecting treebank and parse errors (Boyd et al., 2008; Dickinson and Smith, 2011).

Acknowledgments The research presented in this paper was partially supported by the European Fund for Regional Development within the framework of the INTERREG IV A Allegro Project.

[4]The BLEU score before error mining and correction is not reported here since it has low coverage due to the mismatches between the structures provided by the SR task and those expected by the realiser.

References

Belz, A., White, M., Espinosa, D., Kow, E., Hogan, D., and Stent, A. (2011). The first surface realisation shared task: Overview and evaluation results. In *Proceedings of the 13th European Workshop on Natural Language Generation (ENLG)*, Nancy, France.

Boyd, A., Dickinson, M., and Meurers, D. (2008). On detecting errors in dependency treebanks. *Research on Language and Computation*, 6(2):113–137.

Chi, Y., Yang, Y., and Muntz, R. R. (2004). Hybridtreeminer: An efficient algorithm for mining frequent rooted trees and free trees using canonical form. In *Proceedings of the 16th International Conference on and Statistical Database Management (SSDBM)*, pages 11–20, Santorini Island, Greece. IEEE Computer Society.

de Kok, D., Ma, J., and van Noord, G. (2009). A generalized method for iterative error mining in parsing results. In *Proceedings of the 2009 Workshop on Grammar Engineering Across Frameworks (GEAF 2009)*, pages 71–79, Suntec, Singapore. Association for Computational Linguistics.

Dickinson, M. and Smith, A. (2011). Detecting dependency parse errors with minimal resources. In *Proceedings of the 12th International Conference on Parsing Technologies (IWPT 2011)*, Dublin, Ireland.

Gardent, C. and Narayan, S. (2012). Error mining on dependency trees. In *Proceedings of the 50th Meeting of the Association for Computational Linguistics (ACL)*, pages 592–600, Jeju, Korea.

Narayan, S. and Gardent, C. (2012). Structure-driven lexicalist generation. In *Proceedings of the 24th International Conference on Computational Linguistics (COLING)*, Mumbai, India.

Quinlan, J. R. (1986). Induction of decision trees. *Machine Learning*, pages 81–106.

Rajkumar, R., Espinosa, D., and White, M. (2011). The osu system for surface realization at generation challenges 2011. In *Proceedings of the 13th European Workshop on Natural Language Generation (ENLG)*, pages 236–238, Nancy, France.

Sagot, B. and de la Clergerie, E. (2006). Error mining in parsing results. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 329–336, Sydney, Australia.

Sarkar, A. and Joshi, A. K. (1996). Coordination in tree adjoining grammars: Formalization and implementation. In *Proceedings of the 16th International Conference on Computational Linguistics (COLING)*, pages 610–615.

van Noord, G. (2004). Error mining for wide-coverage grammar engineering. In *Proceedings of the 42nd Meeting of the Association for Computational Linguistics (ACL)*, pages 446–453, Barcelona, Spain.

Structure-Driven Lexicalist Generation

Shashi Narayan[1] Claire Gardent[2]
(1) Université de Lorraine/LORIA, Nancy, France
(2) CNRS/LORIA, Nancy, France
shashi.narayan@loria.fr, claire.gardent@loria.fr

Abstract
We present a novel algorithm for surface realisation with lexicalist grammars. In this algorithm, the structure of the input is used both top-down to constrain the selection of applicable rules and bottom-up to filter the initial search space associated with local input trees. In addition, parallelism is used to recursively pursue the realisation of each daughter node in the input tree. We evaluate the algorithm on the input data provided by the Generation Challenge Surface Realisation Task and show that it drastically reduce processing time when compared with a simpler, top-down driven, lexicalist approach.

Title and Abstract in Hindi

संरचना-प्रेरित शब्द-सज्जित उत्पादन

हम शब्द-सज्जित व्याकरण की मदद से वाक्यों के सतह संपादन के लिये एक नयी तकनीक प्रस्तुत कर रहे हैं। इस तकनीक में आगत संरचना का उपयोग दोनों "ऊपर से नीचे" उपयुक्त नियमों के चयन मे तथा "नीचे से ऊपर" आगत संरचना में मौजुद स्थानीय वृक्षों से जुड़े प्रांरभिक परीक्षण क्षेत्र को फिल्टर करने में किया गया है। इसके साथ, समांतरवाद का उपयोग आगत वृक्षों के शाखों की सतह संपादन हेतु कुशल तरीके से किया गया है। हम अपने तकनीक का मुल्यांकन "उत्पादन प्रतियोगिता: सतह संपादन" के आंकड़ों पर करते हैं तथा हम यह दिखातें हैं कि प्रस्तुत तरीका दूसरे साधारण "ऊपर से नीचे" प्रेरित शब्द-सज्जित तरीकों के मुकाबले उत्पादन समय को अत्यधिक घटा देता है।

Keywords: Generation, Tree Adjoining Grammar, Surface Realization.

Keywords in Hindi: उत्पादन, वृक्ष-सट-व्याकरण, सतह संपादन.

Proceedings of COLING 2012: Technical Papers, pages 2027–2042,
COLING 2012, Mumbai, December 2012.

1 Introduction

Depending on the type of semantic representation encoded by the grammar, two main types of algorithms have been proposed for generating sentences with bi-directional, unification-based grammars such as CCG (Combinatory Categorial Grammar, (Espinosa et al., 2010)), HPSG (Head-Driven Phrase Structure Grammar, (Carroll et al., 1999)) and TAG (Tree Adjoining Grammar, (Gardent and Kow, 2005)).

For recursive semantic representations such as first-order logic formulae, head-driven algorithms (Shieber et al., 1990) have been argued to be best because they restrict the combinatorics inherent to bottom-up search; they avoid non termination by using lexical items to guide the search ; and they allow for semantically nonmonotonic grammars (i.e., grammars where the semantics of a rule at the left hand side need not be subsumed by the semantics of the rule at the right hand side). One main issue with this approach however is the so-called logical form equivalence problem (Shieber, 1993). A logic formula may have several logically equivalent but syntactically distinct formulae. For instance $p \wedge q$ is logically equivalent to $q \wedge p$. In general though, a grammar will associate with natural language expressions only one of these logically equivalent formula. Hence a generator will be able to produce the natural language expression E only when given the formula ϕ associated by the grammar with E. For all other formulae logically equivalent to ϕ, it will fail. Since, the problem of computing logical equivalence for first order logic is undecidable, the problem is quite deep.

For flat semantic representations such as MRSs (Minimal Recursion Semantics, (Copestake et al., 2001)) on the other hand, lexicalist approaches (Espinosa et al., 2010; Carroll and Oepen, 2005; Gardent and Kow, 2005) have extensively been used because (i) they impose few constraints on the grammar thereby making it easier to maintain bi-directional grammars that can be used both for parsing and for generation; and (ii) the approach eschews the logical form equivalence problem – Since the semantic representations are unstructured, there is no requirement on the generator to mirror a semantic structure. One known drawback of lexicalist approaches however is that they generally lack efficiency. Indeed, previous work has shown that the high combinatorics of lexicalist approaches stem from (i) strong lexical ambiguity (each input element is usually associated with a large number of grammatical structures thereby inducing a very large initial search space); (ii) the lack of order information in the input (as opposed to parsing where the order of words in the input string restricts the number of combinations to be explored); and (iii) intersective modifiers (given n modifiers applying to the same constituent, there are $n!$ ways to combine these together).

In this paper, we present an algorithm for surface realisation that combines techniques and ideas from the head-driven and the lexicalist approach. On the one hand, rule selection is guided, as in the lexicalist approach, by the elementary units present in the input rather than by its structure – In this way, the logical form equivalence issue is avoided. On the other hand, the structure of the input is used to provide top-down guidance for the search and thereby restrict the combinatorics.

To further improve efficiency, the algorithm integrates three additional optimisation techniques. From the lexicalist approach, it adapts two techniques designed to prune the search space, namely a so-called polarity filter on local input trees (Bonfante et al., 2004); and the use of a language model to prune competing intermediate substructures. In addition, the algorithm is parallelised to explore the possible completions of the top-down predictions simultaneously rather than sequentially.

The algorithm was implemented using a Feature-Based Lexicalised Tree Adjoining Grammar for English and tested on the Generation Challenge Surface Realisation task data (Belz et al., 2011). We compare our algorithm with a baseline lexicalist approach which processes the input tree top

down. The results show that the algorithm we propose drastically improves on the baseline, reducing generation time for sentences longer than 6 words w.r.t. this baseline.

This paper is structured as follows. Section 2 situates our approach with respect to related work. Section 3 introduces the input data provided by the Generation Challenge Surface Realisation task and used for the evaluation. Section 4 introduces the tree adjoining grammar used by the algorithm. Section 5 presents the surface realisation algorithm we developed. Section 6 describes the evaluation setup and the results obtained. Section 7 concludes with pointers for further research.

2 Related Work

Most of the recent proposals on optimising surface realisation with unification grammars focuses on lexicalist approaches, they place minimal requirements on the grammar and eschew the logical form equivalence problem. We now review the optimisation techniques used in these approaches. We also briefly review recent work on statistical approaches to surface realisation.

For HPSG, (Carroll and Oepen, 2005) present a bottom-up, lexicalist, surface realiser which uses a chart based strategy, subsumption-based local ambiguity factoring and a procedure to selectively unpack the generation forest according to a probability distribution given by a conditional, discriminative model. The algorithm is evaluated on the *hike* treebank, a collection of 330 sentences of instructional text taken from Norwegian tourism brochures with an average length of 12.8 words. Practical generation times average below or around one second for outputs of 15 words.

For TAG, (Gardent and Kow, 2007) propose a three step surface realisation algorithm for FB-LTAG (Feature-Based Lexicalised Tree-Adjoining Grammar) where first, a so-called polarity filter is used to prune the initial search space second, substitution is applied to combine trees together and third, adjunction is applied.

In essence, polarity filtering filters out combinations of FB-LTAG elementary trees which cover the input semantics but cannot yield a valid parse tree either because a syntactic requirement cannot be satisfied or because a syntactic resource cannot be used. In this way, the exponential impact of lexical ambiguity can be reduced. Furthermore applying substitution before adjunction means that first a skeleton sentence is built before modifiers are adjoined. This permits reducing the combinatorics introduced by intersective modifiers as the multiple intermediate structures they may license do not propagate to the rest of the sentence tree. In practice however, evaluation is restricted to short input and the algorithm fails to scale up (Gardent and Perez-Beltrachini, 2010).

(Koller and Striegnitz, 2002) present a surface realisation algorithm where (i) the XTAG FB-LTAG grammar (The XTAG Research Group, 2001) is converted to a dependency grammar capturing the derivation trees of XTAG and (ii) a constraint-based dependency parser is used to construct derivation trees from semantic representations. The parser used was specifically developed for the efficient parsing of free word order languages and is shown to efficiently handle both the lexical ambiguity and the lack of order information in the input that are characteristic of surface realisation from a flat semantics. The evaluation however is restricted to a few hand constructed example inputs; and the grammar conversion ignores feature structure information.

To address these shortcomings, (Gardent and Perez-Beltrachini, 2010) present an approach which makes use of the procedure for converting an FB-LTAG to a Feature-Based Regular Tree Grammar (FB-RTG) described in (Schmitz and Roux, 2008). Like in (Koller and Striegnitz, 2002), the initial FB-LTAG is converted to a grammar of its derivation trees. However in this case, the grammar conversion and the resulting feature-based RTGs accurately translates the full range of unification

mechanisms employed in the initial FB-LTAG. An Earley, bottom-up algorithm is developed and the approach is tested on a large benchmark of artificially constructed examples illustrating different levels of linguistic complexity (different input lengths, different numbers of clauses and of modifiers). The approach is shown to outperform the algorithm presented in (Gardent and Kow, 2007) in terms of space. Speed is not evaluated however and the algorithm is not evaluated on the real life data.

Probabilistic techniques have also been proposed to improve e.g., lexical selection, the handling of intersective modifiers and the selection of the best output. For instance, (Bangalore and Rambow, 2000) uses a tree model to produce a single most probable lexical selection while in CCG based White's system (White, 2004), the best paraphrase is determined on the basis of n-gram scores. To address the fact that there are $n!$ ways to combine any n modifiers with a single constituent, (White, 2004) proposes to use a language model to prune the chart of identical edges representing different modifier permutations, e.g., to choose between *fierce black cat* and *black fierce cat*. Similarly, (Bangalore and Rambow, 2000) assumes a single derivation tree that encodes a word lattice (*a {fierce black, black fierce} cat*), and uses statistical knowledge to select the best linearisation. Recently, (Espinosa et al., 2008) adapted the supertagging techniques first proposed for parsing (Bangalore and Joshi, 1999) to surface realisation. Given a treebank in the appropriate format, this technique permits filtering the initial search space by using a model trained on that treebank. Supertagging was shown to improve the performance of symbolic parsers and generators significantly. However, it requires the existence of a treebank in a format appropriate to generate the supertagging model.

In sum, various symbolic and statistical techniques have been developed to improve the efficiency of grammar-based surface realisation. However, statistical systems using supertagging require the existence of a treebank in an appropriate format while the purely symbolic systems described in (Carroll and Oepen, 2005; Gardent and Kow, 2005; Koller and Striegnitz, 2002; Gardent and Perez-Beltrachini, 2010) have not been evaluated on large corpora of arbitrarily long sentences such as provided by the surface realisation (SR) task (Belz et al., 2011).

Recently, (Guo et al., 2011; Bohnet et al., 2011; Stent, 2011) have developed statistical dependency realisers that do not make use of an explicit grammar but use cascaded classifiers and n-gram models to map in SR input data to sentences. They obtain the best results in the SR task partly because, for grammar based systems, converting the provided input into the format expected by the grammar proved to be extremely difficult.

The algorithm we propose departs from these approaches in that it is a grammar-based approach; it is optimised by combining parallel processing, top-down prediction and local bottom-up polarity filtering; and it was evaluated on a large scale using the input data provided by Generation Challenge SR Task.

3 Input Representations

Recently, the Generation Challenge has promoted a Surface Realisation (SR) task (Belz et al., 2011) where the input provided to test and compare surface realisers are (deep or shallow) dependency structures. Here we assume as input to surface realisation, the shallow dependency structures provided by this task namely, unordered trees whose edges are labelled with syntactic functions and whose nodes are labelled with lemmas, part of speech tags, partial morphosyntactic information such as tense and number and, in some cases, a sense tag identifier. All words of the original sentence are represented by a node in the tree. An example of the shallow dependency trees used

for the SR task is given in Figure 1.

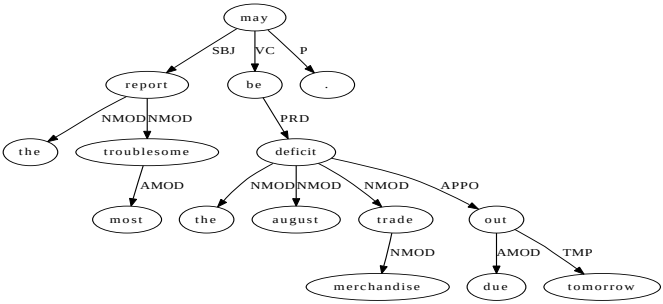


Figure 1: Input shallow dependency tree from the Generation Challenge Surface Realisation Task for the sentence "*The most troublesome report may be the August merchandise trade deficit due out tomorrow .*"

Note that contrary to the flat semantic representations often used by surface realisers, the SR data has a clear tree structure. Thus the combinatorics induced by the lack of order in flat semantic representations is less in this task. Indeed, the algorithm we present exploits this structure to minimize the combinatorics. Similarly, (White, 2006) applies chunking constraints to the graph structure of flat semantic representation to constrain the generation of coordinate structures and address the issue of semantically incomplete phrases.

4 Grammar

Following (Gardent and Perez-Beltrachini, 2010), we perform surface realisation using a Feature-Based Regular Tree Grammar (FB-RTG) describing the derivation trees of a Feature-Based Lexicalised Tree Adjoining Grammar (FB-LTAG, (Joshi and Schabes, 1996)) rather than the FB-LTAG itself. In what follows, we briefly introduce FB-LTAG and the derived FB-RTG used for generation.

4.1 FB-LTAG

The grammar underlying the surface realisation algorithm presented in the next section is an FB-LTAG for English consisting of roughly 1000 trees and whose coverage is similar to XTAG (The XTAG Research Group, 2001).

Figure 2 shows an example FB-LTAG. Briefly, an FB-LTAG consists of a set of elementary trees which can be either initial or auxiliary. Initial trees are trees whose leaves are labeled with substitution nodes (marked with a down-arrow) or terminal categories. Auxiliary trees are distinguished by a foot node (marked with a star) whose category must be the same as that of the root node. In addition, in an FB-LTAG, each elementary tree is anchored by a lexical item (lexicalisation) and the nodes in the elementary trees are decorated with two feature structures called *top* and *bottom* which are unified during derivation. Two tree-composition operations are used to combine trees: substitution and adjunction. Substitution inserts a tree onto a substitution node of some other tree while adjunction inserts an auxiliary tree into a tree. Derivation in an FB-LTAG yields two trees: a *derived tree* which is, like for context free grammars, the tree produced by combining the grammar rules (here, the elementary trees) licensed by the input; and a *derivation tree* which indicates how

the derived tree was built i.e., which elementary trees were used and how they were combined. Figure 3 show the derived and derivation trees associated by the grammar shown in Figure 2 with the sentence "*Which fruit has John eaten?*". For a detailed presentation of the FB-LTAG formalism, the reader is referred to (Vijay-Shanker and Joshi, 1988).

4.2 FB-RTG

As shown in (Koller and Striegnitz, 2002; Gardent and Perez-Beltrachini, 2010), processing the derivation trees of a given FB-LTAG rather than its derived trees is more efficient. Following (Gardent and Perez-Beltrachini, 2010), we therefore use not the initial FB-LTAG described in the previous section, but the FB-RTG grammar of derivation trees that can be derived from it. That is, the surface realisation algorithm first builds a derivation tree. The generated sentence is then extracted from the derived tree[1] which can be reconstructed from this derivation tree using the original FB-LTAG.

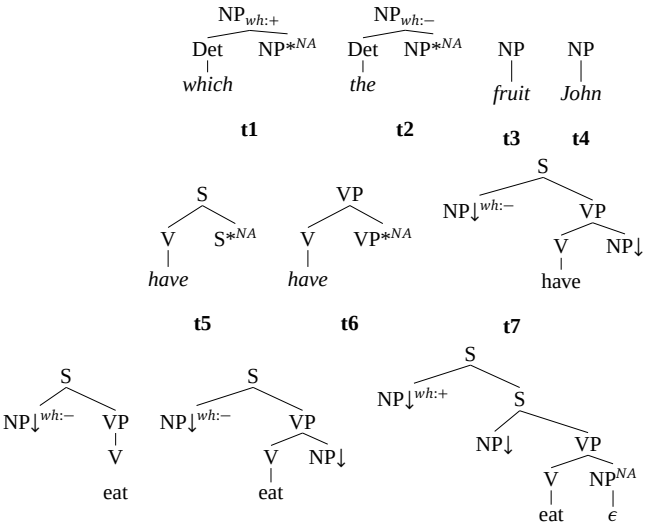
Figure 2 shows an example FB-LTAG and the corresponding FB-RTG. The conversion from FB-LTAG to FB-RTG is described in detail in (Schmitz and Roux, 2008). Intuitively, the FB-RTG representation of an FB-LTAG elementary tree t, is a rule whose left hand side (LHS) describes the syntactic requirement satisfied by t (e.g., S_S for an initial tree rooted in S and VP_A for an auxiliary tree rooted in VP) and whose right hand side (RHS) describes its requirements. Adjunction is handled as an optional requirement which can be satisfied by the adjunction of an empty string and subscripts indicates the nature of the requirement (S for a substitution and A for adjunction). For instance, the rule **r8** in Figure 2 repeated below for convenience, describes the contribution of the elementary tree **t8** lexicalised with the lemma *eat* to a derivation tree as follows: **t8** can satisfy a requirement for a substitution on a node labelled with the S category (LHS with the category S_S) and requires one substitution on a node labelled with the NP category (NP_S on the RHS) and two optional adjunctions of category S and VP respectively (S_A, VP_A on the RHS).

$$S_S^{[t:T,b:B]} \rightarrow eat(S_A^{[t:T,b:B]}\ NP_S^{[t:[wh:-]]}\ VP_A)$$

The derivation process in FB-RTG produces trees that are almost identical to the FB-LTAG derivation trees. Figure 3 shows the FB-LTAG derived, FB-LTAG derivation and FB-RTG derived tree for the sentence "*Which fruit has John eaten?*". When abstracting away from the categorial nodes, the FB-RTG derivation tree mirrors the derivation tree of the original FB-LTAG. The *A* and *S* subscripts indicate which operation was used for combining; and the nodes at which each FB-LTAG elementary tree adjoins or substitutes is encoded by features in these trees: for instance, the subject node of **t9** will have the feature *subject* while its object node will have the feature *object*. By comparing the dependency relations present in the input tree with the feature values given by the grammar, it is thus possible to determine on which nodes of the mother tree in the derivation tree, its daughter trees should combine.

Note that the FB-RTG tree is unordered. During generation, the appropriate linearisation of the lexical items is obtained by constructing the FB-LTAG derived tree from the FB-RTG derivation tree. Morphological realisation is carried out in a post-processing step from the list of lemmas and feature structures decorating the yield of the FB-LTAG derived tree.

[1] in FB-LTAG, the mapping from derivation tree to derived tree is one-to-one.



r1	$NP_A^{[t:T]}$	$\rightarrow$	$\mathit{which}(NP_A^{[t:T,b:[wh:+]]})$
r2	$NP_A^{[t:T]}$	$\rightarrow$	$\mathit{the}(NP_A^{[t:T,b:[wh:-]]})$
r3	$NP_S^{[t:T]}$	$\rightarrow$	$\mathit{fruit}(NP_A^{[t:T]})$
r4	$NP_S^{[t:T]}$	$\rightarrow$	$\mathit{John}(NP_A^{[t:T]})$
r5	$S_A^{[t:T]}$	$\rightarrow$	$\mathit{have}(S_A^{[t:T]})$
r6	$VP_A^{[t:T]}$	$\rightarrow$	$\mathit{have}(VP_A^{[t:T]})$
r7	$S_S^{[t:T,b:B]}$	$\rightarrow$	$\mathit{have}(S_A^{[t:T,b:B]}\ NP_S^{[t:[wh:-]]}\ VP_A\ NP_S)$
r8	$S_S^{[t:T,b:B]}$	$\rightarrow$	$\mathit{eat}(S_A^{[t:T,b:B]}\ NP_S^{[t:[wh:-]]}\ VP_A)$
r9	$S_S^{[t:T,b:B]}$	$\rightarrow$	$\mathit{eat}(S_A^{[t:T,b:B]}\ NP_S^{[t:[wh:-]]}\ VP_A\ NP_S)$
r10	$S_S^{[t:T,b:B]}$	$\rightarrow$	$\mathit{eat}(S_A^{[t:T,b:B]}\ NP_S^{[t:[wh:+]]}\ S_A\ NP_S\ VP_A)$
r11	$X_A^{[t:T,b:T]}$	$\rightarrow$	ϵ

Figure 2: A toy FB-LTAG and the corresponding FB-RTG. For the sake of clarity, feature structures are abbreviated. **r11** (not present in the original FB-LTAG) implements optional adjunction for arbitrary categories with X, a variable ranging over all syntactic categories.

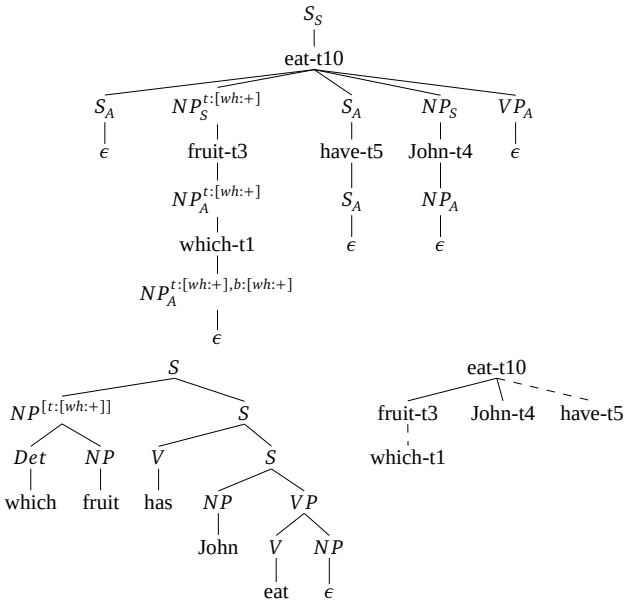


Figure 3: The FB-RTG derivation for "*Which fruit has John eaten*" and the corresponding FB-LTAG derived and derivation trees. In the derivation tree, the nodes are labelled with a lemma/FB-LTAG tree name pair; dashed lines indicate adjunction and solid lines substitution. Adjunction and substitution sites have been omitted.

5 The Surface Realisation Algorithm

Surface realisation starts from the root node of the input tree and processes all children nodes in parallel by spreading the lexical selection constraints top-down and completing the FB-RTG rules bottom-up. Figure 4 shows the architecture of the surface realiser. The controller provides the interface to our surface realization system. It takes a shallow dependency tree as input and produces a ranked list of sentences as output. More specifically, the controller defines a process pool such that each process present in this pool represents a node (a lemma) in the input dependency tree and the communication scheme among processes reflects the dependency relations among nodes in the input dependency tree. In this way, generation is guided by the structure of the input dependency tree.

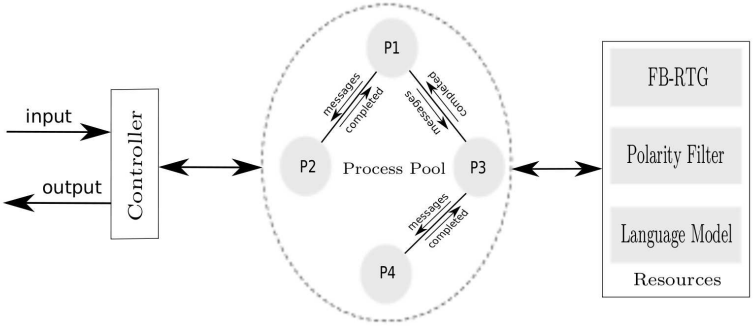


Figure 4: A Parallel Architecture for Surface Realisation

The algorithm proceeds in five major steps as follows.

Top-Down Rule selection and Filtering. Starting from the root node, the input dependency tree is traversed top-down to associate each node in the input tree with a set of grammar rules (from the FB-RTG). This step corresponds to the lexical lookup phase of lexicalist approaches whereby each literal in the input selects the grammar rules whose semantics subsumes this literal. Our approach differs from existing lexicalist approaches however in that it uses the top-down information given by the structure of the input to filter out some possibilities that cannot possibly lead to a valid output. More precisely, for each input node n with lemma w, only those rules are selected which are associated with w in the lexicon. In addition, the left-hand side (LHS) category of each selected rule must occur at least once in the right-hand side (RHS) of the rules selected by the parent node.

For instance, given the input dependency tree shown in Figure 5 for the sentence "*Which fruit has John eaten?*", and the grammar given in Figure 2, all rules **r8**, **r9** and **r10** associated with the lemma '*eat*' will be selected because all of them corresponds to the S[2] rooted initial trees[3].

[2]The controller triggers the root process "eat" with the initial lexical selection constraint (S_S, S rooted initial trees) to generate complete sentences.

[3]The grammar is lexicalised with lemmas rather than forms. The appropriate forms are generated at the end of the generation process based on the lemmas and on the feature structures decorating the yield of the trees output by the generator.

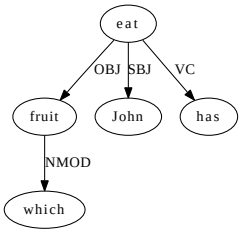


Figure 5: Dependency Tree for "*Which fruit has John eaten*"

$$\begin{array}{lllll} \surd & \textbf{r8} & S_S^{[t:T,b:B]} & \rightarrow & eat(S_A^{[t:T,b:B]}\ NP_S^{[t:[wh:-]]}\ VP_A) \\ \surd & \textbf{r9} & S_S^{[t:T,b:B]} & \rightarrow & eat(S_A^{[t:T,b:B]}\ NP_S^{[t:[wh:-]]}\ VP_A\ NP_S) \\ \surd & \textbf{r10} & S_S^{[t:T,b:B]} & \rightarrow & eat(S_A^{[t:T,b:B]}\ NP_S^{[t:[wh:+]]}\ S_A\ NP_S\ VP_A) \end{array}$$

The parent process creates a new lexical selection constraint message consisting of its RHS requirements in selected RTG rules and passes it to its children processes. In Figure 5, the process associated with the node '*eat*' will send a message consisting of S_A, NP_S and VP_A (RHS requirements of rules **r8**, **r9** and **r10**) to its children processes associated with '*fruit*', '*John*' and '*have*'.

Starting from the trigger initiated by the *controller*, the process of message spreading happens in recursive and parallel manner throughout the process pool reflecting the input dependency tree in a top-down fashion. It eliminates all RTG rules which cannot possibly lead to a valid output well before carrying out any substitution and adjoining operation on the RTG rules.

For instance, the rule **r7** for '*have*' will not be selected because its left-hand side is S_S which does not satisfy the lexical selection constraints (S_A, NP_S and VP_A) sent by its parent '*eat*'.

$$\begin{array}{lllll} \surd & \textbf{r5} & S_A^{[t:T]} & \rightarrow & have(S_A^{[t:T]}) \\ \surd & \textbf{r6} & VP_A^{[t:T]} & \rightarrow & have(VP_A^{[t:T]}) \\ \times & \textbf{r7} & S_S^{[t:T,b:B]} & \rightarrow & have(S_A^{[t:T,b:B]}\ NP_S^{[t:[wh:-]]}\ VP_A\ NP_S) \end{array}$$

Leaf closure. When reaching the leaf nodes of the input tree, the top and bottom feature structures of the rules selected by these leaf nodes are unified. The completed rules of a leaf node are sent back to its parent.

Local Polarity filtering. As mentioned in Section 2, polarity filtering (Gardent and Kow, 2005) eliminates from the search space those sets of rules which cover the input but cannot possibly lead to a valid derivation either because a substitution node cannot be filled or because a root node fails to have a matching substitution site[4] While (Gardent and Kow, 2005) applies polarity filtering to the initial search space (the set of rules selected by all literals in the input), we apply polarity filtering to each local tree while going up the input tree. Thus, this filtering will weed out all combinations

[4]Since it only eliminates combinations that cannot possibly lead to a valid parse, polarity filtering does not affect completeness. Nor does it place any particular constraint in the grammar. All that is required is that the grammar encodes a notion of resources and requirements i.e., of items that cancel each other out. Typically, grammar rules support this constraint in that e.g., the left-hand side of a rule and one category in the right-hand side of another rule can be viewed as canceling each other out if they match.

of mother rules and completed immediate daughter rules which cannot possibly yield a complete tree either because some daughter rule cannot be used or because some requirement of the mother rule cannot be satisfied. For instance, after processing the daughters of the '*eat*' node in the input dependency tree shown in Figure 5, all combinations of **r8** (intransitive '*eat*') with the daughter trees will be excluded. This is because at this stage of processing, the trees built bottom up for '*which fruit*', '*John*' and '*have*' includes two NPs with LHS category NP_S (Figure 6) while the **r8** rule only requires one such NP. That is, for this rule, the completed daughter rule for *which fruit* will show up as a superfluous syntactic resource.

$$\begin{array}{lll} \times & \textbf{r8} & S_S^{[t:T,b:B]} \rightarrow eat(S_A^{[t:T,b:B]}\ NP_S^{[t:[wh:-]]}\ VP_A) \\ \surd & \textbf{r9} & S_S^{[t:T,b:B]} \rightarrow eat(S_A^{[t:T,b:B]}\ NP_S^{[t:[wh:-]]}\ VP_A\ NP_S) \\ \surd & \textbf{r10} & S_S^{[t:T,b:B]} \rightarrow eat(S_A^{[t:T,b:B]}\ NP_S^{[t:[wh:+]]}\ S_A\ NP_S\ VP_A) \end{array}$$

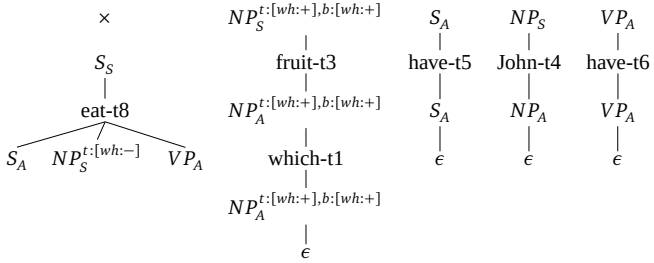


Figure 6: Polarity Filtering will filter out the **r8** rule for '*eat*' since one of trees '*which fruit*' or '*John*' would then appear as a superfluous syntactic resource as illustrated by the above derivation.

By restricting polarity filtering to local input trees, we avoid the computation of the very large automaton required when filtering the global initial search space as done in (Gardent and Kow, 2005).

As noted by one of our reviewers, supertagging models can probably approximate local polarity filtering. For instance, a supertagger model might learn that an intransitive category is very unlikely whenever the input dependency tree contains one or more core arguments.

The combined effect of top-down filtering and local polarity filtering avoids considering most of RTG rules which can never lead to valid output well before carrying out any substitution and adjoining operation on the RTG rules to try to complete them. The Earley, bottom-up algorithm (Gardent and Perez-Beltrachini, 2010) also achieves some amount of top-down filtering during its prediction stage but the lexical selection constraint is limited to the top of the RHS requirements of the RTG rule being processed, hence it may try completing the RTG rules which cannot possibly lead to a valid output whereas in our proposed approach all RHS requirements of the selected RTG rules are available as the lexical selection constraint information during both top-down filtering and local polarity filtering steps.

Bottom-Up generation. For each local tree in the input, the rule sets passing the local polarity filter are tried out for combination. The completed daughter RTG rules are combined to the local

initialized RTG rule using substitution and adjoining operations. The local initialized RTG rule fails to complete if any feature conflicts are found.

Note that for each rule set let through by polarity filtering, the category and the number of daughter trees exactly match the requirement of the associated mother rule. For instance, as explained above, the rule **r8** representing an intransitive use of the verb '*eat*' is ruled out by polarity filtering since it does not permit "consuming" the NP_S resource provided by one of NPs '*which fruit*' or '*John*'. Conversely, given an input tree of the form `eat(john, has)`, the rules **r9** and **r10** representing a transitive use of the verb '*eat*' would be filtered out by polarity filtering. As a result, the intermediate structure shown below will not be computed. That is, while the global polarity filtering used in (Gardent and Kow, 2005) permits weeding out global combination of trees that are invalid, local polarity filtering additionally permits reducing the number of intermediate structures built first, because there is no need for prediction i.e., for active chart items and second, because intermediate structures that cannot possibly lead to a valid derivation are not built.

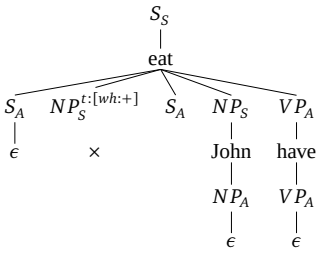


Figure 7: Given the input tree `eat(john, has)`, local polarity filtering filters out this intermediate structure because it cannot be completed given the input

N-gram filtering using a Language Model. To further prune the search space and to appropriately handle word order, the SR algorithm also integrates a language model and can be parametrized for the number of best scoring n-grams let through after each bottom-up generation step. In this way, not all possible orderings of intersective modifiers are produced, only those that are most probable according to the language model.

6 Empirical Evaluation

We now report on the results obtained when running the algorithm and the grammar described above on the shallow input data provided by the Generation Challenge Surface Realisation Task. Because we are presenting an algorithm for surface realisation rather than a surface realiser, the main focus of the evaluation is on speed (not coverage or accuracy). Nevertheless, we also report coverage and BLEU score as an indication of the capabilities of the surface realiser i.e., the algorithm combined with the grammar and the lexicon.

6.1 Runtimes

The SR data on which we evaluate our surface realisation algorithm are the shallow dependency trees described in Section 3.

We use as a baseline the FB-RTG based lexicalist approach (**B**ASELINE) described in (Narayan, 2011). In this approach, FB-RTG rules are selected top-down following the structure of the input dependency tree and all FB-RTG rules selected for a given local input tree are tried out for combination using a chart-based approach. This baseline thus permits observing the impact of the various optimisations described below. In future work, it would be interesting to obtain time information from the systems participating in the SR challenge and to compare them with those of our system.

TDBU-PAR (top-down, bottom-up and parallelised) is the algorithm presented here running on a 4 core system. To evaluate the impact of parallelism on runtimes, we also computed runtimes for a sequential version of the same algorithm (**T**DBU-SEQ). In **T**DBU-SEQ, daughter subtrees (processes) of the input dependency tree are processed sequentially.

Table 1 shows the runtimes for the three surface realisation algorithms **B**ASELINE, **T**DBU-SEQ and **T**DBU-PAR with varying sizes of sentences. For the TDBU algorithms, the n-gram filtering is set to 10 that is, for each local input tree, the 10 best n-grams are passed on. We split the data into 4 sets according to the input length where the input length is the number of nodes (or words) in the input dependency tree. The average number of words in a sentence in the first set $S(0-5)$ is 4, in the second set $S(6-10)$, 7, in the third set $S(11-20)$, 15, and in the final set $S(All)$ (all lengths), 17. The maximum length of a sentence in the final set $S(All)$ is 74. To make comparisons between **B**ASELINE, **T**DBU-SEQ and **T**DBU-PAR possible, the maximum arity of words present in the input dependency trees is set to 3 (because BASELINE mostly fails on input containing nodes with higher arity).

Algorithm	Sentences (Length L)							
	$S(0-5)$		$S(6-10)$		$S(11-20)$		$S(All)$	
	Total	Succ	Total	Succ	Total	Succ	Total	Succ
	1084	985	2232	1477	5705	520	13661	2744
BASELINE	0.85	0.87	10.90	10.76	110.07	97.52	–	–
TDBU-SEQ	1.49	1.63	2.84	3.64	4.36	6.03	4.52	3.18
TDBU-PAR	1.53	1.66	2.56	3.28	2.66	4.14	2.57	2.78

Table 1: Comparison between generation times (seconds)

BASELINE turns out to be faster than **T**DBU-PAR and **T**DBU-SEQ for sentences of smaller length (≤ 5). It can be explained because of the parallelism and the multiprocessing overhead. But **T**DBU-PAR and **T**DBU-SEQ leaves behind **B**ASELINE for longer sentences. For input longer than 10, the simple **B**ASELINE algorithm times out whereas **T**DBU-PAR remains stable. For $S(All)$, **T**DBU-PAR achieves a reasonable average of 2.57 seconds for all sentences (Total) and 2.78 seconds for successful sentences (Succ).

Table 1 does not show a big difference in performance between **T**DBU-PAR and **T**DBU-SEQ because the maximum arity of the input dependency trees is kept low (maximum 3). In Table 2, we split the data by arity whereby the dataset $S(i)$ consists of input dependency trees with maximum arity i. As can be seen, the difference between the two algorithms steadily increases with the arity of the input thereby demonstrating the impact of parallelism.

6.2 Coverage and Accuracy

The grammar and lexicon used to test the surface realisation algorithm presented in this paper are under development so that coverage and accuracy are still low. Table 3 shows the coverage and

Algorithm	Sentences (Arity)											
	$S(1)$		$S(2)$		$S(3)$		$S(4)$		$S(5)$		$S(6)$	
	Total	Succ	Total	Succ	Total	Succ	Total	Succ	Total	Succ	Total	Succ
	190	178	1218	964	3619	1039	5320	605	2910	137	1093	18
TDBU-SEQ	0.89	0.94	2.52	2.63	3.65	3.39	5.07	4.54	5.24	4.62	8.20	7.29
TDBU-PAR	0.97	1.03	2.35	2.50	2.63	3.10	2.91	3.77	2.86	3.88	3.09	4.76

Table 2: Comparison between generation times (seconds) with varying arities.

accuracy (on the covered sentences) results obtained for sentences of size 6 (S-6), 8 (S-8) and all (S-All). The dataset S-All differs from the dataset $S(All)$ discussed in previous section. S-All considers all sentences without any restriction over the maximum arity in the input dependency trees. S-All consists of 26725 sentences with the average length of 22 and the maximum length of 134. The maximum arity in these sentences varies from 1 to 18 with an average of 4.

Data Type	Total	Coverage (#)		Coverage (%)	BLEU Score
		Covered	Uncovered		
S-6	3877	3506	371	90.43	0.835
S-8	3583	3038	545	84.79	0.800
S-All	26725	10351	16374	38.73	0.675

Table 3: Coverage and Bleu Scores for covered sentences.

As can be seen coverage markedly decreases for longer sentences. Error mining on this data indicates that failure to generate is due mostly to complex sentence coordinations (e.g., verb coordination, gapping phenomenon) (Sarkar and Joshi, 1996) which could be very common in sentences of average length 22 in S-All. Other failure causes are inadequate treatments of multiword expressions and foreign words.

7 Conclusion

We presented a novel algorithm for surface realisation with lexicalised grammar which takes advantage of the input structure (a tree) to filter the initial search space both top-down and bottom up; and to parallelise processes. We evaluated this algorithm on large scale data and showed that it drastically reduces runtimes on this data when compared to a simple lexicalist approach which explores the whole search space.

As mentioned in section 3, the input data provided by the SR task differs from the flat semantic representations assumed by most existing surface realisers in that it displays a clear tree structure. The algorithm presented here makes use of that structure to optimize performance. In future work, we plan to investigate whether the hybrid top-down, bottom-up approach we developed to guide the SR search can be generalised to the graph structure of semantic representations.

Acknowledgments We would like to thank Laura Perez-Beltrachini for useful discussion on generation using the FB-RTG. We are grateful to the Generation Challenge SR Task organisers for providing us with the SR data and to Anja Belz and Mike White for providing us with the task evaluation scripts. The research presented in this paper was partially supported by the European Fund for Regional Development within the framework of the INTERREG IV A Allegro Project.

References

Bangalore, S. and Joshi, A. K. (1999). Supertagging: An approach to almost parsing. *Computational Linguistics*, 25(2):237–265.

Bangalore, S. and Rambow, O. (2000). Using TAGs, a tree model and a language model for generation. In *Proceedings of the 5th International Workshop on Tree Adjoining Grammars and Related Formalisms (TAG+5)*, Paris, France.

Belz, A., White, M., Espinosa, D., Kow, E., Hogan, D., and Stent, A. (2011). The first surface realisation shared task: Overview and evaluation results. In *Proceedings of the 13th European Workshop on Natural Language Generation (ENLG)*, Nancy, France.

Bohnet, B., Mille, S., Favre, B., and Wanner, L. (2011). <stumaba >: From deep representation to surface. In *Proceedings of the Generation Challenges Session at the 13th European Workshop on Natural Language Generation*, pages 232–235, Nancy, France. Association for Computational Linguistics.

Bonfante, G., Guillaume, B., and Perrier, G. (2004). Polarization and abstraction of grammatical formalisms as methods for lexical disambiguation. In *Proceedings of the 20th International Conference on Computational Linguistics (COLING)*, Geneva, Switzerland.

Carroll, J., Copestake, A., Flickinger, D., and Paznański, V. (1999). An efficient chart generator for (semi-)lexicalist grammars. In *Proceedings of the 7th European Workshop on Natural Language Generation*, pages 86–95, Toulouse, France.

Carroll, J. and Oepen, S. (2005). High efficiency realization for a wide-coverage unification grammar. In *Proceedings of the 2nd International Joint Conference on Natural Language Processing (IJCNLP)*, Jeju Island, Korea.

Copestake, A., Lascarides, A., and Flickinger, D. (2001). An algebra for semantic construction in constraint-based grammars. In *Proceedings of the 39th Annual Meeting of the Association for Computational Linguistics*, Toulouse, France.

Espinosa, D., Rajkumar, R., White, M., and Berleant, S. (2010). Further meta-evaluation of broad-coverage surface realization. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 564–574, Cambridge, MA. Association for Computational Linguistics.

Espinosa, D., White, M., and Mehay, D. (2008). Hypertagging: Supertagging for surface realization with CCG. In *Proceedings of the 40th Annual Meeting of the Association of Computational Linguistics (ACL): Human Language Technologies (HLT)*, pages 183–191, Columbus, Ohio. Association for Computational Linguistics.

Gardent, C. and Kow, E. (2005). Generating and selecting grammatical paraphrases. In *Proceedings of the 10th European Workshop on Natural Language Generation (ENLG)*, Aberdeen, Scotland.

Gardent, C. and Kow, E. (2007). A symbolic approach to near-deterministic surface realisation using tree adjoining grammar. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 328–335, Prague, Czech Republic. Association for Computational Linguistics.

Gardent, C. and Perez-Beltrachini, L. (2010). RTG based surface realisation for TAG. In *Proceedings of the 23rd International Conference on Computational Linguistics (COLING)*, pages 367–375, Beijing, China.

Guo, Y., Hogan, D., and van Genabith, J. (2011). Dcu at generation challenges 2011 surface realisation track. In *Proceedings of the Generation Challenges Session at the 13th European Workshop on Natural Language Generation*, pages 227–229, Nancy, France. Association for Computational Linguistics.

Joshi, A. K. and Schabes, Y. (1996). Tree-adjoining grammars. *Handbook of Formal Languages and Automata*.

Koller, A. and Striegnitz, K. (2002). Generation as dependency parsing. In *Proceedings of the 40th Annual Meeting of the Association of Computational Linguistics (ACL)*, Philadelphia.

Narayan, S. (2011). RTG based surface realisation from dependency representations. Master's thesis, University of Nancy 2 and University of Malta. Erasmus Mundus Master "Language and Communication Technology".

Sarkar, A. and Joshi, A. K. (1996). Coordination in tree adjoining grammars: Formalization and implementation. In *Proceedings of the 16th International Conference on Computational Linguistics (COLING)*, pages 610–615.

Schmitz, S. and Roux, J. L. (2008). Feature unification in TAG derivation trees. In *Proceedings of the 9th International Workshop on Tree Adjoining Grammars and Related Formalisms*, Tubingen, Germany.

Shieber, S. (1993). The problem of logical form equivalence. *Computational Linguistics*, 19(1):179–190.

Shieber, S. M., Noord, G. V., Pereira, F. C. N., and Moore, R. C. (1990). Semantic-head-driven generation. *Computational Linguistics*, 16.

Stent, A. (2011). Att-0: Submission to generation challenges 2011 surface realization shared task. In *Proceedings of the Generation Challenges Session at the 13th European Workshop on Natural Language Generation*, pages 230–231, Nancy, France. Association for Computational Linguistics.

The XTAG Research Group (2001). A lexicalised tree adjoining grammar for english. Technical report, Institute for Research in Cognitive Science, University of Pennsylvannia.

Vijay-Shanker, K. and Joshi, A. (1988). Feature structures based tree adjoining grammars. In *Proceedings of the 12th International Conference on Computational Linguistics*, Budapest, Hungary.

White, M. (2004). Reining in CCG chart realization. In *Proceedings of the third International Conference on Natural Language Generation (INLG)*, pages 182–191, Brighton, UK.

White, M. (2006). Efficient realization of coordinate structures in combinatory categorial grammar. *Research on Language and Computation*, 4(1):39–75.

A Comparison of Syntactic Reordering Methods for English-German Machine Translation

Jiří Navrátil Karthik Visweswariah Ananthakrishnan Ramanathan
IBM Research
jiri@us.ibm.com, v-karthik@in.ibm.com, aramana2@in.ibm.com

ABSTRACT
We describe two methods for syntactic source reordering developed for English-German SMT. Both methods learn from bilingual data accompanied by automatic word alignments to reorder the source such that it resembles that of the target. While the first method is an extension of a parse-based algorithm and accommodates contextual triggers in the parse, the second method uses a linear feature-based cost model along with a Traveling Salesman Problem (TSP) solver to perform the reordering. Our results indicate that both methods lead to improvements in BLEU scores in both directions, English→German and German→English. Significant gains in human translation quality assessment are observed for German→English, however, no significant changes are observed in the human assessment for English→German.

KEYWORDS: Parse-based reordering, TSP, English-German SMT.

Proceedings of COLING 2012: Technical Papers, pages 2043–2058,
COLING 2012, Mumbai, December 2012.

1 Introduction

Language-specific word order differences have presented a long-standing challenge in the development of statistical machine translation (SMT) systems. Most mainstream SMT approaches do incorporate word order information, either implicitly (e.g., as part of word phrases), or explicitly, e.g., by means of lexicalized distortion models. However, challenges remain, particularly in modeling long-range word movement. Furthermore, certain language pairs, more than others, exhibit particularly demanding reordering patterns. One such pair is English-German - the focus of this work. With its long-range verb movement and its use of separated verb prefixes, German and English word ordering can induce movements spanning an entire sentence. Section 1.1 describes some of these phenomena more in detail.

Recent improvements in phrase-based SMT algorithms have included source word reordering so as to resemble its target word order. A reordering typically uses rules obtained either manually, e.g., (Niessen and Ney, 2001; Collins et al., 2005), or derived from data by automatic means, e.g., (Xia and McCord, 2004; Rottmann and Vogel, 2007; Zhang et al., 2007; Crego and Habash, 2008; Niehues and Kolss, 2009).

In this paper we further develop two methods first introduced in (Visweswariah et al., 2010, 2011) with the aim to address specific issues arising in English↔German SMT. In particular, we extend the parse-based reordering introduced for a variety of language pairs in (Visweswariah et al., 2010) by creating rules capable of capturing essential contextual triggers in the parse tree hierarchy. For the second method we describe refinements of the feature-based reordering (Visweswariah et al., 2011), which corresponds to solving a Traveling Salesman Problem (TSP). While the method described in (Visweswariah et al., 2011) carries over and performs well for German→English, for English→German we propose a method to integrate the English parse into the reordering model focusing on the issues in English→German reordering. We compare the performance of both methods against various baselines and across multiple evaluation domains for both English→German and German→English directions. Our results indicate that both methods achieve significant improvements in translation quality, as measured by automatic metrics, and some improvements when judged by human reviewers.

Following the description of the methods and refinements in Section 2 we discuss related work in Section 3. Experimental evaluation, results, and discussion are presented in Section 4.

1.1 German Word Order

Despite their common heritage, German and English word order can differ rather dramatically. Most frequently such differences relate to verbs, particularly to verb groups, including auxiliary, main verbs, and their participles. Verb movements tend to span large portions of the sentence, thus presenting a considerable challenge to basic reordering models in standard SMT systems.

Word ordering can often be identical to that in English, as shown in an example in Figure 1. At the same time, an addition of a modal verb in the same example sentence triggers a movement of the main verb to the end of the clause, as shown in Figure 2. Similarly distant verb movements also occur in subordinate clauses. Besides verb movement, word order may vary in other parts of the sentence, including negation, adverbial phrases, etc.

2 Data-Driven Syntactic Reordering

We use data-driven syntactic reordering to mitigate some of the differences in word order mentioned above. Specifically, we investigate two methods of syntax-based reordering: (1) an

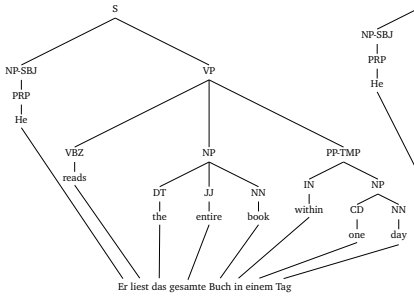


Figure 1: A single verb sentence with monotone ordering pattern

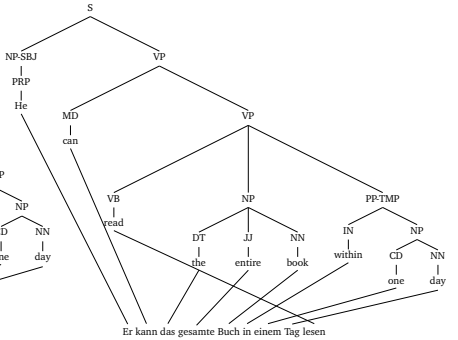


Figure 2: A modal verb triggering a reordering pattern

extension of the parse-based word reordering introduced in (Visweswariah et al., 2010), and (2) feature-based reordering model described in (Visweswariah et al., 2011). Both methods use a bilingual corpus, automatic word alignments, and some source syntax information (parse tree in the first case, POS in the second) to train a reordering model. In both cases the objective is to minimize a measure of an overall distortion observed in the word alignments. We apply the reordering to source sentences in both the training and the test of the system as part of preprocessing.

2.1 Parse-Based Reordering

Visweswariah et al. (Visweswariah et al., 2010) described reordering of each sentence using a set of rules applied to the parse tree of the source sentence. The goal of these rules is to make the source word order resemble the expected target order. Given automatic word alignments and source parses, the rules are inferred from a training corpus.

The parse-based model is probabilistic in nature. Given a source constituency tree, S, it aims to assign the highest probability $P(T|S)$, among all possible reordered trees T, to a tree with its constituents reordered so as to reflect the expected target word order. The model restricts itself to trees that can be obtained only by permuting child nodes of any of its non-terminal constituents, and it makes a simplifying assumption that the children of a node are permuted independently of any other node in the tree. Thus, the overall probability simplifies to a product of constituent-level permutations:

$$P(T|S) = \prod_{n \in I(S)} P(\pi(d_n)|d_n) \quad (1)$$

where n denotes a node, $I(S)$ the set of all non-terminal (interior) nodes of the tree, d_n are the children of the node n, and π is the permutation function. Given the word alignment information, each node in d_n, is permuted based on the individual average *target* position. The average target position is calculated using the alignments $a(w)$ as follows:

$$tpos(n) = \frac{1}{|D(n)|} \sum_{w \in D(n)} a(w) \quad (2)$$

whereby $D(n)$ denotes the set of all descendant leaves (words) of a node n, and $a(w)$ is a function returning the position index of a source word w in the target translation. Each such permutation is recorded over the entire training corpus $\mathscr{C}$ and its permutation probabilities are then estimated to maximize the likelihood $\prod_{S \in \mathscr{C}} P(T|S)$:

$$P(\pi(d_n)|d_n) = \frac{count(\pi(d_n))}{count(d_n)} \tag{3}$$

As already pointed out in (Visweswariah et al., 2010), the above model bears several weaknesses. Besides generic issues regarding parser and alignments accuracy, the assumed independence of a permutation from other nodes (context) can be harmful. This turns out to be particularly true for German. As in the example in Figure 2, a full verb following a modal verb is typically parsed as a VP child node of a VP modal node. In this example the main verb should be placed at the end of the clause due to the presence of the modal, while in the absence thereof typically no reordering occurs (Figure 1). The lack of such distinction in the simplified $\pi(d_n)$ renders extracted rules indiscriminative. The authors reported improvements in translation quality for several languages (French, Spanish, Hindi) with the exception of German where no gains over an unreordered baseline were observed highlighting this weakness as a probable cause.

We propose extending the permutation probability function to include contextual information from S. In this extension each permutation probability is now a function of the permutation node set d_n and a context node subset $\phi_n \in S$:

$$P(T|S) = \prod_{n \in I(S)} P(\pi(d_n)|d_n, \phi_n) \tag{4}$$

Although the subset ϕ can be arbitrary we choose ϕ_n to constitute subtrees of S that are related to d_n via its parental lineage as well as its siblings. For instance, in the Figure 2 this relation would be the subtree including the non-terminals (top-down): S, VP, MD, VP, VB, NP, and PP-TMP. More specifically, in our experiments we investigated ϕ to include the parent of d_n, i.e n, siblings of n (both to its left and right), as well as k levels of grand parents (with k varying between 1 and 5). We relaxed the subtree matching by assuming "wildcards" between any of the siblings. This increases the rule recall for parses with minor variations from the observed patterns (in other words, we apply a *tree grep* instead of a strict tree match).

The probability of a permutation is estimated as in Eq. (3) with observation counts now collected with respect to the context ϕ of each observed permutation. A permutation rule pruning is performed based on an absolute observation count (in our case any rule with less than 20 observations) as well as a significance threshold (count must be 20% higher relative to the next competing permutation). We only retain the best permutation (in the maximum likelihood sense) for a unique pattern (ϕ, d_n), thus, a final rule now consists of a left- and right-hand side: $(\phi, d_n) \rightarrow \pi(\phi, d_n)$. Reusing the example in Figure 2 the best applicable rule actually extracted from the training corpus is:

```
[S [VP MD [VP VB NP PP-TMP VP] VP] S] --> 1 2 0
```

which enacts a move of the verb "read" (VB) to the end of the sentence and resolves the crossing alignment shown in Figure 2.

Given a parse tree all matching rules are applied recursively (matched always against the original parse) to create a new reordered tree.

2.2 TSP-Based Reordering

In (Visweswariah et al., 2011) a reordering model was proposed that does not require a parser, and learns to reorder words based on reference reorderings derived from hand alignments. The model assigns pairwise costs $c(m,n)$ for word w_m immediately preceding word w_n. The cost of a reordering permutation π for a sentence $\mathbf{w}$ is the sum of these pairwise costs:

$$C(\pi|\mathbf{w}) = \sum_i c(\pi_i, \pi_{i-1}).$$

A sentence $\mathbf{w}$ is reordered by choosing the permutation that minimizes the cost $C(\pi|\mathbf{w})$. The minimization problem is an asymmetric TSP problem, which is converted to a symmetric TSP problem with double the number of states (Visweswariah et al., 2011) and then is solved using the Lin-Kernighan heuristic. The costs $c(m,n)$ are parametrized as a linear model

$$c(m,n) = \theta^T \Phi(\mathbf{w}, m, n)$$

where Φ is a vector of binary feature vectors as described in (Visweswariah et al., 2011). θ is a weight vector learned from reorderings derived from hand alignments using the MIRA update algorithm.

Although the base reordering model does not require a parser, for English→German reordering we experimented with using the TSP model that works on top of an input parse tree. Since English-German word order differences mainly relate to verb movements, we transform the parse tree so that any internal node which has a verb as a descendant is expanded down to its children. If a node does not have a verb descendant we represent the entire subtree by the constituent label of the node. Thus the parse tree in Figure 2 gets transformed to the following being passed as input to the reordering model: *NP-SBJ can read NP PP-TMP*.

2.2.1 Features

We adopted the same base features as described in (Visweswariah et al., 2011) and also experimented with additional features specific to word order differences in English-German. The basic features $\Phi(\mathbf{w}, m, n)$ are binary features that fire based on the identity of the words w_m and w_n and the POS tags of these words. Additionally there are features that examine the identities of words and POS tags one word to the left and right of the positions m and n (see (Visweswariah et al., 2011) for details of feature templates used).

Specific to the English→German direction reordering we use transformed parses (see above) as input and use the constituent labels instead of the words. Additionally, to handle the fact that verbs move to the end of the clause in subordinate clauses, we mark if a verb is a descendant of an SBAR node in the parse tree on the POS tag of the word.

Specific to the German→English direction we added an extra feature template that fires if word w_m and w_n are verbs and there is no verb between positions m and n.

2.3 Manual Rules

We do not have a suitable German parser. Therefore, as an alternative method to the TSP-based method (in the German→English direction only) we consider reordering using hand-crafted rules. The rules use German-side POS sequence as features and focus on verbs:

- Move full verb behind its closest auxiliary to the left
- Move negation behind closest auxiliary to the left
- In a subordinate clause, move auxiliary and full verbs behind an estimated subject
- Void any of the above rules that would cross a barrier token (e.g., a comma)

The rules are applied to a sentence repeatedly until there is no word movement. As will be shown, this small set of heuristic rules improves the output quality. We will compare this method with an unreordered baseline as well with a system pre-ordered via the TSP method.

2.4 Sentence Pre-Selection

The quality of reordering rules, i.e. their precision, depends on several factors: (1) the alignment, tagging, and parsing accuracy, and (2) the bilingual training data quality. The latter is considered here from a point of view of suitability for extraction of meaningful reordering patterns. Bilingual corpora typically contain sentence pairs with a varying degree of mutual parallelism. While a sentence pair may be considered an acceptable translation, the individual sides may differ significantly in terms of their syntactic and clausal structure. Non-literal translations may introduce considerable noise into the rule extraction. Consider this example taken from the Europarl (Koehn, 2005) corpus:

- English: *A ban would have less serious repercussions in this respect if it applied throughout the EU.*
- German: *Im Falle eines europäischen Verbots von Nachtflügen sind diese Folgen weniger stark ausgeprägt.*
- Gloss: *In case of a European ban of night flights are these consequences less strongly contrastful.*

Obviously, issues will arise for rule extraction given that these two sentences are composed quite differently. In particular the verb "applied" remains unaligned in this case thus introducing an error to the estimated target position for the corresponding set of node descendants, as described in Section 2.1.

To mitigate the impact of noise in bilingual data we propose an automatic sentence preselection algorithm. The goal here is to assign a quality score to each sentence pair given its automatic alignment and to select a subset of higher quality (i.e. literally translated) sentences for the rule extraction. We borrow the confidence measure proposed originally for block extraction in (Huang, 2009) and adopt it as an indicator for sentence parallelism. A description of the algorithm follows.

Let (V,W) denote a bilingual sentence pair where $V = \{v_1,...,v_I\}$ and $W = \{w_1,...,w_J\}$ is the source and the target sentence, respectively. The sentences are word-aligned via a set of alignment links $A = \{a_{ij}\}$. The alignment A represents the best way to relate the content of V to W and is obtained by automatic means (e.g. Giza or Maxent methods). The quality measure involves calculating two quantities:

$$P(W|V,A) = \epsilon \prod_{j=1}^{J} \sum_{\forall i:a_{ji} \in A} p(w_j|v_i) \tag{5}$$

which is the lexical probability of the target sentence given the source words and their alignment, and

$$P(W|V) = \frac{\epsilon}{(I+1)^J} \prod_{j=1}^{J} \sum_{i=1}^{I} p(w_j|v_i) \tag{6}$$

the lexical probability of W given V, independent of A. The terms $p(w_j|v_i)$ are Model 1 word translation probabilities as estimated during training. The constants ϵ and $\frac{1}{(I+1)^J}$ are due to the simplifying assumptions of Model 1 as described in (Brown et al., 1993), namely sentence lengths, and alignments being all equally probable. These constants will later cancel out. Similar to the above, transposed quantities $P(V|W)$ and $P(V|W,A')$ are also obtained using Model 1 probabilities in in the reverse direction along with their corresponding alignment A'. Noting that

$$P(W,A|V) = P(A)P(W|V,A) = \frac{1}{(I+1)^J} P(W|V,A) \tag{7}$$

we now define the bilingual sentence confidence score as follows:

$$F(V,W) = \frac{1}{J} \log \frac{P(A,W|V)}{P(W|V)} + \frac{1}{I} \log \frac{P(A',V|W)}{P(V|W)} \tag{8}$$

Since $F(V,W)$ is an average of alignment posteriors $P(A|V,W)$ and $P(A'|V,W)$, we expect it to produce large values for translations with sharp posterior distributions which, in turn, can be viewed as relatively literal and complete. For sentence pairs with a lower degree of parallelism, for incomplete, or perhaps incorrect translations $F(V,W)$ should tend to be small.

The overall sentence pre-selection step consists of calculating $F(V,W)$ values for all sentence pairs in the reordering training corpus followed by thresholding and selecting a certain proportion of best scoring sentences for the rule training. In our case the proportion was set to be around 50%.

In our preliminary experiments we observed translation quality gains of up to 1 point BLEU due to this selection method.

3 Related Work

A significant quantity of work in syntax based reordering has accumulated in the machine translation literature. Relevant to our topic, there have been studies investigating sources of improvements (Zwarts and Dras, 2007) due to syntax-based reordering. Initial efforts (Niessen and Ney, 2001) were made at improving German-English translation using handwritten rules by handling two phenomena: question inversion and detachable verb prefixes in German. In (Collins et al., 2005; Carl, 2007) rules are developed for translation from German to English based on source POS and manual rules covering a variety of patterns including verb movement.

There have been studies that try to learn rules from the data, among others (Ringger et al., 2004; Rottmann and Vogel, 2007; Zhang et al., 2007; Crego and Habash, 2008; Niehues and Kolss, 2009; Khalilov and Sima'an, 2011). Work by Rottmann and Vogel utilizes sequences of source POS and the corresponding alignments to automatically extract reordering rules along with their left and right context. They then reorder a source sentence using applicable rules thus obtaining a word lattice for decoding. Building on this work, Niehues

and Kolss explore an extension of such rules introducing a variable length gaps to better generalize long range reordering patterns. Both studies report significant gains in BLEU scores for both English→German and German→English translation.

The work in (Rottmann and Vogel, 2007; Niehues and Kolss, 2009) relates to both our parse-based as well as feature-bases reordering methods in that they take the context of the reordered tokens into account - a key to English-German word order. Our method, like (Niehues and Kolss, 2009), allows for gaps in the matching patterns. In contrast to the above mentioned work, the parse-based method operates on the entire source parse, including coarse levels where reordering can induce long range word movements. We use contextual information from the parse tree thus modeling left and right context which, in contrast to (Niehues and Kolss, 2009), is hierarchical. Our second, feature-based method operates on word level and takes into account arbitrary syntactic and lexical source features. The method utilizes machine learning to find appropriate model to capture most beneficial source word order, given a set of automatic alignments.

Recently, reordering models that can learn to reorder source sentences to make them match the target language order without requiring a parser have been proposed. (Tromble and Eisner, 2009) propose a model based on the Linear Ordering Problem, where for each pair of words in the sentence the cost that one of them occurs somewhere before the other is modeled. (DeNero and Uszkoreit, 2011) take a two pronged approach where they first learn to parse the sentence and then learn a reordering model on the resultant parses. (Visweswariah et al., 2011) proposed a model based on the Travelling Salesman Problem to learn to reorder sentences where costs of a word immediately preceding another word in the sentence are learned. This model was shown to be better than (Tromble and Eisner, 2009) in terms of reordering performance. In this work we extend and apply the (Visweswariah et al., 2011) to German-English reordering.

While the focus of our paper is on pre-ordering techniques, there has been considerable work on handling the reordering problem as part of the decoding process (Chiang, 2007; Yamada and Knight, 2002; Galley et al., 2006; Liu et al., 2006; Zollmann and Venugopal, 2006). These approaches are computationally expensive compared with phrase based systems due to the inclusion of bilingual parsing in the decoding process.

4 Experimental Evaluation

4.1 Parse-Based Rules

The rule extraction was performed using approximately 2.6 million sentence pairs obtained through the sentence pre-selection process as described in Section 2.4, operating at a 50% sentence rejection rate. The sentence pairs were aligned by a maximum entropy aligner (Ittycheriah and Roukos, 2005) trained using a subset of the Europarl corpus, and selected computer manual corpora). A maximum entropy parser (Ratnaparkhi, 1999) was used to generate the parse trees for the English side in the English→German direction, and a maximum entropy tagger to generate POS (using the Stuttgart-Tübingen tag set (Schiller et al., 1995)) for the German side in the German→English direction. With a significance threshold of 1.2 and minimum count threshold of 20, we obtain about 3200 rules in the final reordering model.

4.2 TSP-Based Model

For both English→German and German→English reordering we train the TSP-based models on a set of hand alignments (roughly 30K sentences) and a subset of machine alignments selected using the sentence pre-selection method described in Section 2.4 (roughly 300k sentences). For English→German we use the transformed English parse as described in Section 2.2 instead of the original word sequence.

During development stage we used the monolingual BLEU score (mBLEU) to compare the reordered output to a reference reordering derived from hand alignments on a test set of 400 sentences from the news domain. For German→English we obtain larger improvements, going from a mBLEU score of 61.5 for unreordered German to 72.5 for German reordered using the TSP-based model. For English→German we get a relatively modest improvement; going from 64.2 for unreordered English to 67.6 for reordered English.

4.3 SMT Model

The phrase-based systems were trained in both directions using same amounts of training data. In a first stage, smaller models were created adhering to the WMT2010 constrained training condition (WMT Website, 2010). In a second stage, about 16M sentence pairs spanning a variety of publicly available (e.g. Europarl) as well as internal corpora (IT and news domains) served the training of unconstrained systems. The phrase pairs were extracted based on a union of HMM and maxent alignments with corpus-selective count pruning. The lexicalized distortion model (Al-Onaizan and Papineni, 2006) was used with a window width of up to 5 and a maximum number of 2 skipped (not covered) words during decoding. The distortion model assigns a probability to a particular word to be observed with a specific jump. The decoder uses a 5-gram interpolated language model spanning the various domains mentioned above, except in the constrained mode where a single 5-gram language model was trained on the WMT2010 training data (WMT Website, 2010).

4.4 Evaluation Sets

The methods were evaluated in contrastive experiments utilizing the following (single-reference) test sets:

- *News*: 166 sentences (8700 words) from the news domain.
- *TechA:* 600 sentences from a computer-related technical domain, this has been used as a dev set.
- *TechB:* 1038 sentences from a similar domain as *TechA* used as a blind test.
- *Dev09:* 1026 sentences defined as the *news-dev2009b* development set of the Workshop on Statistical Machine Translation (WMT) 2009 (WMT Website, 2009). Results of others on this set can be found, for example, in (Popovic et al., 2009).
- *WMT10:* 2034 sentences from news domain used as the eval set in the WMT 2010. Results of others on this test set can be found in (WMT Website, 2010).

4.5 Evaluation Metrics

The translation quality in our experiments is evaluated using BLEU (Papineni et al., 2002), as well as using human assessment. The latter is carried out by a judge rating the quality of three translations for each source sentence. The defined assessment levels correspond to

following judgements: 0=Exceptionally poor, 1=Poor (difficult to understand the meaning), 2=Not good enough (errors in grammar, vocabulary, and style make understanding difficult), 3=Good enough (there are errors, however one can understand the meaning with a reasonable confidence), 4=Very good (there may be minor errors, but one can understand the meaning with high confidence), 5=Excellent (the information is presented clearly and with appropriate grammar, vocabulary, and style.) When performing the assessment, the reviewer is presented the source and the competing (blind) translations with their order randomized. The ratings are averaged within the test resulting in a single human score per system. We employed a single judge, who was not involved in the related technical work, and who is proficient in both English and German.

4.6 Results

The cased BLEU scores for English→German are shown in Table 1, and for German→English in Table 2. Overall, we make the following observations: (1) the scores behave consistently between the constrained and unconstrained training, (2) all reordering methods in both directions improve the BLEU scores over their unreordered baselines, (3) the parse- and TSP-based methods seem to fare comparably across the testsets with BLEU differences less than 0.3 points, (4) while the manual rules for German→English do improve the translation quality, the automatic TSP reordering outperforms the manual rules by more than 1 BLEU point leading to overall improvements of about 2 points in the unconstrained condition. Tables 3 and 4 show

English→German Test	Baseline (no RO)	Parse-Based RO Cased BLEU	TSP RO	Diff (TSP vs. Bsl.)
TechA	0.170	**0.184**	0.181	+0.019
TechB	0.190	**0.201**	0.199	+0.018
News	0.248	0.253	**0.255**	+0.023
Dev09	0.142	0.144	**0.146**	+0.018
WMT10	0.150	0.156	**0.158**	+0.012
Dev09 (Constrained)	0.137	0.138	**0.140**	+0.003
WMT10 (Constrained)	0.148	0.154	**0.155**	+0.007

Table 1: BLEU scores for English→German phrase-based machine translation with and without reordering (RO). Last column shows score differences between TSP and Baseline.

German→English Test	Baseline (no RO)	Manual RO Cased BLEU	TSP RO	Diff (TSP vs. Bsl.)
TechA	0.297	0.303	**0.316**	+0.019
TechB	0.294	0.298	**0.312**	+0.018
News	0.250	0.262	**0.273**	+0.023
Dev09	0.184	0.194	**0.202**	+0.018
WMT10	0.194	0.197	**0.206**	+0.012
Dev09 (Constrained)	0.180	0.186	**0.190**	+0.010
WMT10 (Constrained)	0.186	0.191	**0.196**	+0.010

Table 2: BLEU scores for English→German phrase-based machine translation with and without reordering (RO). Last column shows score differences between TSP and Baseline.

the human assessments for the two directions, English→German and German→English, respectively. In both cases 50 randomly sampled sentences from the *TechB* testset were used.

English→German Assessment	Average	Counts per rating grade					
		0	1	2	3	4	5
Bsl (no RO)	2.4	0	15	13	12	8	2
Parse RO	2.4	0	14	17	9	5	5
TSP RO	2.4	0	13	16	12	8	1

Table 3: Human assessment results of English→German MT output for systems with and without reordering (RO) based on 50 randomly sampled sentences from the TechB testset.

German→English Assessment	Average	Counts per rating grade					
		0	1	2	3	4	5
Bsl (no RO)	2.3	1	12	22	8	1	6
Manual RO	3.0	0	11	11	10	5	13
TSP RO	3.4	0	4	10	10	14	12

Table 4: Human assessment results of German→English MT output for systems with and without reordering (RO) based on 50 randomly sampled sentences from the TechB testset.

4.6.1 Discussion

In the German→English direction the results let us conclude that the TSP method leads to improvements in the translation quality. A post-review analysis of the human assessment indicates that most gains can be linked to either word order or better phrase match. The latter is a consequence of more monotone alignments and thus an improved phrase extraction. We have investigated the latter point by recording the count of phrase pairs extracted from the same training corpus (WMT2010) with and without reordering. While in the unreordered system, the extraction resulted in 30.0M phrase pairs, in the reordered system this number increased to 36.3M phrase pairs - corresponding to about 20% increase in extraction efficiency. The reordered phrases also tend to be slightly longer, on average. An example from the human assessment illustrates the nature of the improvement:

- Source: *Die Vergabekriterien sind für alle Kategorien klar definiert, wie auch ihre Beurteilung.*
- Baseline: *The award for all categories clearly defined, and their assessment.* (Rating: 2)
- TSP-RO: *The lending criteria are clearly defined for all categories, as well as their assessment.* (Rating: 5)

The baseline output seems to suffer not only from incorrect word order but also from dropping essential words (“criteria” and “are”), both of which are rectified in the reordered output.

We observe a somewhat different picture for the English→German SMT with results mixed between the automatic and the human metrics. The improvements in BLEU fail to produce noticable difference for the human judge. This could be caused by several factors. First, the overall translation quality is low (2.4), therefore word order improvements may not help in presence of dominant errors. An example from the assessment sheet illustrates this issue:

- Source: *The fix is to relax these attributes as optional.*
- Baseline: *Der Fix ist sich zu entspannen diese Attribute optional sein soll.* (Rating: 1)
- TSP-RO: *Der Fix ist diese Attribute als optionale zu lockern.* (Rating: 1)

Clearly, the word choice of the baseline ("sich zu entspannen" - "to rest" or "to unwind") is poor as is its placement. The reordered system has undoubtedly better word order and arguably better word choice, however, the judge still considered the overall translation as "poor."

Second, correctly placing the verbs in the English→German direction may be a harder problem. Elaborating on this conjecture, we may first consider the relevant word patterns in the two language directions: while going into English the reordering moves (mostly) verbs together (e.g., with the auxiliary verb being an anchor), it moves them far apart when going into German. Given that these verbs (e.g., auxiliary and their main verb) tend to form a single semantic unit of the sentence, they will thus be captured by the phrase extraction as a unit. Conversely, when splitting the verb groups and moving them apart, the positive effect of a more monotone word alignment may be out-weighed by the fact that these verbs will not be captured in one unit. Moreover, we hypothesize an accurate verb placement to be a harder problem going into German than into English. In the latter case an unambiguous placement pattern is typical, e.g., a main verb being placed immediately behind its auxiliary. On the other hand, the placement of a main verb to resemble German depends heavily on the parser accuracy (clause boundaries), and it can sometimes be ambiguous (e.g., verbs may or may not move past embedded clauses). One way to mitigate this problem might be combining reordered and unreordered phrase pairs in the training and then rely on word reordering lattices being used inside the decoder to choose the best matching alternative (as proposed in (Rottmann and Vogel, 2007)), which, however, increases the search complexity of the SMT system.

4.7 Future Directions

We described two methods for syntactic source reordering developed for English-German SMT. Our results indicate that both methods, applied as a pre-processing step, lead to improvements, as measure by BLEU. Significant gains in human assessment of the translation quality are observed for German→English, however, no observable changes were achieved for English→German according to the judge. The latter finding poses several questions: (1) why does same reordering method help significantly more in one direction than vice versa? and (2) how can the discrepancy between our automated metric BLEU and the human assessment results be mitigated? Regarding (1) we hypothesized the reordering problem to be harder when going into German due to a higher word "scatter" as well as sensitivity to parser and tagger accuracy. We plan to further investigate this problem and to refine both methods to to increase their robustness. Independently, the challenge of the overall low quality of the English→German remains to be addressed. Improving the baseline could help the reordering gains in BLEU to show up also in the human assessment, as a better word order will arguably tend to make more impact going from a rating of 3 to a rating of 4, or 5, than, say, from 1 to 2. English→German translation is a challenging language pair and we believe significant improvements will be achieved only by addressing several problems beyond just word ordering. These include generating correct inflections, grammatical agreement, and preventing content word loss. Our results also confirm human assessment to be a necessary component of the overall system evaluation helping to avoid over-optimistic conclusions.

References

Al-Onaizan, Y. and Papineni, K. (2006). Distortion models for statistical machine translation. In *Proceedings of ACL*.

Brown, P. F., Pietra, V. J., Pietra, S. A. D., and Mercer, R. L. (1993). The mathematics of statistical machine translation: Parameter estimation. *Computational Linguistics*, 19:263–311.

Carl, M. (2007). Metis-ii. the german to english mt system. In *In Proceedings of the 11th Machine Translation Summit*, pages 65–72.

Chiang, D. (2007). Hierarchical phrase-based translation. *Comput. Linguist.*, 33(2):201–228.

Collins, M., Koehn, P., and Kučerová, I. (2005). Clause restructuring for statistical machine translation. In *Proceedings of ACL*.

Crego, J. and Habash, N. (2008). Using shallow syntax information to improve word alignment and reordering for smt. *Proceedings of the Third Workshop on Statistical Machine Translation*, pages 53–61.

DeNero, J. and Uszkoreit, J. (2011). Inducing sentence structure from parallel corpora for reordering. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '11, pages 193–203, Stroudsburg, PA, USA. Association for Computational Linguistics.

Galley, M., Graehl, J., Knight, K., Marcu, D., DeNeefe, S., Wang, W., and Thayer, I. (2006). Scalable inference and training of context-rich syntactic translation models. In *Proceedings of ACL*.

Huang, F. (2009). Confidence measure for word alignment. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 932–940, Suntec, Singapore. Association for Computational Linguistics.

Ittycheriah, A. and Roukos, S. (2005). A maximum entropy word aligner for arabic-english machine translation. In *Proceedings of HLT/EMNLP*.

Khalilov, M. and Sima'an, K. (2011). Context-sensitive syntactic source-reordering by statistical transduction. In *Proceedings of 5th International Joint Conference on Natural Language Processing*, pages 38–46, Chiang Mai, Thailand. Asian Federation of Natural Language Processing.

Koehn, P. (2005). Europarl: A Parallel Corpus for Statistical Machine Translation. In *Conference Proceedings: the tenth Machine Translation Summit*, pages 79–86, Phuket, Thailand. AAMT, AAMT.

Liu, Y., Liu, Q., and Lin, S. (2006). Tree-to-String alignment template for statistical machine translation. In *Proceedings of ACL*.

Niehues, J. and Kolss, M. (2009). A pos-based model for long-range reorderings in smt. In *In Proc. of Fourth ACL Workshop on Statistical Machine Translation*.

Niessen, S. and Ney, H. (2001). Morpho-syntactic analysis for reordering in statistical machine translation. In *Proc. MT Summit VIII*.

Papineni, K., Roukos, S., Ward, T., and Zhu, W.-J. (2002). Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, ACL '02, pages 311–318, Stroudsburg, PA, USA. Association for Computational Linguistics.

Popovic, M., Vilar, D., Stein, D., Matusov, E., and Ney, H. (2009). The RWTH machine translation system for WMT 2009. In *Proceedings of WMT 2009*.

Ratnaparkhi, A. (1999). Learning to parse natural language with maximum entropy models. *Machine Learning*, 34(1-3).

Ringger, E., Gamon, M., Moore, R. C., Rojas, D., Smets, M., and Corston-Oliver, S. (2004). Linguistically informed statistical models of constituent structure for ordering in sentence realization. In *Proceedings of the 20th international conference on Computational Linguistics*, COLING '04, Stroudsburg, PA, USA. Association for Computational Linguistics.

Rottmann, K. and Vogel, S. (2007). Word reordering in statistical machine translation with a pos-based distortion model. In *Proc. of the 11th Internat. Conf. on Theoretical and Methodological Issues in Machine Translation*, Sweden.

Schiller, A., Teufel, S., and Thielen, C. (1995). Guidelines für das tagging deutscher textcorpora mit stts. Technical report, IMS Stuttgart/Seminar f. Sprachwiss. Tübingen.

Tromble, R. and Eisner, J. (2009). Learning linear ordering problems for better translation. In *Proceedings of EMNLP*.

Visweswariah, K., Navratil, J., Sorensen, J., Chenthamarakshan, V., and Kambhatla, N. (2010). Syntax based reordering with automatically derived rules for improved statistical machine translation. In *COLING*, pages 1119–1127.

Visweswariah, K., Rajkumar, R., Gandhe, A., Ramanathan, A., and Navratil, J. (2011). A word reordering model for improved machine translation. In *EMNLP*, pages 486–496.

WMT Website (2009). http://statmt.org/wmt09/.

WMT Website (2010). http://statmt.org/wmt10/.

Xia, F. and McCord, M. (2004). Improving a statistical mt system with automatically learned rewrite patterns. In *Proceedings of Coling*.

Yamada, K. and Knight, K. (2002). A decoder for syntax-based statistical MT. In *Proceedings of ACL*.

Zhang, Y., Zens, R., and Ney, H. (2007). Chunk-level reordering of source language sentences with automatically learned rules for statistical machine translation. In *NAACL-HLT AMTA Workshop on Syntax and Structure in Statistical Translation*.

Zollmann, A. and Venugopal, A. (2006). Syntax augmented machine translation via chart parsing. In *Proceedings on the Workshop on Statistical Machine Translation*.

Zwarts, S. and Dras, M. (2007). Syntax-based word reordering in phrase-based statistical machine translation: Why does it work? In *Proc. MT Summit*.

Grounded Language Acquisition: A Minimal Commitment Approach

Sushobhan NAYAK[1,2] *Amitabha MUKERJEE*[2]
(1) Stanford University, CA, USA
(2) Indian Institute of Technology Kanpur, India
nayaks@stanford.edu, amit@iitk.ac.in

ABSTRACT
We take up the challenge of learning a grounded model of language when our agent has a body of machine learning algorithms and no prior knowledge of either the physical domain or language, in the sense of "least commitment". Based on a 2D video and co-occurring raw text, we demonstrate how this cognitively inspired model segments the world to obtain a meaning space, and combines words into hierarchical patterns for a linguistic pattern space. By associating these two spaces under temporal co-occurrence constraints, we demonstrate the acquisition of term-meaning pairs for names, actions and relations. We next map physical arguments for actions and relations to syntactical constructions resembling a cognitive grammar framework. Thus the system is able to bootstrap a rudimentary lexicon and syntax. While experiments are primarily in English, we present partial results for Hindi obtained without any change in the methods, to indicate its potential application to other languages.

KEYWORDS: Cognitive grammar, image schema, ADIOS.

Proceedings of COLING 2012: Technical Papers, pages 2059–2076,
COLING 2012, Mumbai, December 2012.

1 Language learning: The minimal-commitment approach

We investigate a minimal-commitment approach to learn both a grounded lexicon and some rudimentary grounded syntax of an unknown language. By *minimal commitment*, we wish to restrict the prior knowledge available to our learning agent to a minimal set of abilities, and almost no resources or models of the language or domain. By *grounded lexicon* we would like to learn a bipolar relation between a unit of language and a perceptual pattern, and *grounded syntax* refers to a similar mapping from syntactic patterns to relations or events in the perceptual space (Langacker, 1987). Further, given the minimal prior knowledge formalism, the set of perceptual schemas that constitute models for meaning are obtained from the visual input in an unsupervised manner. The input to the work consists of visual sequences with simple shapes, and a set of narratives of this situation generated by adult subjects. We focus on the relation of *containment* (A in B), and the event of *chase*, and show how constructions corresponding to these are learned.

There are many works dealing with the grounded learning of words (Roy and Reiter, 2005; Siskind, 1994; Steels, 2003; Regier, 1996). Our input for word learning however, is significantly more challenging, since the narrative is a set of sentences from an unconstrained narrative, and the lexical and syntactic choices as well as the referential intentions of the speakers vary considerably. This problem is partly resolved by enabling the agent with a bottom-up model of dynamic attention, which has been shown to help computational simulations of word learning (Yu and Ballard, 2007; Mukerjee and Sarkar, 2007).

A more significant difference with earlier work is that target of the reference (a set of "concepts") is not given, but has to be discovered. A few approaches (Regier, 1996; Roy and Reiter, 2005) do discover some aspects of the semantics, but the structure is given. Thus, all approaches to grounding permit the agent to have some knowledge of the task domain in order to constrain the structures for the conceptual space; some provide a set of predicates outright (Siskind, 1994; Bergen et al., 2004; Dominey and Boucher, 2005; Caza and Knott, 2012). Let us illustrate the difficulty of making no semantic commitment with the example for containment. Without the convenience of a pre-defined predicate, the relation of being contained (the target for "in") cannot be known *a priori* but has to be first discovered as a distinct cluster in some sensory space. However, we demonstrate that such clusters emerge at least for some of the concepts of interest. In this work we simply use mean-shift clustering, but in other situations we have found that the presence of intrinsic goals can substantially improve the discrimination (e.g. for the containment task, the agent may have an intrinsic goal of inserting an object into an orifice such as a mouth).

Since the perceptual discovery operates independent of language, we assume that a set of such characterizations are already available at the start of the linguistic association process. The availability of such proto-concepts also has strong cognitive plausibility; infants are able to discriminate situations from 3 months onwards, and by 9 months, it is this ability to cluster data into groupings that lead human infants to discover the phonemic structure of their language (Mandler, 1992). One of the observations of this work is that this pre-linguistic proto-semantic discovery enables a set of categories that are specific to the domain and the goals of the agent, and eventually lead to a set of predicates that are more relevant to the situation and hence more likely to appear in linguistic discourse. Thus, this unsupervised discovery process forms the scaffolding on which the bootstrapping process works.

Attempts to learn grammatical structure range from attempts that ignore semantics altogether

to richly grounded models. The purely syntactic forms have been quite successful in inducing probabilistic grammars for tasks such as machine translation (Marino et al., 2006), and analysis of such as n-grams and path alignment have been used to determine grammars from single language corpora as well (Solan et al., 2002). In our work, we use the approach based on simple n-grams, as well as the more sophisticated (Solan et al., 2002) model, to identify the candidate syntactic structures that will be associated with the proto-semantic structures to discover constructions.

Grammar inductions that model the semantics are given a semantic structure which is matched to user narratives, obtained for instance while performing a task. The visual inputs are analyzed using a vision system into the actions identified, and these are then used to induce some aspects of grammar. This is used to learn some grammatically distinct structures (such as active or passive voice, or prepositional terms) in (Dominey and Boucher, 2005). Another body of work considers formal logical description of scenes, and induces probabilistic grammars by unambiguous sentence pairing. An impressive gain in this area has been to reduce the commitment to language knowledge, so that grammatical structures of questions in languages as different as Turkish and Japanese can be learned (Kwiatkowski et al., 2010). However, the scalability of a process that requires a large numbers of predicate specific training sets limits the scalability of the method. The objective of minimizing commitment to prior language models is essentially aimed at modeling the ability to acquire any ambient grammar.

The present work differs from all these in minimizing the dependence on prior knowledge of either the perceptual space or the language being learned. Both the perceptual structures as well as syntactic structures are obtained using unsupervised techniques, and the association performed thereafter. One important observation is that the semantics helps narrow the corpus to those sentences uttered while a specific concept is in focus, thus helping acquire structures related to it.

1.1 Capabilities of the learning agent

Start Frame	End Frame	Subject 1	Subject 2
617	635	the little square hit the big square	they're hitting each other
805	848	the big square hit the little square	and they keep hitting each other
1145	1202	the big square goes inside the box; (and) the door closes	another square went inside the big square

Table 1: Sample descriptions of events. Note the differing referential and lexical choices.

We may now define the capabilities of our minimal commitment language learning agent. We assume the agent has a) a wide range of machine learning algorithms, b) some awareness of the mental state of other agent (*Theory of Mind (Mukerjee and Sarkar, 2007)*), c) task-independent (bottom-up) dynamic perceptual attention, d) a mechanism for fixing goals (intrinsic motivation). In addition we also assume the agent has the ability to segment words from the linguistic inputs. We note that possibly such an ability may have been based on earlier exposure to the target language.

The input for our agent is a video sequence (based on Heider/Simmel (Heider and Simmel,

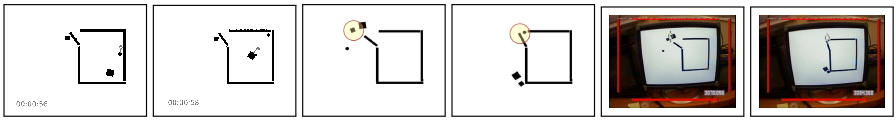


Figure 1: *Perceptual input: 2D video based on Heider/Simmel. Also showing dynamic attention model*. Three rigidly translating shapes, a big-square (**[BS]**), a small-square (**[SS]**) and a circle (**[C]**) interact playfully (velocities shown with arrows). Part of the container, a door (**[D]**) opens or closes at times. Figures 3 and 4 show the synthetic gaze computation, and Figs 5 & 6, actual gaze data for a viewer, showing reasonable correlation to predicted gaze.

1944), Fig. 1)[1]. The English linguistic database consists of 40 commentaries, collected from subjects who were simply asked to "describe the video", while they differ in certain respects, the speakers were not constrained in any form in terms of lexical choice, focus, or other aspects of their narratives. A group of 13 was collected as part of (Hard and Tversky, 2003) were asked to comment on a fine-grained vs coarse-grained temporal segmentation of the video. The other 27 narratives were collected by us. All narrators were students in the 20-25 age-group. The English narratives constitute a corpus of 4200 words (700+ sentences), and exhibit a wide range of linguistic variation both in focus (perspective) and on lexical and construction choice (see Table 1). Both the visual and the linguistic input are unlabelled.

It is possible that better results may be possible with a stronger social emphasis than was maintained in this work (e.g. joint attention vs individual attention). However, the requirement that an actual human be present makes it more difficult to scale up. As it exists, the system can possibly be tuned for a number of domains and languages.

2 Language acquisition as association

The problem of language acquisition with minimal commitment has two phases. In the pre-associative phase, the problem is to identify the semantic and syntactic primitives independently.

- *Perceptual structure discovery*. Given a perceptual space W, discover the set of structures Γ in this space, possibly focusing on high-frequency situations, or those that are relevant to its goals. Some patterns $\gamma \in \Gamma$ are simple (e.g. a shape that moves rigidly), versus others that are encode relations over space or time. Some of the simpler patterns participate in the more complex interactions.
- *Linguistic structure discovery*. Given a linguistic space L, the system is exposed to a set of sentences, each a sequence of words (w). It attempts to identify sequential patterns Λ (possibly hierarchical) that, would enable a more compact description of the input.

Strictly speaking the strong independence of the perceptual and linguistic spaces is required only for a small set of initial phrase-meaning mappings; subsequently, mappings that are known to be present in the situation can substantially constrain other possibilities, leading to great efficiencies in acquisition (Yu, 2008; Bloom, 2000) (the vocabulary spurt). In this work, we are focused on the very first, bootstrapping steps, so we maintain a strict independence. We note however, a small aspect of this independence. We observe that both linguistic and conceptual

[1]This video was developed and some of the narratives collected by (Hard and Tversky, 2003). We are grateful to Barbara Tversky for permission to use this data in this different enterprise.

structures are hierarchically organized. Thus, the a relation *A contains B* is defined over objects A and B, which are themselves structures on their own right. Thus, it is reasonable that A and B would be brought to awareness (reified) before the relation of containment. Hence, if the names of two participating objects are known, then we do use this knowledge to constrain phrases that may describe a relation between them.

We note that each sentence in the input is uttered over a particular temporal window, and includes a set of linguistic patterns, $\lambda_1, ...\lambda_m$. If the perceptual patterns observed during this utterance interval are $\gamma_1, ...\gamma_n$, then one may expect some of these λ to be mapped to some of the γ. Over many narratives describing similar situations, we are able to access a large set of such association candidates; and the association process merely posits some of the strongest co-occurrences in this large set. This association task may now be defined: Map fragments of language to the patterns of the perceptual data. Discover mappings from the meaning space to the syntax space, i.e $\gamma \in \Gamma \mapsto \lambda \in \Lambda$, where γ, λ are the perceptual and linguistic patterns that co-occur during a sentence-utterance duration.

For each pattern γ or λ, the system should be able to determine if a given perceptual or linguistic situation is an instance of the particular pattern or not. Since the initial discovery of these patterns are based on unsupervised methods like clustering, binary discriminations can be performed by bayesian techniques on the two distributions; and new instances assigned to a suitable class, or a new cluster may be initiated. In general, this means that each distribution may change significantly as more experience accumulates, but in this work, we shall not be expanding our repertoire of concepts so we shall not encounter this situation.

We note that what we are learning on the linguistic side is far from what is normally understood by syntax. We are learning merely a map from the sentential space S to a pattern space Γ, which is chosen so as to induce the largest structures that do not cause contradictions. As we shall see, these structures will use hierarchies which look rather like syntactic categories, but are quite different from traditional parts of speech or other treatments. However, we note that there can be many grammars that explain a given set of sentences, and the grammar that describes this particularly input may differ substantially from human-crafted grammars.

We observe that the grounding - i.e. the availability of a mapping to a space outside the set of logical tokens - is a very crucial part of the process by which the initially learned mappings are expanded on in further usage. Without it we would not be able to constrain the linguistic parts that are relevant when a known mapping arises(Langacker, 1987; Bergen et al., 2004).

In the first pre-associative stage, we need to define the perceptual structures. In the video (Fig. 1), the referent objects – a big square, a small square and a circle (from now on referred to as **[BS]**, **[SS]** and **[C]** respectively) – are a set of rigid translating pixels, and are easily segmented. We note here that while we claim a low degree of domain dependence, had the video been 3D or the agents been humans, it would have imposed considerable difficulties. Thus, to an extent, the perceptual analysis is limited to this kind of domain, but nonetheless, there is nothing specific to the scene. The segmented objects constitute the first level of the perceptual abstraction, and despite the variation in the names for these objects in the narrative, associating these is much simpler compared to the multi-object relations or actions.

3 Association measures

The learning objective is to create mappings between the meaning space Γ and the sentential space Λ. Suppose $\Gamma = \cup_i \gamma_i$ and $\Lambda = \cup_i \lambda_i$ be the random variables denoting the said spaces.

Each γ_i denotes a particular hypothesis for a realization of the meaning space. For example, in a verb meaning space created through clustering of motion features, γ_i would denote a cluster. Similarly, for a reference resolution task containing objects o_i, $\gamma_i = o_i$. The λ_i's similarly define partitions of the sentential space, which, based on context, might denote monograms, bigrams or other syntactical structures derived from the purely linguistic input.

We follow a statistical approach to map the two domains to each other (Fazly et al., 2010)[2] instead of an inductive approach(Siskind, 1996), since, not only ours in an unconstrained commentary, but knowing even the position of the referring expressions would be unhelpful because of noise introduced in any single isolated evidence. Though they do not use unconstrained linguistic input, there is much research on cross-situational associations between words and their meanings(Frank, 2010; Roy and Pentland, 2000; Smith and Yu, 2008; Yu et al., 2005; Yu and Ballard, 2007). We employ two association measures to correlate the meaning and sentential spaces. The *relative association*, a Bayesian metric, is defined as

$$P(\gamma_j|\lambda_i) = \frac{P(\lambda_i|\gamma_j)P(\gamma_j)}{P(\lambda_i)} \propto \frac{P(\lambda_i|\gamma_j)}{P(\lambda_i)} = A^{rel}_{ij}.$$

While working well for frequent linguistic elements, the metric is prone to give erroneous results for rare occurrences. For instance, it gives a maximum value of 1 to the correlation between a word w, which has been uttered only once in the whole discourse, and the meaning it has co-occurred with. We, consequently, also employ an information theoretic measure, the *mutual association*, defined as

$$A^{mut}_{ij} = P(\lambda_i, \gamma_j) \log \frac{P(\lambda_i, \gamma_j)}{P(\lambda_i)P(\gamma_j)}$$

because it's the contribution of each (λ_i, γ_j) pair in the mutual information of Γ and Λ

$$I(\Gamma, \Lambda) = \sum_i \sum_j P(\lambda_i, \gamma_j) \log \frac{P(\lambda_i, \gamma_j)}{P(\lambda_i)P(\gamma_j)}$$

It might also be noted that while A^{rel}_{ij} was inadequate for low frequency words, A^{mut}_{ij} gives unusually high scores for highly frequent words like *the* which have only syntactic relevance, due to a high $P(\lambda_i, \gamma_j)$, thus supporting the use of both measures for the investigation.

The goal of this work, however, is not to discover which association measure works best for word learning. The idea we are trying to support is that all categories – nouns, verbs, relational prepositions etc. – can be mapped to their corresponding meaning space through simple associations. Many association measures have been proposed in literature, which work differently in different situations. It seems cognitively implausible that an infant uses only one kind of association to discover the linguistic element with the highest correlation with the meaning and learns that as the label. Neither are many association measures too different in their output, except for a few artefacts. We instead propose that instead of looking for a *perfect* association metric to discover a single label, a more plausible approach would be to discover a *label set*, which would be refined through further syntactical, perceptual or social evidence. Since Bayesian and information-theoretic associations are well-known in NLP, we take the above

[2]However, while (Fazly et al., 2010) use an artificial meaning space for word-meaning correlation, with the meaning space created from sentential input only and essentially treated as a given, we form the meaning space from perceptual input.

[BS]			[SS]			[C]		
word(s)	A_{ij}^{rel}	A_{ij}^{mut}	word(s)	A_{ij}^{rel}	A_{ij}^{mut}	word(s)	A_{ij}^{rel}	A_{ij}^{mut}
square	0.70	1.41	little	0.66	0.79	circle	0.79	2.11
big	0.89	1.11	small	0.72	0.63	square	0.41	1.54
box	0.69	0.78	square	0.46	1.12	little	0.68	1.22
the big	0.87	0.71	small square	0.93	0.53	the little	0.71	0.81
big square	0.94	0.75	little square	0.89	0.46	little circle	0.91	0.60
large square	0.86	0.15	the little	0.70	0.54	the big	0.48	0.61

Table 2: *Noun label learning*: Word associations for the referent objects in attentional focus.

two as representative. As we shall see, both work in most situations. Note that we make no assumptions on the category of words or distinguish them as nouns/verbs. However, different machine learning algorithms are required for learning perceptual objects, events, and spatial relations; thus these are distinguished in the semantic space.

3.1 Noun reference resolution

Noun learning is well known to be easier than verbs(Fleischman and Roy, 2005). In our case, the rigid shapes are easily segmented and tracked, and the mapping of object labels to their visual representation is achieved with the help of a bottom-up dynamic saliency model. Many computational approaches have been proposed for grounded word learning(Iida et al., 2011; Prasov and Chai, 2008), though instead of working on unconstrained utterances, the referring phrases are a given. (Fang et al., 2009) uses static referents in game-like contexts, as opposed to our dynamic referents. We observe that unlike many of these approaches, our learning agent is exposed to complex discourse in which phrases are embedded, and not by isolated one-word labels.

We use visual attention to constrain the region of visual interest and identify the constituents participating in an utterance. In fact, past works like (Prasov and Chai, 2008; Iida et al., 2011) have used gaze cues from speakers to conduct reference resolution. In our case, however, since the learner is presented with only the visual stream and is not in the presence of the speaker, attention is mediated by visual saliency alone, and not by cues received from the speaker's gaze. Therefore, to simulate gaze-based visual attention, we follow the assumption of Perceptual Theory of Mind(Mukerjee and Sarkar, 2007), that the salient features that our cognitive agent discovers through image processing, would also be salient for the speaker involved in the associated commentaries, letting us correlate the visual and linguistic elements coherently. (Mukerjee and Sarkar, 2007) uses a bottom-up visual attention model to predict the gaze, the results of which are shown in Figure 1. This works as our eye-gaze model for the perceptual input. The salient agents being attended to constitute the meaning space Γ, with γ_i = object feature set. For example, the hypothesis denoting **[SS]** might represent γ_{SS} = [color:`black`, size: `25 × 25 px.`, shape: `square`, orientation: `NIL`].[3] Notice that this schema varies according to the number of features the agent is capable of deriving. The object in visual salience is then correlated with utterances that have temporal overlap with the object in focus. Since we do not assume any syntactic information at this point, every linguistic element

[3]Notice that shape: `square` is a high level concept. At present, the model can only determine through image processing techniques that length = breadth and **[BS]** and **[SS]** are of similar shape.

C1 (Come-Close)		C2 (Move-Away)		C3 (Chase)		C4 (Chase)	
move	0.033	chase	0.066	chase	0.479	chase	0.371
toward	0.028	away	0.025	try	0.115	try	0.106
corner	0.023	move	0.022	start	0.093	run	0.050

Table 3: Associating language labels to action clusters from the unsupervised algorithm

is a possible label for the objects. The association between mono- and bi-grams with the objects, for both the association measures, are shown in Table 2[4].

3.2 Verb acquisition

Once the objects are discovered, the next step is to derive perceptual relations for their interaction with each other and the surrounding. We next turn to modeling the mutual interaction between moving agents (for our input, [BS], [SS], [C]). We assume that our artificial agent is capable of employing basic unsupervised machine learning on image data, particularly the ability to segment a picture/frame, generate spatial features and cluster them into separate classes, to achieve the above mentioned goals.

Attempts to learn verbs have involved neurally inspired models of contact actions (x-schemas, (Bailey, 1997)); or a set of actions and their visual parses (Siskind, 1994; Dominey, 2005). As mentioned earlier, our perceptual schemata are discovered based on unsupervised clustering in the perception space. In this case, our perceptual extraction process depends on a specific feature that is not discovered, but these are fairly general and involve the product and difference of relative position and velocity. In earlier work (Satish and Mukerjee, 2008), we show that temporal data mining on the sequence of these feature vectors, based on Merge Neural Gas (Strickert and Hammer, 2005), yields four action clusters, two of which correspond to [come-closer] and [move-away], and two correspond to [chase].

These clusters constitute the hypothesis space for verb acquisition. These are next related to the linguistic input. For this, those sentences, which overlap temporally with the period when the action clusters are active, are taken into account, using an approach similar to (Roy and Reiter, 2005). At this point, it is assumed that the learner knows the nouns (discovered in Section 3.1), which are not considered as labels for verbs. Extremely frequent words (e.g. *the*, *an* etc.) are also dropped from consideration for mapping to actions. The strongest associations for the action clusters are shown in Table 3, with Clusters 3 and 4 (**[chase]**) having a strong association with the word *chase*.

3.3 Perceptual schema for containment

In spatial reasoning, there have been several attempts at defining spatial relations involving continuum measures defined over different geometric features on object pairs. Regier(Regier, 1996), a seminal work in preposition grounding, uses angle measures and a connectionist

[4]Notice that one word association alone might not provide sufficient information since some objects might be referred to through phrases, the validity of which is considered in Section 4.1. Also, to compare both the measures side by side, the mutual association has been scaled appropriately. The most frequent monogram *the* has been ignored in these results, which has the highest A_{ij}^{m} in all the four cases. It would later be eliminated from the probable label set anyway as it provides no information due to its occurrence in all the four label sets.

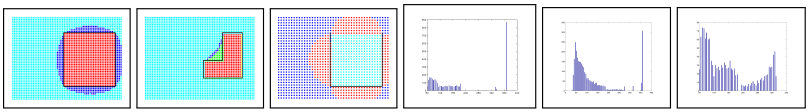


Figure 2: *Clustering through spatial features*: Figs 1-3 represent Visual Angle feature clusters. The inside of all the containers has been clearly identified as a separate cluster only in the latter case. Figs 4-6 are the visual angle histograms.

network to correlate videos and prepositions. The work, however, is limited in the sense that Regier uses videos annotated with single words like IN, OUT, THROUGH etc. while we hope to learn these schemas by clustering the untagged video. Also, because his videos are tagged with prepositions, he never has to work to *discover* the preposition; we have to discover these units from the unconstrained unparsed narrative. (Mukerjee and Sarkar, 2007) use the same dataset as ours, but use a measure based on visual proximity - the *Stolen Voronoi Area* - to cluster space using Kohonen SOMs. We initially tried these two approaches and found that in unsupervised clustering tasks (k-means and mean-shift), these earlier models do not work well for distinguishing the inside and outside of irregular (L- or U-shaped) containers. In a supervised scenario they show good results training with sophisticated neural-nets over multiple epochs, but our goal is to try not to use supervision data.

Another feature implicated in place learning in animals is *visual angle*- the angle subtended by a landmark on the retinal image. We attempted to improve on the previous features by using a single feature – the total angle subtended by a landmark at the object position. With this measure, we find that when the resulting feature space is clustered, one of the clusters works quite well for identifying the IN-schema. Computing this feature involves computing the angle that the landmark, **[box]**, would subtend at each point in the space; the result is measured and clustered using *Mean-Shift*(Fukunaga and Hostetler, 1975), so as to get non-parametric natural clusters. We can see in Fig 2 that one cluster completely covers what may be thought as the inside of **[box]**, whereas the the outside is graded between a number of clusters. If we accept this as a characterization for an image schema for containment, then the distribution of visual angle in this cluster(say the *IN-cluster* or $\mathscr{C}_{in}$) will serve to represent this relation. To test whether this model really represents the *category* of containment relations, we generalize and evaluate it over a number of other shapes. The results of clustering on two novel shapes is shown in Fig 2. We find that regions with varied levels of 'IN-ness' have been separately grouped, validating our choice of features. While for closed convex shapes the measure has a clear demarcation of 'inside'(360° angle), as is evident from the angle histograms in the figure, it gives a more graded assessment for open figures as well, such as the open-top square.

The clusters are the hypothesis space for spatial schemas. The correlation of the prominent IN-cluster($\mathscr{C}_{in}$) with words is shown in Table 4. The sentence space contained all the utterances that occurred when any of the objects in attention was inside the IN-cluster. Of interest to us is also the *change in state*, so that sentences overlapping with the object in attention moving in/out of the IN-cluster are also considered separately(Table 4, results to the right). Words *in/inside/into* are prominent, as are *into,enter* and *out,leave* for transitions in/out of IN-cluster.

4 Linguistic construct acquisition: Rudiments of syntax

At this stage, the agent is aware of some word-meaning mappings; a cognitively plausible incremental approach would suggest that the first glimmers of sentential constructions would

IN	A_{ij}^{rel}	A_{ij}^{mut}	INTO	A_{ij}^{rel}	A_{ij}^{mut}	OUT OF	A_{ij}^{rel}	A_{ij}^{mut}
inside	0.79	11.78	into	0.82	6.98	out	0.65	5.71
into	0.90	9.43	inside	0.53	1.03	leaves	1.00	4.16
in	0.61	4.16	enters	1.00	4.85	exits	1.00	3.46

Table 4: Associating language labels to the prominent IN/containment cluster

form around these recognised words. In fact, children's initial syntactic representations may be centered around individual verbs/relational items, instead of fully abstract grammars(Tomasello, 2003). There has been much work in describing such structures(Mintz, 2003; Saffran et al., 1996). The consensus seems to be that concrete n-grams or patterns ('constructional islands') like 'in the box' emerge first, with these being generalised to abstract syntactic construction like 'in the X' through distributional information in the linguistic input. We, consequently, started our discovery of syntactic structure by analyzing bi- and tri-gram correlations for containment. While the prominent bi-grams were *inside the*, *into the* and *in the*, the tri-grams that emerged were *inside the box*, *in the box* and *into the box*.

Despite some glimmers though, the n-gram approach is not very illuminating regarding the construction encoding for containment. We specifically avoid standard parsers since we are unsupervisedly discovering syntactical structures, not trained word-class labels. We, consequently, follow a richer model of syntactic structure in the spirit of cognitive grammar(Langacker, 1987), ADIOS(Solan et al., 2002), which integrates statistical and classical (generative, rule-based) approaches to syntax. It constructs syntactic representations of a sample of language from unlabeled corpus data unsupervisedly. It first creates a Representational Data Structure(RDS) by morphologically segmenting the input sentences and creating directed edges between vertices corresponding to transitions in the corpus. It then repeatedly scans and modifies the RDS to detect significant patterns through a Pattern Acquisition(PA) algorithm. A pattern tagged as significant is added as a new vertex to the RDS graph, replacing the constituents and edges it subsumes, the process being repeated and bootstrapped. Two representative patterns found through a run of ADIOS through the whole English commentary set are presented below, which provide the first indications of grouping of words in to syntactical classes:

$$1.\ \left[\begin{array}{c} the \rightarrow \left[\begin{array}{c} big \\ large \end{array}\right] \rightarrow square \\ the \rightarrow square \end{array}\right] \rightarrow \left[\begin{array}{c} scares \\ approaches \\ chases \end{array}\right] \rightarrow \left[the \rightarrow \left[\begin{array}{c} small \\ little \end{array}\right]\right]$$

$$2.\ \left[\begin{array}{c} the \rightarrow \left[\begin{array}{c} ball \\ box \\ door \\ square \end{array}\right] \\ circle \\ it \end{array}\right] \rightarrow \left[\begin{array}{c} moved \\ moves \\ runs \end{array}\right]$$

4.1 Syntactic classes refine object labels

From Pattern 2 notice that *circle, square, box, door,it,he* (say Group 1) belong to an equivalent class. Similarly, combination of words like *the big square, the little square* etc. are syntactically similar to Group 1 (Pattern 1 & 2). *open, move* etc. , on the other hand, are syntactically completely different from the aforementioned words. Also notice that *big,little,small* by themselves are not equivalent to Group 1; but as part of a bigger phrase, like *the big square*, they are equivalent to Group 1. Similar is the story with *the*. The noun label-sets from Table 2 are now curtailed to Group 1 words only, due to their high individual and combined associativity. So, while bi- and tri-grams like *small square*, *the square* are retained due to their syntactical equivalence to monograms, utterances like *door closes, circle moves* are treated as argument structures and discarded as labels. Thus, while many of these structures may offend linguists who feel they are overfitted to this particular input, it cannot be denied that it is a working model for characterizing path level agglomerations in the input.

However, in the mean while, we would like to emphasize that the process of deriving syntactic information (this Section) and mutual association of linguistic and perceptual elements (Sec 3) are not mutually exclusive or ordered processes. Even though we have described syntactic information discovery after we have motivated perceptual to linguistic element mappings, we do not assume that they are ordered that way. In fact, being independent events and mutually informative, they might as well run parallelly.

4.2 Verb and relational argument structure

Other important *containment and chase-specific* patterns, however, are hardly discovered due to the presence of myriads of different structures leading to the diffused nature of the dataset. We, consequently, follow the cognitively plausible incremental approach. We isolate those parts of the corpus that co-occur with containment/chase situations in the perceptual input, with a view that unsupervised analysis of this sub-corpus discovers regularities that are more *specific* than can be achieved under the same computational constraints for a broader corpus. The knowledge of linguistic elements *in/into/inside* and *chase* from word-label association helps constrain the corpus to a focused sub-set of 107 sentences for IN(each containing one of those words) and 36 sentences for CHASE, which facilitates discovery of some prominent structures:

1. [Group 1 word/phrase] → [CHASE (*chases/is chasing*)] → [Group 1 word/phrase]
2. [CHASE (*chased*)] → [by] → [the] → [little]
3. [CHASE (*chases*)] → [little] → [Group 1 word/phrase]

1. [Group 1 word/phrase] → [IN] → [the]
2. [Group 1 word/phrase] → [verb] → [IN] → [the]
3. [Group 1 word/phrase] → [verb] → [IN]
4. [Group 1 word/phrase] → [verb] → [other linguistic elements] → [IN]
5. [IN]→[the]→[Group 1 word/phrase]

	CHASE			IN				
Pattern No.	**1**	**2**	**3**	**1**	**2**	**3**	**4**	**5**
Frequency	17	3	2	10	26	7	10	36
'ground-truth argument' frequency	20/34	1/3	1/2	10	18	5	9	27
'ground-truth argument' % age	59	33	50	100	69	71	90	75

Table 5: *Correlating perceptual and linguistic argument structure-CHASE & IN*

In fact, this practice of constraining hypothesis space to facilitate quick learning is supported in literature(Siskind, 1996), albeit in different contexts. Before proceeding though, we would like to emphasize that the claim is not that these are not the only patterns that are possible:[5] Our focus here is to investigate how amongst the set of plausible patterns, some are preferably acquired due to evidence and bias and strong correlation to perceptual domain. The above patterns each include at least one object term(Group 1 term) and one relation/verb term, following the hypothesis that out of patterns emerging from purely statistical data, the patterns that have a previously learned label, might be favorably acquired.

While these structures have been derived from purely linguistic input, their *grounding*(bootstrapping in perception) is possible only if they show remarkable correlation with the perceptual argument structure, which they indeed do, as can be ascertained from Table 5. In the table, 'ground-truth argument' means the perceptual and linguistic agents are in conformation. Conflict cases involve both when the linguistic agent is different from the perceptual agent (e.g. in video [CHASER] is **[BS]**, but in the utterance, [CHASER] is *circle*) and when the linguistic agent is unfamiliar (e.g. in video [CHASER] is **[BS]**, but in the utterance, [CHASER] is *big block – block* is syntactically equivalent to *square* so that the structure is valid, but the agent has not *yet* associated it with any perceptual objects from past evidence). Since Structure 1 has two referents, the total number of referents for 17 sentences is 34. While raw correlation is greater than 50%, if we discount the sentences with unfamiliar linguistic agents, there is 100% correlation between linguistic and perceptual schemas, thereby making the linguistic argument structure concrete.

All the IN patterns show more than ∼70% correlation between the two domains, leading to the grounding of respective structures. Note that the attention based model for learning nouns cannot learn the container/box, which is never dynamically salient. Thus, it's label is unknown. However it is prominent in these containment sentences, and discounting the frequent word *the* in trigrams such as "*{ inside/in/into} the box*", we may associate *box* with **[box]**, treating it as a label for the container. It logically follows since **[box]** is a physical object, and based on the past experience of the agent, should be assigned a linguistic element that syntactically confirms to concepts of other physical objects[6] like **[BS], [SS]** etc. From the above patterns, the syntactical equivalence of *box* is supported by its grouping with *door, square* etc.(see Pattern IN4). This grouping into equivalent classes is the first evidence of word category acquisition. The primary mapping of *box* to **[box]** is further strengthened from Table 5, where *box* has been taken as the 'ground-truth argument' for **[box]** and with this assumption, more than 75% of the IN2 pattern sentences agree in both perceptual and linguistic domain, thereby facilitating the

[5]In fact, from a statistical viewpoint, with change in length of input and variation in ADIOS parameters, myriads of patterns can emerge.

[6]For the present set-up, all the moving objects and the container can be derived through image segmentation, thereby being similar 'physical' objects.

[BS]			[SS]			[C]			[IN]		
word(s)	A_{ij}^{rel}	A_{ij}^{m}	word(s)	A_{ij}^{rel}	A_{ij}^{m}	word(s)	A_{ij}^{rel}	A_{ij}^{m}	word(s)	A_{ij}^{rel}	A_{ij}^{m}
बक्सा baksA/box	.77	.37	बक्सा baksA/box	.62	.44	गोला golA/ball	.83	.54	अन्दर andar/in	.80	1.30
बडा(badA/ big) बक्सा	.85	.18	छोटा(chota/ small) बक्सा	.90	.25	बक्से के(ke/–)	.63	.27	बाहर (bA-har/out)	.78	.73

Table 6: *Noun label learning*: Word associations for the referent objects in attentional focus.

acceptance of the assumption. In fact, of the 9 mismatched referents, only 3 are wrong (*in the door/corner/place*, while the rest are due to synonymous references (*in the room*, *in the square*).

One possible consequence of grounded syntax is a facilitation of acquisition of minor labels, synonyms and anaphoras, which are overwhelmed by other salient labels in a simple association task. While we have not entered into a serious investigation of this, some rudimentary results are apparent nonetheless. *ball* and *room* occur 5 and 6 times respectively in positions A and B in sentential domain in 'A in/inside/into B' and map exclusively to $x =$ **[C]** and $y =$ **[box]** for xINy in perception, so that they are treated as synonyms for *circle* and *box* (in this scenario). *block* has been used for both **[BS]**(75%) and **[SS]**(25%), creating an equivalence with *square*. Similarly, *it* maps to **[BS]**, **[SS]** and **[C]** in 56, 25 and 19 percent of its occurrences, thus showing the first evidence of anaphora acquisition, though we leave a detailed investigation to a future work. How these acquired structures can further be extended to assimilate metaphors and how they are developmentally salient for the concept of containment, has been investigated in much detail elsewhere (Nayak and Mukerjee, 2012).

Potential application to other languages To investigate the potential extension of this approach to other languages, we obtained results for Hindi obtained without any change in the methods. The Hindi database consisted of 10 commentaries from first-language speakers, with more than 200 sentences and 2000 words, describing the same video (e.g. तो लगता है कि यहां एक बडा बक्सा है जिसमें एक चौकोर है [to lagtA hE ki yahAN ek badA baksA hE jismeN ek chokour hE / `It seems that there is a big box here in which there is a square present.`]). Hindi is a much more richly inflected language than English, with abundance of gender and number agreements. Even verbs have several modal affixes in addition to tense and, several postpositional markers for case, which sometimes mark for source and destination as well. The constructions derived from ADIOS, therefore, are diffuse and minimal, owing to the small dataset(200 sentences, compared to 700 in English). So, while a detailed investigation of syntax grounding is far-fetched at present, the emerging patterns show consistencies with their English patterns nonetheless (*comes/runs out of box*):

$$\begin{bmatrix} \text{डब्बे(dabbA/box)} \\ \text{बक्से(bakse/box)} \end{bmatrix} \rightarrow \begin{matrix} \text{के} \\ \text{(ke/–)} \end{matrix} \rightarrow \begin{bmatrix} & \text{बाहर(bAhar/out)} & \\ \text{बाहर} & \begin{bmatrix} \text{आ(aa/come)} \\ \text{भाग(bhAg/run)} \end{bmatrix} & \text{जाता} \\ \text{(bAhar/out)} & & \text{(jAtA/goes)} \end{bmatrix}$$

Furthermore, similar results to English are found for word-to-meaning mappings (See Table 6). Also, we have पीछा(pichA/chase) as the dominant label for the verb clusters, with $(A_{ij}^{rel}, A_{ij}^{mut})$ of $(0.78, 19.79)$ and $(0.64, 13.7)$ respectively for Clusters 3 and 4 from Sec 3.2.

Conclusion and perspectives

The work described is able to learn a limited set of lexical items and grammatical constructions for a small domain. If it would be possible for the system scale up, one may suggest that such a system might be able to learn an increasing number of concepts from a larger number of domains, as it happens with children. Indeed, the holy grail of computational linguistics would be to create such semantically rich models of language, and alternatives like this approach are at the very least worth investigating further. But how difficult would it be to scale up this computational approach to new domains and new concepts?

The reason why minimal commitment is attractive is of course, precisely because it makes it easier to extend the approach to other physical domains or other languages. Situations where other concepts have saliency would make it more likely that associations with linguistic expressions mapping them would be learned. Also, with the capacity for simulation, it is no longer necessary to have direct grounding for learning everything. In a sentence where a novel term or concept is introduced, the meaning of the term, or the concept itself, may be understood by simulating what is known from the other parts of the expression. Indeed this is how humans learn the vast majority of our immense vocabularies (Bloom, 2000).

What the initial term-meaning pairing provides is an index into the space of meanings. The next time a similar expression is encountered, the new semantics are compared with the previous mental model so it can be extended. Thus if our system here, which knows "in the box", now encounters "in the basket" and "in the room", it would gradually be able to generalize the argument of "in the" to the concept of container (and would eventually reject "in the banana", say). Further, as we have demonstrated elsewhere (Nayak and Mukerjee, 2012), once the system encounters increasingly figurative expressions such as "in the team", "in the school", "in the spotlight", "in the doghouse", it would be able to extend this using the analogy mechanism inherent in the sensorimotor schemas used for grounding. We emphasize again the need for sensorimotor grounding without which such sense extensions are clearly not possible.

This work presents a view that takes seriously and implements computationally, the ideas in Cognitive Grammar (CG) (Langacker, 1987), as opposed to traditional grammars. This view is part of the models in (Regier, 1996; Bergen et al., 2004; Chang and Maia, 2001), but ours is the first to propose a model that also learns the semantics, thus freeing it up to learn new structures in new domains. What is really being learned in this process is what is called an image schema in CG and for larger structures, a more elaborate schematization. With the grounded model, as we encounter increasingly complex perceptual schema that map to longer phrases, it should also be possible to discover the processes of composition in CG.

At this point, such claims may seem too remote, but they are not completely implausible and given their potential for changing the way NLP works today, we would argue at least for the widespread development of corpora with image streams along with multiple raw text narratives in many situations and many languages. We make a humble start in this direction by making our image and text corpora in several languages available on the web. Agents that accumulate the learning from several domains may prove (or disprove, or suggest new directions) in the enterprise. The main claim we are making is that this is a novel approach to language induction, and while its potential is far from clear, at least it is worthy of further investigation.

References

Bailey, D. (1997). *A Computational Model of Embodiment in the Acquisition of Action Verbs*. PhD thesis, UC Berkeley, Dept EECS.

Bergen, B., Chang, N., and Narayan, S. (2004). Simulated action in an embodied construction grammar. In *Proc. of the 26th Annual Meeting of the Cognitive Science Society*.

Bloom, P. (2000). *How Children Learn the Meanings of Words*. MIT Press, Cambridge, MA.

Caza, G. and Knott, A. (2012). Pragmatic bootstrapping: a neural network model of vocabulary acquisition. *Language Learning and Development*, 8(2):113–135.

Chang, N. and Maia, T. (2001). Grounded learning of grammatical constructions. In *AAAI Spring Symp. On Learning Grounded Representations*.

Dominey, P. (2005). Emergence of grammatical constructions: Evidence from simulation and grounded agent experiments. *Connection Science*, 17(3-4):289–306.

Dominey, P. F. and Boucher, J.-D. (2005). Learning to talk about events from narrated video in a construction grammar framework. *Artificial Intelligence*, 167(1-2):31–61.

Fang, R., Chai, J., and Ferreira, F. (2009). Between linguistic attention and gaze fixations in multimodal conversational interfaces. In *Proceedings of the 2009 International Conference on Multimodal Interfaces*, pages 143–150. ACM.

Fazly, A., Alishahi, A., and Stevenson, S. (2010). A probabilistic computational model of cross-situational word learning. *Cognitive Scince*, 34(6):1017–1063.

Fleischman, M. and Roy, D. (2005). Why verbs are harder to learn than nouns: Initial insights from a computational model of intention recognition in situated word learning. In *Proceedings of the 27th Annual Meeting of the Cognitive Science Society*.

Frank, M. (2010). *Early word learning through communicative inference*. PhD thesis, Brain and Cognitive Sciences, Massachusetts Institute of Technology.

Fukunaga, K. and Hostetler, L. (1975). The estimation of the gradient of a density function, with applications in pattern recognition. *IT, IEEE Transactions on*, 21(1):32 – 40.

Hard, B. and Tversky, B. (2003). Segmenting ambiguous events. In *Proceedings of the 25th Annual Meeting of the Cognitive Science Society*.

Heider, F. and Simmel, M. (1944). An experimental study of apparent behavior. *American Journal of Psychology*, 57:243–259.

Iida, R., Yasuhara, M., and Tokunaga, T. (2011). Multi-modal reference resolution in situated dialogue by integrating linguistic and extra-linguistic clues. In *Proc. of IJCNLP*, pages 84–92.

Kwiatkowski, T., Zettlemoyer, L., Goldwater, S., and Steedman, M. (2010). Inducing probabilistic ccg grammars from logical form with higher-order unification. In *Proceedings of the 2010 conference on empirical methods in natural language processing*, pages 1223–1233.

Langacker, R. (1987). *Foundations of Cognitive Grammar I: Theoretical Prerequisites*. Stanford University Press.

Mandler, J. M. (1992). How to Build a Baby .2. Conceptual Primitives. *Psychological Review*, 99(4):587–604+.

Marino, J., Banchs, R., Crego, J., de Gispert, A., Lambert, P., Fonollosa, J., and Costa-jussà, M. (2006). N-gram-based machine translation. *Computational Linguistics*, 32(4):527–549.

Mintz, T. H. (2003). Frequent frames as a cue for grammatical categories in child directed speech. *Cognition*, 90(1):91 – 117.

Mukerjee, A. and Sarkar, M. (2007). Grounded perceptual schemas: Developmental acquisition of spatial concepts. In *Spatial Cognition V Reasoning, Action, Interaction*, volume 4387, pages 210–228. Springer Berlin / Heidelberg.

Nayak, S. and Mukerjee, A. (2012). Learning containment metaphors. In *Proc. of CogSci*.

Prasov, Z. and Chai, J. (2008). What's in a gaze?: the role of eye-gaze in reference resolution in multimodal conversational interfaces. In *Proceedings of the 13th international conference on Intelligent user interfaces*, IUI '08, pages 20–29, New York, NY, USA. ACM.

Regier, T. (1996). *The Human Semantic Potential: Spatial Language and Constrained Connectionism*. Bradford Books.

Roy, D. and Pentland, A. (2000). Learning words from sights and sounds: A computational model. *Cognitive Science*, 26:113–146.

Roy, D. and Reiter, E. (2005). Connecting language to the world. *Artificial Intelligence: Special Issue on Connecting Language to the World*, 167:1–12.

Saffran, J., Aslin, R., and Newport, E. (1996). Statistical learning by 8-month-old infants. *Science*, 274(5294):1926–1928.

Satish, G. and Mukerjee, A. (2008). Acquiring linguistic argument structure from multimodal input using attentive focus. In *ICDL 2008*, pages 43 –48.

Siskind, J. (1994). Grounding language in perception. *AI Review*, 8:371–391.

Siskind, J. (1996). A computational study of cross-situational techniques for learning word-to-meaning mappings. *Cognition*, 61:39–91.

Smith, L. and Yu, C. (2008). Infants rapidly learn word-referent mappings via cross-situational statistics. *Cognition*, 106(3):1558 – 1568.

Solan, Z., Ruppin, E., Horn, D., and Edelman, S. (2002). Automatic acquisition and efficient representation of syntactic structures. In *Proc. of NIPS*.

Steels, L. (2003). Evolving grounded communication for robots. *Trends in Cognitive Sciences*, 7(7):308–312.

Strickert, M. and Hammer, B. (2005). Merge SOM for temporal data. *Neurocomputing*, 64:39–71.

Tomasello, M. (2003). *Constructing a Language: A Usage-Based Theory of Language Acquisition*. Harvard University Press.

Yu, C. (2008). A statistical associative account of vocabulary growth in early word learning. *Language learning and Development*, 4(1):32–62.

Yu, C. and Ballard, D. H. (2007). A unified model of early word learning: Integrating statistical and social cues. *Neurocomputing*, 70(13-15):2149–2165.

Yu, C., Ballard, D. H., and Aslin, R. N. (2005). The Role of Embodied Intention in Early Lexical Acquisition. *Cognitive Science*, 29(6):961–1005.

Bayesian Text Segmentation for Index Term Identification and Keyphrase Extraction

David Newman,♢ Nagendra Koilada,♢ Jey Han Lau♡♠ and Timothy Baldwin♡♠
♢ Dept of Computer Science, University of California, Irvine, USA
♡ Dept of Computing and Information Systems, The University of Melbourne, Australia
♠ NICTA Victoria Research Laboratory, Australia
newman@uci.edu, nkoilada@uci.edu, jhlau@csse.unimelb.edu.au, tb@ldwin.net

ABSTRACT
Automatically extracting terminology and index terms from scientific literature is useful for a variety of digital library, indexing and search applications. This task is non-trivial, complicated by domain-specific terminology and a steady introduction of new terminology. Correctly identifying nested terminology further adds to the challenge. We present a Dirichlet Process (DP) model of word segmentation where multiword segments are either retrieved from a cache or newly generated. We show how this DP-Segmentation model can be used to successfully extract nested terminology, outperforming previous methods for solving this problem.

KEYWORDS: Dirichlet Process, segmentation, Bayesian learning, index term, keyphrase extraction.

Proceedings of COLING 2012: Technical Papers, pages 2077–2092,
COLING 2012, Mumbai, December 2012.

1 Introduction

Index terms and keyphrases are widely used to enhance the browsing/accessibility of documents, especially scientific literature (Gutwin et al., 1999). Index terms are the main terms and concepts in a document, as would be contained in the index of a book or edited collection of papers, for example. They typically number in the hundreds for a single volume, and are conventionally annotated for their key occurrences in the document. Keyphrases are assigned to each document in the form of a handful of terms which capture the key topics or concepts covered in the document. Keyphrases are commonly used as a means of clustering/linking documents in digital libraries. While there is no question of the utility of both index terms and keyphrases, annotating a document with index terms is an arduous and largely unrewarding task, and annotators are notoriously inconsistent at index term and keyphrase assignment (Leonard, 1977; Kim et al., 2010). As such, there is a real need for computational methods which both speed up and homogenize the assignment of index terms and keyphrases.

The focus of this paper is an unsupervised methodology for automatically identifying both index terms in a document collection and keyphrases for a single document. The methodology is called Dirichlet Process Segmentation (DP-seg) and was originally developed by Goldwater et al. (2009) for the task of word segmentation over phoneme streams, in the context of child language acquisition. DP-seg is particularly well suited to the tasks of index term identification and keyphrase extraction because: (1) it is able to identify individual type-level occurrences of a given term; (2) it is context-sensitive and able to robustly deal with "nested" multiword terminology (e.g. are *support vector* and *vector machine* also index terms, or just *support vector machine*?); (3) it is highly sensitive and can be applied to small-scale corpora (important for keyphrase extraction, where the input may be a single document); and (4) it is highly parameterizable.

Our contributions in this paper are: (1) the novel application of DP-seg to the tasks of index term identification and keyphrase extraction; (2) construction of a number of index term identification datasets spanning a number of scientific domains; and (3) extensive empirical comparison with standard methods for multiword extraction, and demonstration of the empirical superiority of DP-seg.

2 Related Work

There are several closely related language modeling tasks in this area that differ in important ways. Here we further clarify what tasks we are addressing in this paper, and review related work for each.

We define a multiword term (i.e. n-gram, where $n \geq 2$) as any contiguous sequence of terms in a sentence from some text. We use the usual convention of *types* being the list of distinct multiword terms (at a document or document collection level), and *tokens* being the specific appearances of those multiword terms in the text. Note that we can extract *types* and/or *tokens* either from a single document, or a collection of documents, and our focus for this work will be collection/corpus-based extraction.

Collocation extraction is a well known task that has been extensively researched. In collocation extraction, the goal is to extract a list of types from a single document, often of a particular type (e.g. as defined by a POS tag sequence: Evert (2004); Pecina (2008)). Evaluation is done by comparing to a gold standard produced by lexicographers. Identification of multiword expressions (MWEs) is a related task, but operates at the token level, often relative to a prede-

termined MWE type or lexicon of MWE candidates (Kim and Baldwin, 2010; Baldwin and Kim, 2009). Arguably the closest task to what we are doing is Keyphrase extraction, and indeed our content of interest is scientific literature (Witten et al., 1999; Kim et al., 2010). Keyphrase extraction operates over individual documents, often based on corpus-level lexical analysis. In this paper, in addition to experimenting over keyphrase extraction as conventionally defined, we introduce the additional task of index term extraction.

Extraction of index terms from a corpus is important and useful for a wide variety of digital library and scholarly applications. It extends keyphrase extraction (which is typically performed on a single article or document) to the broader setting of extracting all concepts from a book or collection of papers. Thus index terms tend to be more numerous than keyphrases (the index of a textbook may list hundreds of index terms). Unlike keyphrases, index terms cover all concepts appearing in text — from fairly general concepts (e.g. *training data*) to specific concepts (e.g. *textual entailment*). We are interested in both type-level extraction (for the index) and token-level identification (to be able to identify the individual occurrences in the text) of index terms. An important aspect of our task is uncovering the correct *nesting*. Extraction methods need to correctly determine when to choose a shorter and more general n-gram (e.g. *machine translation*) vs. a longer and more specific n-gram (e.g. *statistical machine translation*), while avoiding extracting nonsensical n-grams (e.g. *statistical machine*).

Keyphrase extraction is closely related to index term extraction. Recently, there was a shared task on keyphrase extraction in SemEval-2010 (Kim et al., 2010), where over twenty teams submitted systems to extract keyphrases from a set of 244 computer science articles. Evaluation was done against a matching set of gold standard keyphrases. We include this SemEval dataset in our experiments, and compare to those published results.

Perhaps the closest related work to our current work is the c/nc-value algorithm (Frantzi et al., 2000), which is designed for extracting and ranking a list of terminology types from a corpus. The c/nc-value algorithm specifically finds and ranks *nested* multiword terminology (such as *support vector* as well as *support vector machine*). The c/nc-value algorithm is a heuristic that takes as input the exhaustive list of 2- through 5-gram noun chunks. The c-value is a score that is an "effective" frequency, which includes an upweighting of $\log n$ for n-grams, to take into account the geometrically fewer n-grams as n increases. The algorithm also includes a discounting of n-grams that appear in longer n-grams. The goal of the algorithm is to discover all nested terminology. The nc-value part of the algorithm is an extra boosting of the signal by using context information. Frantzi et al. showed c/nc-value to work well for producing the lexicon of index term types from a relatively small set of documents.

3 Bayesian Segmentation

Our DP-seg model is a direct adaptation of the Bayesian model of Goldwater et al. (2009). In that work, Goldwater et al. took a stream of phonemes to discover English words, with the goal of correctly identifying word boundaries. We adapt their model and apply it to the problem of taking a stream of words, and finding suitable boundaries between multiword segments. To provide a context of what is an appropriate multiword segment, we have defined our goal as identifying and extracting index terms (i.e. terms that would plausibly appear in an index). We note that our multiword version of this problem operates on a much larger scale than the multi-phoneme problem. Instead of learning a lexicon of 1000s of words from approximately 100 phonemes, we are learning a lexicon of possibly millions of multiword segments from a list of $\sim$100,000 words.

We present and describe the DP-seg model almost verbatim from Goldwater et al. (2009). The model assumes that the sequence of n-grams or multiword segments (MWE) in the corpus is generated using the following four steps:

1. Generate the number of MWE types that will be in the MWE lexicon
2. Generate the token frequency for each MWE type
3. Generate the word-sequence representation of each MWE type
4. Generate an ordering for the set of MWE tokens.

In the unigram version of the model where multiword segments are statistically independent, the ith MWE is generated as follows:

1. Decide if mwe_i is a novel lexical item
2. **(a)** If so, generate an MWE form (words $w_1 \ldots w_M$) for mwe_i
   (b) If not, choose an existing lexical form x for mwe_i

We assign probabilities to each of the above steps as follows:

1. $\text{P}(\text{mwe}_i \text{ is novel}) = \frac{\alpha}{n+\alpha}$, $\text{P}(\text{mwe}_i \text{ is not novel}) = \frac{n}{n+\alpha}$
2. **(a)** $P(\text{mwe}_i = w_1 \ldots w_M | \text{ mwe}_i \text{ is novel}) = p_{\#}(1-p_{\#})^{M-1}\Pi_{j=1}^{M} P(w_j)$
   (b) $P(\text{mwe}_i = x | \text{ mwe}_i \text{ is not novel}) = \frac{n_x}{n}$

where α is a parameter of the model, n is the number of previously generated MWEs, n_x is the number of times lexical item x has occurred in those MWEs, and $p_{\#}$ is the probability of generating a segment boundary. We combine these probabilities to get the following expression for the distribution over mwe_i given mwe_{-i}, the previous MWE:

$$P(\text{mwe}_i = x | \text{mwe}_{-i}) = \frac{n_x}{i-1+\alpha} + \frac{\alpha P_0(\text{mwe}_i = x)}{i-1+\alpha} \quad (1)$$

where P_0 is the probability of generating the MWE anew, as per Step 2(a) above. Note that in all of the above, the MWE can consist of a single word.

As Goldwater et al. point out, this model is quite intuitive. In Step 1, when n is small, we are more likely to generate a new multiword segment. As n grows, we increasingly tend to retrieve the multiword segment from the cache. Also, the model discourages long segments through longer products of probabilities. By having the likelihood of retrieving a multiword from the cache depend on its frequency, we produce a power-law distribution of multiword types, which is appropriate for language modeling.

This model is a type of Dirichlet Process, a popular tool in Bayesian statistics for nonparametric modeling, where the dimensionality of our MWE lexicon grows with the size of the data. The Dirichlet Process has two parameters, the concentration parameter α and the base distribution P_0. Formally, $\text{mwe}_i | G \sim G$, $G|\alpha, P_0 \sim \text{DP}(\alpha, P_0)$. As is customary, we integrate out G, and estimate the conditional probability of sampling a MWE by the Chinese Restaurant Process (CRP). Simply stated, the CRP is where the ith customer entering a Chinese Restaurant sits at table i with probability proportional to the number of people sitting at table i, or sits at a new table with probability proportional to α. Note this generation of MWEs is the two stage adaptor grammar framework described by Goldwater et al. (2005), with P_0 as generator, and CRP as the adaptor. The CRP is the basis of the cache model: One either generate MWEs by either retrieving from cache (existing table), or generate anew (new table).

3.1 Inference

For inference, we use the Gibbs sampling procedure presented by Goldwater et al. The Gibbs sampler considers one possible segment boundary point at a time, so each sample is from a set of two hypotheses, H_0 and H_1. These hypotheses contain all the same segment boundaries except at the one position under consideration, where H_1 has a boundary and H_0 does not. To sample a hypothesis, we calculate the relative probabilities of H_0 and H_1. Since H_0 and H_1 are the same except for a few rules, this is straightforward. Let H^- be all of the structure and data shared by the two hypotheses, including n words. Then, given MWEs w_1 and w_2 and the combined MWE w_{12}, using Equation 1 we get:

$$P(H_0|H^-) = \frac{n(w_{12}) + \alpha P_0(w_{12})}{n + \alpha} \quad (2)$$

$$P(H_1|H^-) = \frac{n(w_1) + \alpha P_0(w_1)}{n + \alpha} \cdot \frac{n(w_2) + I(w_1 = w_2) + \alpha P_0(w_2)}{n + 1 + \alpha} \quad (3)$$

During preliminary experiments, we learned segmentations with DP-seg using between 5000 and 20000 sweeps through each corpus. We performed convergence checks to determine an appropriate number of iterations of the Gibbs sampler. Convergence can depend on thematic coherence and phrasing redundancy in the text (for example, a collection of abstracts from a PubMed search will have more repeated n-grams than a similar sized collection of full-text articles). By creating an approximate gold standard (using 20,000 iterations and averaging over 10 differently initialized runs), we determined that 5,000 iterations of the Gibbs sampler produced reasonably well-converged segmentations. For all experiments, except for those investigating the effect of parameters, we used Goldwater's settings of $\alpha = 20$ and $p_{\#} = 0.5$. We also used an annealing schedule with initial exponent of 0.1, going up to a exponent of 1 after 90% of the iterations, then no annealing for the final 10% of the iterations.

3.2 Pre-Processing

As with many NLP tasks, we have a variety of choices for preprocessing the raw input text. Some of these choices include whether to lemmatize (including stemming as a crude form of lemmatization), whether to POS tag, and whether to remove frequently occurring stopwords. Because we compare DP-seg to c/nc-value algorithm results, for the majority of experiments we selected a preprocessing strategy that was consistent with the input required by the c/nc-value algorithm. We used OpenNLP for lemmatization and POS tagging, and filtered out stopwords from a very limited list of common stopwords consisting of prepositions and determiners (fewer than 100 stopwords in total). Note that this would make terminology such as *part of speech* appear as *part speech*. Since DP-seg does not require *any* preprocessing at all, for some experiments, we omitted all preprocessing to examine its effect.[1]

4 Datasets

We used a combination of new and existing datasets for our experiments. Since there are no readily-available gold-standard datasets for index terms, we generated our own. What constitutes an index term can be subtle and debatable: some index terms are obvious, such as *semantic role labeling*; others are less so, e.g. *training data* is a concept, albeit general, that

[1]Note, however, that preprocessing has a significant impact on runtime for DP-seg: the larger the vocabulary and longer the texts, the longer the runtime.

Collection	Source	Num. Docs	Num. Words
ACL	ACL Corpus	17k fulltext	40M
NIPS	NIPS Corpus	2k fulltext	5.4M
DNA	PubMed search results	60k abstracts	7.0M
HGT	PubMed search results	7k abstracts	0.7M

Table 1: Description and sizes of the four document collections.

could reasonably be found in an index, while *data point* is probably too generic to be found in most indexes.

4.1 Document Collection Data

Our experiments start with four document collections from scientific/technical literature, described in Table 1. The ACL corpus consists of fulltext research articles from computational linguistics (Radev et al., 2009). The NIPS corpus consists of fulltext articles in machine learning and neurobiology (from `books.nips.cc`). The final two document collections are search results sets from PubMed (`www.pubmed.gov`). The DNA collection contains titles and abstracts in the search results from the query *dna microarray*. The HGT collection contains titles and abstracts in the search results from the query *horizontal gene transfer*. All four of these collections are rich in technical terminology and nested index terms, and thus ideal for experimentation.

4.1.1 Creating Gold Standard

We created a list of gold standard index terms from our four datasets to evaluate the performance of DP-seg. Because of the size of these datasets (over 80K documents and 50M words), it was not possible to annotate an exhaustive/comprehensive list of index terms. Therefore we used a pooling strategy to create a gold standard from top-ranked candidate terms, using the top-1000 index terms from DP-seg, c/nc-value and the Student's t-test (a baseline lexical association measure for collocation extraction), restricted to noun chunks. Note that the ranking for c/nc-value and t-test is based directly on the scores returned by the methods, while the ranking for DP-seg is based on the token-level frequency of each term in the output.

We used the union of all of these candidate index terms for annotation, and randomized the set so that annotators had no insights into which candidates came from which method. A preliminary observation of this list indicated that these top-ranked index terms were dominated by bigrams. Since we were interested in understanding the performance of DP-seg and c/nc-value on nested terminology, we also included for annotation the top-100 4-grams and 3-grams, and all 3-gram and 2-gram subphrases for each.

We used a pool of 25 annotators to annotate subsets of the combined list of 6500 candidate index terms, requesting a binary judgment on whether the candidate term was a plausible index term. We had a minimum of 3 annotators per candidate index term, and an average of 4.4 annotations per candidate index term. In cases of disagreement, we took the majority decision. Our instructions to annotators stated:

> For each of the following expressions, judge its appropriateness as an "index term" (as in it would be appropriate for inclusion in the index of a book or edited volume of papers). The expression has a

well-defined standalone meaning and would be appropriate for inclusion in an index of terms for that domain (e.g. *decision tree*, in the case of the machine learning domain, but not *observable markov* as it is an incorrectly-extracted sub-string of something like *partially observable markov decision process*).

The annotation tasked proved reasonably straightforward, with an inter-annotator agreement of 0.65 using Fleiss' kappa, suggesting substantial agreement.

4.2 SemEval Keyphrase Extraction Data

In addition to our four collections of index terms, we used an existing dataset for keyphrase extraction evaluation — the SemEval-2010 keyphrase extraction data. The SemEval data is a collection of 244 scientific articles released as part of a shared task for keyphrase extraction (Kim et al., 2010). In each document, there is a set of author- and reader-assigned keywords, and all keyphrases are stemmed using the English Porter stemmer. The gold standard keyphrases for each document are largely extracted from the document text, although approximately 15%-19% of the keyphrases do not appear in the paper.

To evaluate the participating systems, micro-averaged F-scores are calculated over the top-5/10/15 candidates. A candidate is considered a match if it is an exact match with one of the gold standard keyphrases; partial matching is not considered in the evaluation.

Note that there is a subtle difference between the two tasks: index term identification is annotated/evaluated at the *document collection* level (separately for each of our four document collections), while keyphrase extraction is annotated/evaluated at the *document* level. Another reason for evaluating our method over keyphrase extraction is that the task is well established, and we can benchmark DP-seg against results for pre-existing systems tuned to the task.

5 Results

Standard evaluation for this type of task is based on precision (P), recall (R) and F-score (F), at the level of either types or tokens. In our index term data, we don't have exhaustive annotations, meaning we aren't able to evaluate recall. Instead, we use precision at N (P@N), which measures what proportion of the top-N ranked n-grams match gold standard index terms. We are guaranteed coverage up to N=1000, as that is what was used to create the list of n-grams for annotation. We also use Mean Average Precision (MAP), a scalar version of the precision measurement, making it useful to evaluate choices of parameter settings or other operational decisions.

5.1 Index Term Results

The output of DP-seg is a nonoverlapping partitioning of the input corpus into a sequence of n-gram segments. For all our methods, we require an index term to be an n-gram segment of length $n \geq 2$ that is a noun chunk. Note that we start with a simplistic definition of a noun chunk: any n-gram that includes a sequence of adjectives or nouns ending in a noun. An initial indication of the segmented output from DP-seg can be seen in the segmentation of recent COLING and ACL article titles in Table 2. Based on cursory observation, many familiar bigrams (*dependency parsing*, *entity disambiguation*) appear in segments (note that we are showing the lemmatized and stopped text input). We also see some familiar longer segments (*loopy belief propagation*, *latent variable model*).

Paper ID	DP-seg Title
C10-1002	identify | multiword expression | leverage | morphological | syntactic idiosyncrasy
C10-1007	fast accurate | arc | filtering | dependency parsing
C10-1008	hierarchical | classifier | applied | multi-way | sentiment detection
C10-1032	entity disambiguation | knowledge base population
C10-1034	empirical study | learning rank | tweet
C10-1040	emdc | semi-supervised approach | word alignment
P10-1005	identifying | generic | noun phrases
P10-1006	structural semantic relatedness | knowledge-based method | name entity disambiguation
P10-1007	correct error | speech recognition | articulatory dynamics
P10-1025	towards | open-domain | semantic role label
P10-1035	accurate | context-free parsing | combinatory categorial grammar
P10-1038	conditional random fields | word | hyphenation
P10-1040	word representation | simple | general | method | semi-supervised learning
P10-1044	latent dirichlet allocation | method | selectional preference
P11-1006	fast accurate | method | approximate string search
P11-1017	comprehensive | dictionary | multiword expression
P11-1026	interactive | topic modeling
P11-1027	faster | language models
P11-1033	joint | bilingual | sentiment classification | unlabeled | parallel corpora
P11-1034	pilot study | opinion summarization | conversation
P11-1039	topical | keyphrase extraction | twitter
P11-1048	comparison | loopy belief propagation | dual decomposition | integrated | ccg supertagging | parsing
P11-1053	semi-supervised relation extraction | large-scale | word clustering

Table 2: Sample DP-seg of recent COLING and ACL paper titles (Paper ID is the ID from ACL Anthology).

ACL	NIPS	DNA	HGT
training data (11924)	training data (1475)	gene expression (9822)	horizontal gene transfer (1371)
language model (6518)	data set (1300)	transcription factor (3557)	horizontal transfer (869)
test set (6233)	data point (1217)	gene expression profile (3536)	gene transfer (677)
noun phrase (5884)	loss function (1162)	dna microarray (3516)	escherichia coli (625)
natural language (5239)	training set (1133)	cell line (3462)	lateral gene transfer (586)
machine translation (4734)	objective function (1011)	microarray analysis (3429)	e. coli (524)
training set (4671)	covariance matrix (868)	breast cancer (2605)	phylogenetic analysis (434)
test data (4168)	generative model (867)	gene expression profiling (2313)	gene cluster (372)
data set (3920)	probability distribution (840)	microarray data (2147)	lateral transfer (283)
lexical item (3745)	optimization problem (838)	cdna microarray (1869)	evolutionary history (275)
target word (3703)	graphical model (822)	expression profile (1860)	antibiotic resistance (273)
lexical entry (3504)	special case (818)	target gene (1754)	genomic island (232)

Table 3: Twelve most frequent types, corpus-wide.

Table 3 shows the 12 most frequent *n*-gram types from DP-seg, on our four corpora. The number in parentheses is the non-overlapping count of that exact type in the output of DP-seg (e.g. the 838 count of *optimization problem* for NIPS does not include any counts of *convex optimization problem*). The list of most frequent types in the table looks reasonable, given the domain of each corpus. Note that the most frequent types include some very general concepts (e.g. *training data* and *training set* are in both the ACL and NIPS top-12 list), and in the case of the PubMed search results, some unsurprising specific concepts, including our original query *dna microarray* and *horizontal gene transfer*. Note that the ACL and NIPS corpora only include bigrams in their top-12 list, reflecting the wider range of topics covered in these collections.

Table 4 and Table 5 show the 10 most frequent 3-gram and 4+ gram noun phrases from DP-seg for the ACL and NIPS corpora, with the number in parentheses again showing the count. We see that the 3-grams are approximately one order of magnitude less frequent than the bigrams shown in Table 3. We see another drop in the frequency for the 4+-grams, compared to the 3-grams. In the 4+-grams we see another phenomena: that of the definitional mention of a concept followed by the three (or more) letter acronym, such as *word sense disambiguation wsd*. In the original text, this would have appeared as *Word Sense Disambiguation (WSD)* with

3-grams	4+ grams
natural language processing (1983)	natural language processing system (372)
word sense disambiguation (1838)	statistical machine translation system (348)
machine translation system (1463)	natural language processing nlp (346)
department computer science (1002)	word sense disambiguation wsd (338)
statistical machine translation (973)	support vector machine svm (260)
wall street journal (798)	statistical machine translation smt (238)
source target language (784)	noun verb adjective adverb (220)
semantic role labeling (768)	word error rate wer (207)
natural language processing (712)	natural language understanding system (199)
natural language generation (682)	latent semantic analysis lsa (172)

Table 4: Ten most frequent 3-grams and 4+ grams for ACL.

3-grams	4+ grams
support vector machine (230)	markov decision process mdp (93)
convex optimization problem (169)	support vector machine svm (87)
latent variable model (151)	markov chain monte carlo (86)
stochastic gradient descent (127)	latent dirichlet allocation lda (82)
high dimensional data (110)	markov chain monte carlo mcmc (67)
binary classification problem (103)	reproducing kernel hilbert space (62)
training test set (101)	principal component analysis pca (55)
number training example (101)	dirichlet process mixture model (47)
natural language processing (100)	conditional random field crf (45)
probability density function (97)	reproducing kernel hilbert space rkhs (42)

Table 5: Ten most frequent 3-grams and 4+ grams for NIPS.

capitalization and parentheses around the acronym, which is lost in our preprocessing due to lowercasing and the removal of punctuation, including parentheses.

5.1.1 Empirical Evaluation

The P@N results for DP-seg, c-value, nc-value and t-test are shown in Figure 1. The four panels correspond to each of our four corpora, with P@N on the vertical axis, and N on the horizontal axis. We see that DP-seg outperforms c-value, nc-value and t-test across the range of datasets and values of N. P@N is a useful metric operationally, since one might consider or use the top-N ranked items from a system. We see the greatest difference of DP-seg against c/nc-value and t-test for the DNA and HGT datasets. This is largely due to the greater frequency of nested terminology, and the ability to DP-seg to better handle nested terminology.

When we examine the effectiveness over the top-ranked 3- and 4-grams and their subphrases in Figure 2, we see a consistent picture of DP-seg outperforming c-value and nc-value (note: there is no t-test curve here since t-test was only used to produce bigram index term candidates). The larger difference in results, particularly in the ACL and NIPS datasets, is due to the more challenging task of correctly solving the nesting problem. DP-seg is more adept at segmenting nested terms, e.g. it correctly identifies *statistical machine translation* and *machine translation* with high frequency, but *statistical machine* on its own is ranked quite low. Conversely, c/nc-value performs poorly on this task, incorrectly listing, e.g., *horizontal gene* as one of the top-ranked index terms.

This limitation of the c/nc-value algorithm is highlighted in Table 6, which compares the top-ranked index terms from DP-seg and nc-value on the HGT dataset. We see that DP-seg has 100% precision over the top-12 returned results, while nc-value only has 75% precision. Despite its attempt to properly discount nested subphrases, c/nc-value fails to recognize that terms such

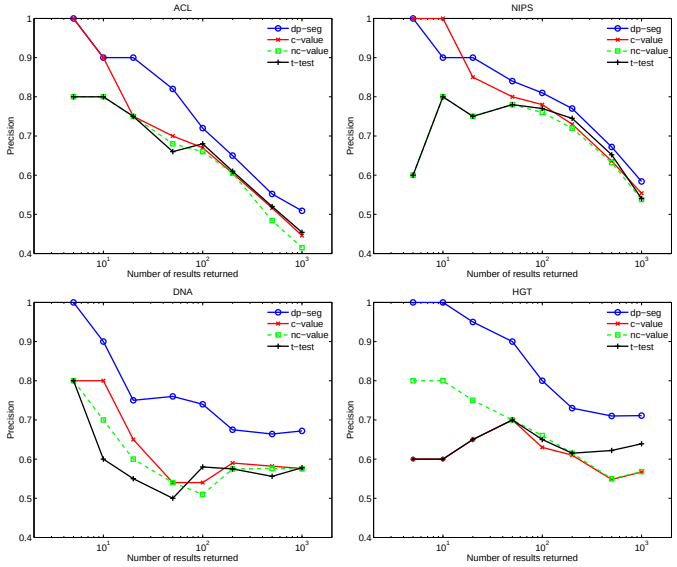


Figure 1: P@N after N results returned, for N up to top-1000 ranked results.

as *horizontal gene* do not exist.

5.1.2 Effect of Model Parameters

The DP-seg model has only two parameters: α and $p_\#$. α controls sparsity, i.e. the number of distinct n-gram types. The probability of segment boundary $p_\#$ controls the average length of n-grams. In our results to date, we have used the default values of $\alpha = 20$ and $p_\# = 0.5$.

We observe the effect of α and $p_\#$ using the NIPS and HGT datasets in Figure 3. The left panel shows the size of the n-gram lexicon versus α. As α increases eight orders-of-magnitude, we see approximately a one order-of-magnitude increase in the size of the multiword lexicon. The right panel shows the percentiles of n-gram lengths versus $p_\#$. As $p_\#$ decreases from 0.999 to 0.001 (viewing right to left), we see an increase in the length of the longest n-grams. Note that around our operating range (i.e. $p_\# = 0.5$), the curves produce a reasonable distribution of n-gram lengths. The average length of n-gram segments has a relatively weak dependence on $p_\#$. As $p_\#$ increases from 0.001 to 0.999 (arguably relatively extreme values), the average length decreases only from 1.34 to 1.30 (NIPS), and decreases from 1.28 to 1.24 (HGT).

The effect of α and $p_\#$ on Mean Average Precision (MAP) for NIPS and HGT is shown in Figure 4. We see a slight decrease in MAP as the DP concentration parameter α increases, although the reduction in MAP is slight compared to the large increase in α. As the probability of segment boundary $p_\#$ increases from 0.1 to 0.9, there is almost no effect on MAP. The rel-

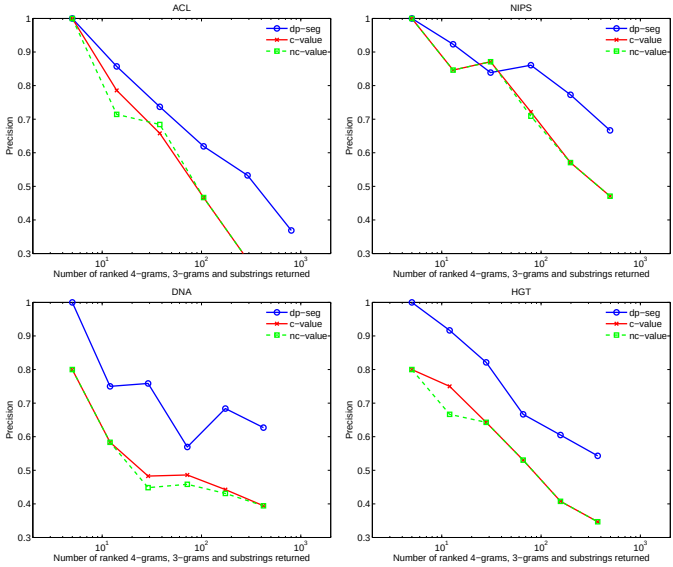


Figure 2: Precision for top-100 4-grams, 3-grams, and subphrases.

atively weak effect of the parameters on MAP is explained by MAP being a precision-focused metric, and that we only have relatively high frequency positive examples labeled. The precision/recall tradeoff is well known for this type of parameter, with precision decreasing and recall increasing as α increases.

Given our Bayesian framework, we could in principle learn α and $p_{\#}$, however we opt for the simpler choice of fixing α and $p_{\#}$, and leave this optimization for future work.

5.1.3 The Effects of Pre-Processing: POS Tagging and Stopword Removal

One advantage of DP-seg is that it can run on any stream of tokens, and does not require lemmatization, POS tagging, parsing or removal of common stopwords (stopping). To allow for a fair comparison with c/nc-value algorithm, we did POS tag and remove stopwords for datasets in Section 4, creating a single input representation used by all algorithms (including t-test). In this section we investigate the effect of relaxing this preprocessing for DP-seg, and running DP-seg without POS tags, and without removing stopwords.

The Mean Average Precision (MAP) results for HGT and NIPS are shown in Table 7a. The first row shows our baseline case of POS tagged and stopped. In the second row, DP-seg is run without POS tags. Since our gold standard only contains noun phrases, we only evaluate using noun phrases. In the third row, DP-seg is run on raw text that only has punctuation removed. Again the final output is filtered for noun phrases. We see a relatively robust result that the performance of DP-seg is relatively good, even with minimal preprocessing.

DP-seg	nc-value
horizontal gene transfer	gene transfer
horizontal transfer	horizontal gene transfer
gene transfer	*horizontal gene**
escherichia coli	lateral gene transfer
lateral gene transfer	horizontal transfer
e. coli	*lateral gene**
phylogenetic analysis	e. coli
gene cluster	escherichia coli
lateral transfer	phylogenetic analysis
evolutionary history	antibiotic resistance
antibiotic resistance	*resistance gene**
genomic island	gene cluster

Table 6: Comparison of top-12 ranked index terms from DP-seg and nc-value on the HGT dataset (incorrect terms are italicized and marked with *).

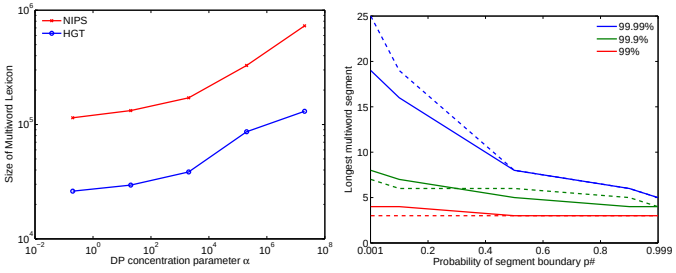


Figure 3: Effect of model parameters α and $p_{\#}$ on the lexicon size and distribution of n-gram length (in the right-hand figure, solid lines are percentiles from NIPS, and dashed lines are percentiles from HGT).

In the SemEval experiments in Section 5.2, DP-seg is run on the provided text as is, without any POS tagging or stopword removal.

5.2 SemEval Results

We ran DP-seg on the SemEval corpus of 244 fulltext articles. We used the input text as is (which was stemmed), for consistency with the published SemEval results. We performed the same evaluation as Kim et al. (2010), computing micro-averaged F-scores calculated over the top-5/10/15 candidates, with a candidate only being considered a match if there is an exact match with one of the gold standard keyphrases. We base the selection of the top-N candidates on only n-grams, ordered by the token-level frequency within the document.

The results are shown in Table 8, with the rows ordered by overall performance. DP-seg (which is completely unsupervised) easily outperforms the unsupervised TF-IDF-based baseline system. In all cases except one, DP-seg outperforms the average score across twenty special-purpose built systems for keyphrase extraction (second row). If DP-seg were competing in this shared task, it would have ranked 11th out of twenty systems based on the top-15 results (and 9th based on the top-5 and top-10 results) and beaten the average for participating systems in all of top-5, top-10 and top-15 evaluations (Kim et al., pear). Note, however, that only two of the systems that performed better than DP-seg are unsupervised (namely El-Beltagy and Rafea (2010) and Bordea and Buitelaar (2010)); all other systems are explicitly trained on the train-

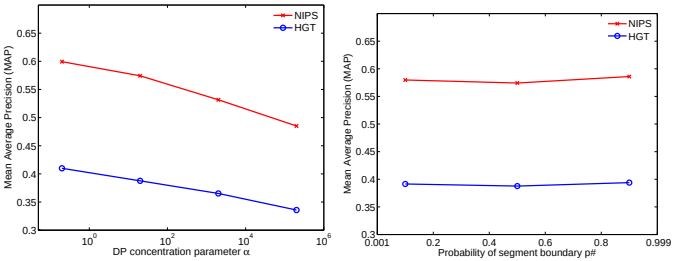


Figure 4: Effect of model parameters α and $p_{\#}$ on Mean Average Precision

Processing	HGT	NIPS
POS tagged & Stopped	0.57	0.40
not POS tagged & Stopped	0.51	0.43
not POS tagged & not Stopped	0.55	0.36

(a) HGT and NIPS Mean Average Precision scores for various pre-processing strategies.

MWE
horizontal gene transfer (1161)
horizontal gene transfer hgt (249)
by *horizontal gene transfer* (176)
horizontal gene transfer and (133)
horizontal gene transfer event (132)
horizontal gene transfer in (90)
via *horizontal gene transfer* (71)
through *horizontal gene transfer* (71)

(b) Effect of not removing stopwords. List of of frequent segments that include "horizontal gene transfer".

Table 7: Effect of POS tagging and stopword removal.

ing documents provided for the task. The fact that we are competitive over the task without any tweaking of the method or reranking of the results is highly encouraging, and complements the P@N results in Section 5.1.1. Also note that this experiment was run without POS tagging and without stopwords removed, adding to the evidence that DP-seg is robust to these preprocessing choices.

6 Discussion and Conclusion

Our results indicate that DP-seg performs well at both index term identification and keyphrase extraction. As part of this, it performs well at disambiguating nested index terms from scientific literature — that is determining an optimal non-overlapping partitioning of a stream of words in a corpus — into a sequence of n-grams. Places where DP-seg tends not to perform so well are examples such as *previous work* and boilerplate text such as authors thanking *anonymous reviewers [for] helpful comments*, which are formulaic and certainly characteristic of the domains we are working with, but not appropriate index terms of keyphrases. Given that DP-seg is based simply on the text stream and doesn't capture positional information in the document, it has no way of differentiating these from conceptually-grounding index terms. One possible research direction for dealing with this issue would be to incorporate positional information (e.g. document zones or argumentative zoning (Teufel et al., 1999)).

As Goldwater et al. found, the unigram DP-seg model tends to be overconfident with very limited data. If it has never seen a particular n-gram, it is reluctant to form that n-gram, but

Method	Top-5			Top-10			Top-15		
	P	R	F	P	R	F	P	R	F
Baseline	22.0	7.5	11.2	17.7	12.1	14.4	14.9	15.3	15.1
Avg System	28.7	9.8	14.6	23.2	15.8	18.8	19.9	20.4	20.1
DP-seg	33.3	10.5	16.0	23.7	16.9	19.7	19.2	21.3	20.2
Best System	39.0	13.3	19.8	32.0	21.8	26.0	27.2	27.8	27.5

Table 8: Precision, Recall, and F-score on the SemEval dataset.

if it has seen just one occurrence, it is very likely to produce more of the same *n*-gram. In response to this, Goldwater proposed a bigram model that includes a Hierarchical Dirichlet Process (HDP) in place of the Dirichlet Process. We plan to investigate the performance of the Bigram/HDP model in future work.

One aspect of index term identification which we have ignored in this research is the location of the index term in the document collection, largely because of the much higher annotation complexity. If we are to deliver on the promise of building an automatic index generator, we need to additionally determine which instances of a given index term should be indexed. For example, for a paper which mentions *support vector machines* on every page, should each page be indexed, just the first mention, or a defining mention (e.g. in a section describing the methodology)? Related to this question is the question of index term equivalence. For example, if the paper mentions both *support vector machines* and *SVMs*, we would ideally like to be able to recognize them as interchangeable and alternative forms of the same index term. Again, the current model is unable to pick up on such equivalences, but a fully-functioned index term identification system should be able to deal with cases such as this, suggesting a direction for future work. Another possilble extension is the determination of noun chunk boundaries, which is currently based on a simple POS sequence heuristic. In terms of precision, the heuristic is robust and not problematic, but in terms of recall, we potentially miss more complex term candidates. Using a noun phrase chunker or parser could be used to boost recall.

An interesting observation is the difficulty c/nc-value has in correctly extracting highly ranked index terms for the HGT dataset, as was shown in Table 6. We believe that, while c/nc-value is designed to discount shorter *n*-grams that appear as subphrases in longer *n*-grams, the wide variety of contexts makes this a hard problem, and that there is no obvious fix to c/nc-value to solve this problem. We argue that the Bayesian segmentation and generative model from DP-seg provides a more principled solution to this problem — and one that performs significantly better. The Bayesian framework also allows for other extensions to incorporate additional lexical or syntactic data into the model. We leave these investigations for future work.

To summarize, we have applied an unsupervised Bayesian method to the tasks of index term identification and keyphrase extraction. Index term identification is the task of automatically determining terms to include in a literary index for a document collection, and we constructed a total of four datasets across a range of scientific domains. Keyphrase extraction is defined in the conventional way, and was evaluated relative to the SemEval-2010 dataset. We found the Bayesian method to outperform both a multiword terminology extraction method and a collocation extraction baseline over the index term identification task, and to rank above the average results for participating systems in the SemEval-2010 task, beaten by only two unsupervised systems (in addition to a number of supervised systems), without any tuning to the task.

Acknowledgements

DN supported by the Intelligence Advanced Research Projects Activity (IARPA) via Department of Interior National Business Center contract number D11PC20155. The U.S. government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright annotation thereon. Disclaimer: The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of IARPA, DoI/NBC, or the U.S. Government. DN also supported by NSF-1106434. NICTA is funded by the Australian government as represented by Department of Broadband, Communication and Digital Economy, and the Australian Research Council through the ICT centre of Excellence programme.

References

Baldwin, T. and Kim, S. N. (2009). Multiword expressions. In Indurkhya, N. and Damerau, F. J., editors, *Handbook of Natural Language Processing*. CRC Press, Boca Raton, USA, 2nd edition.

Bordea, G. and Buitelaar, P. (2010). Deriunlp: A context based approach to automatic keyphrase extraction. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, pages 146–149, Uppsala, Sweden.

El-Beltagy, S. R. and Rafea, A. (2010). Kp-miner: Participation in semeval-2. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, pages 190–193, Uppsala, Sweden.

Evert, S. (2004). *The Statistics of Word Cooccurrences: Word Pairs and Collocations*. PhD thesis, University of Stuttgart.

Frantzi, K., Ananiadou, S., and Mima, H. (2000). Automatic recognition of multi-word terms:. the c-value/nc-value method. *International Journal on Digital Libraries*, 3:115–130.

Goldwater, S., Griffiths, T., and Johnson, M. (2005). Interpolating between types and tokens by estimating power-law generators. In *NIPS*.

Goldwater, S., Griffiths, T., and Johnson, M. (2009). A bayesian framework for word segmentation: Exploring the effects of context. *Cognition*, 112:21–54.

Gutwin, C., Paynter, G., Witten, I., Nevill-Manning, C., and Frank, E. (1999). Improving browsing in digital libraries with keyphrase indexes. *Decision Support Systems*, 27(1-2):81–104.

Kim, S. and Baldwin, T. (2010). How to pick out token instances of English verb-particle constructions. *Language Resources and Evaluation*, 44(1-2):97–113.

Kim, S. N., Medelyan, O., Kan, M., and Baldwin, T. (2010). Semeval-2010 task 5 : Automatic keyphrase extraction from scientific articles. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, pages 21–26, Uppsala, Sweden.

Kim, S. N., Medelyan, O., Kan, M.-Y., and Baldwin, T. (to appear). Automatic keyphrase extraction from scientific articles. *Language Resources and Evaluation*.

Leonard, L. (1977). Inter-indexer consistency studies, 1954-1975: a review of the literature and summary of study results. Technical report, Graduate School of Library and Information Science. University of Illinois at Urbana-Champaign.

Pecina, P. (2008). *Lexical Association Measures*. PhD thesis, Charles University.

Radev, D. R., Muthukrishnan, P., and Qazvinian, V. (2009). The ACL anthology network corpus. In *Proceedings, ACL Workshop on Natural Language Processing and Information Retrieval for Digital Libraries*, Singapore.

Teufel, S., Carletta, J., and Moens, M. (1999). An annotation scheme for discourse-level argumentation in research articles. In *Proceedings of the 9th Conference of the European Chapter of the Association for Computational Linguistics (EACL'99)*, pages 110–117, Bergen, Norway.

Witten, I., Paynter, G., Frank, E., Gutwin, C., and Nevill-Manning, C. (1999). Kea: Practical automatic keyphrase extraction. In *Proceedings of the Fourth ACM Conference on Digital Libraries (DL-1999)*, pages 254–255, Berkeley, California.

Exploiting Category-Specific Information for Multi-Document Summarization

Jun−Ping Ng[1] *Praveen Bysani*[1]
Ziheng Lin[2] *Min−Yen Kan*[1] *Chew−Lim Tan*[1]
(1) National University of Singapore, 13 Computing Drive, Singapore 117417
(2) SAP Research, SAP Asia Pte Ltd, 30 Pasir Panjang Road, Singapore 117440
{junping,bpraveen,kanmy,tancl}@comp.nus.edu.sg, ziheng.lin@sap.com

ABSTRACT
We show that by making use of information common to document sets belonging to a common category, we can improve the quality of automatically extracted content in multi-document summaries. This simple property is widely applicable in multi-document summarization tasks, and can be encapsulated by the concept of *category-specific importance (CSI)*. Our experiments show that CSI is a valuable metric to aid sentence selection in extractive summarization tasks. We operationalize the computation *CSI* of sentences through the introduction of two new features that can be computed without needing any external knowledge. We also generalize this approach, showing that when manually-curated document-to-category mappings are unavailable, performing automatic categorization of document sets also improves summarization performance. We have incorporated these features into a simple, freely available, open-source extractive summarization system, called SWING. In the recent TAC-2011 guided summarization task, SWING outperformed all other participant summarization systems as measured by automated ROUGE measures.

KEYWORDS: text summarization, csi, guided summarization, tac.

Proceedings of COLING 2012: Technical Papers, pages 2093–2108,
COLING 2012, Mumbai, December 2012.

1 Introduction

Studies have been done on many facets of text summarization including multi-document summarization (Radev et al., 2004), query focused summarization (Daumé III and Marcu, 2006), personalized summarization (Díaz and Gervás, 2007), temporal summarization (Bysani et al., 2009), and more recently guided summarization (Owczarzak and Dang, 2010, 2011).

In multi-document summarization, a *topic* consists of a set of related documents. The goal is to generate a coherent summary from this set of documents with minimal information redundancy. In the guided summarization tasks defined by the recent Text Analysis Conference's (TAC) shared tasks, each topic is additionally assigned to one of several broad *categories* such as *Accidents and Natural Disasters* or *Attacks* (see Figure 1). In traditional query-focused summarization, a narrative specific to each topic serves as a hint to the content required in the target summary. However in guided summarization, the narrative is replaced with a series of category-specific templates which contain information elements, or *aspects*. For example, *WHEN* is an aspect that is shared by both the *Accidents and Natural Disasters* and *Attacks* categories. Note that aspects are not specific to a topic; rather, they are associated with the category to which the topic belongs. A summary for a topic should cater to all the aspects of its associated template. Such guided summarization can be usefully applied to product opinion summarization, personalization of summaries for users, and improving user experience in question answering scenarios.

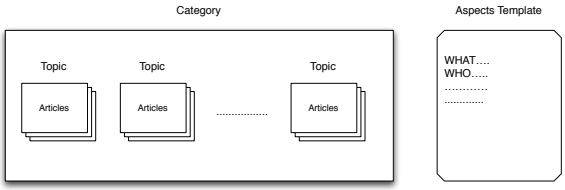


Figure 1: How articles, topics, categories and aspects come together.

Recently, Haghighi and Vanderwende (2009) proposed several content models for summarization. Their models find aspects within a topic which are subsequently combined using KL-divergence as a criterion for selecting relevant sentences. Conroy et al. (2010) augmented their CLASSY system with a query generation component that expands query terms for each aspect of the associated category by performing searches over Google, dictionaries, thesauri and authored world knowledge. Steinberger et al. (2010) generated guided summaries by framing the problem as an information extraction task. Aspect information extracted from an entity extractor is coupled with latent semantic analysis to capture relevant information. They also built lexicons for some category aspects that are not identified by the event extractor. External knowledge such as Wikipedia is also used by many groups for this task. In (Varma et al., 2010), a large set of relevant articles were manually selected from Wikipedia for each category. These articles were used to build domain models, and later to extract important sentences containing events mentioned in the template.

Most of the prior work in guided summarization focuses on producing a summary by selecting relevant aspects common within *a single topic*. However as noted earlier, aspects are shared over multiple topics in a category; thus topic-oriented models do not exploit knowledge shared among topics within the same category. We hypothesize that category-specific information does

encode a useful signal that can improve the quality of guided summaries. To this effect, we propose and develop a robust sentence-extractive summarizer adopting the standard, supervised machine learning framework: we extract features from the input documents, utilize the features to rank the importance of input sentences through a regression model, and finally apply the model on new, unseen test documents.

The fundamental innovation that our summarizer makes over the previous state-of-the-art is that it makes use of the information derived from the category of a topic to calculate the category-specific importance (CSI) of each sentence. We capture CSI through two novel features – *category relevance score* and *category Kullback-Liebler divergence score* – that are explained in later sections of the paper.

Our approach is different from (Conroy et al., 2010; Steinberger et al., 2010) which compiled lexicons manually for each category aspect. Words in these pre-compiled lexicons are treated with equal importance for a category, whereas our method automatically discerns between the different saliency of words across a category. This allows us to address the problem of low recall that hampers the performance of manually-compiled lexicons.

Aker and Gaizauskas (2009) had also made use of the concept of category-specific information for automatic captioning of images. Similar to our proposed approach, they exploited the inherent differences across different object types to influence content selection. Our work is different in two key aspects: 1) our computed statistics are based on actual content to be summarized, instead of a pre-assembled corpus, and 2) besides considering information across categories, we also make use of information across topics, within a category.

When compared with the state-of-the-art summarizers submitted to TAC-2011, our system significantly outperforms all other systems as reported in (Ng et al., 2011).

2 Corpus

The categorization of topics in the guided summarization task at TAC makes the shared task datasets suitable corpora for our work. We use the dataset provided in TAC-2010 for training our system and the TAC-2011 dataset for testing purposes. The documents in TAC-2010 are extracted from AQUAINT and AQUAINT-2; documents used in TAC-2011 came from the newswire portion of the TAC-2010 KBP source data. The test dataset consisted of 44 topics, divided into five categories. The structure of the training data is similar, containing 46 topics. We use only the articles from Set A for our experiments as the task of summarizing Set B, was an update summarization task, a separate task by itself. The distribution of topics into categories for TAC-2010 and TAC-2011 is provided in Table 1. In the rest of this paper, we abbreviate some of the category names for brevity. For example instead of *Accidents and Natural Disasters*, we will use *Accidents*.

Category	TAC-2011	TAC-2010
Accidents and Natural Disasters	9 (90)	7 (70)
Attacks	9 (90)	7 (70)
Health and Safety	10 (100)	12 (120)
Endangered Resources	8 (80)	10 (100)
Investigations and Trials	8 (80)	10 (100)

Table 1: Distribution of topics and documents into categories in TAC-2010 and 2011. The number of documents per category is shown in parentheses.

The TAC-2011 guided summarization task was to write a 100-word summary for a given topic covering all the aspects. A template of aspects for the category *Health and Safety* is shown in Table 2 as an example.

Aspect	Description
WHAT	what is the issue
WHO_AFFECTED	who are affected by the issue
HOW	how are they affected
WHY	why the health/safety issue occurs
COUNTERMEASURES	prevention efforts

Table 2: Template of aspects for the *Health and Safety* category.

Four human-written model summaries are provided per topic for each set. These summaries are used as a gold standard for evaluating machine generated summaries. Both automatic and manual measures were utilized by the TAC organizers to evaluate summaries. Automatic evaluation is commonly performed using ROUGE (Lin, 2004), and was used in TAC. ROUGE determines the quality of a summary through overlapping units such as n-grams, word sequences, and word pairs with human written summaries. Manual measures adopted by TAC organizers included pyramid scoring (Nenkova et al., 2007) and subjective assessments about the quality of the summaries. Since the original TAC manual evaluation team is not known or available, manual evaluation of new summarization systems is not possible. As such, we need a fair, objective comparison of our results with previously published results, and can only adopt automated methods. For this reason, we adopt the automatic ROUGE-2 and ROUGE-SU4 measures. While not ideal, these measures have been found to generally correlate well with manual judgments (Lin and Hovy, 2003).

3 Methodology

Our system, SWING, is a sentence-extractive summarizer that is designed to be an easy-to-use and an effective testbed for comparative evaluation of summarization methods. Input data is pre-processed using standard techniques, incorporating stop word removal and stemming for better computation of relevance. Our summarization system is fundamentally based on a supervised learning framework. A set of features is derived for each sentence in the input documents to measure their importance. We compute two classes of features, at the topic and category levels. We first discuss a set of *generic* features used in SWING. The feature scores are combined together with a set of weights derived from support vector regression (SVR) (Gunn, 1998). Finally, the Maximal Marginal Relevance (MMR) algorithm (Carbonell and Goldstein, 1998) is used to perform sentence re-ranking and selection. Later in Section 4, we introduce features to compute our key innovation: the category-specific importance (CSI) of sentences. SWING combines both generic features and CSI features to produce guided summaries.

3.1 Generic Features

Each sentence is represented by a vector of feature scores for learning. We used three features: (1) sentence position, (2) sentence length, and (3) a modified version of document frequency to calculate the generic topic relevance of a sentence.

Sentence position (Edmundson, 1969) is a popular feature used in summarization especially for news domain. The intuition is that leading sentences in a news article usually contain

important, summary-worthy information. Accordingly, the score of this feature is gradually decreased from the first sentence to the last sentence in a document based on its position.

Sentence length is a binary feature that helps in avoiding noisy short text in the summary. The value of this feature is 1 if the length of sentence is at least 10, and zero otherwise. The value 10 is empirically determined in our system tuning.

Interpolated N-gram Document Frequency (INDF) is an extended formulation of the popular document frequency (DF) measure. The efficacy of DF in summarization has been previously demonstrated by (Schilder and Kondadadi, 2008; Bysani et al., 2009). It computes the importance of a token as the ratio of the number of documents in which it occurred to the total number of documents within a topic. We extend the use of DF from unigrams to bigrams. INDF is the weighted linear combination of the DF for unigrams and bigrams of a sentence. Since bigrams encompass richer information and unigrams avoid problems with data sparseness, we choose a combination of both. The INDF of a sentence s, is computed as:

$$INDF(s) = \frac{\alpha(\sum_{w_u \in s} DF(w_u)) + (1-\alpha)(\sum_{w_b \in s} DF(w_b))}{|s|}$$

where w_u are the unigram and w_b are the bigram tokens in sentence s. α is the weighting factor that is set to 0.3, after tuning on the development set.

3.2 Training and SVR

Each sentence is scored with the three features explained above. The features are given weights by a support vector regression model, following the methodology described in (Bysani et al., 2009). We train the regression model using the ROUGE-2 similarity of the sentences with human models as the objective to maximize. Data from TAC-2010 is used as the training corpus, and the trained regression model is used to predict the saliency scores of each sentence in the TAC-2011 test set.

3.3 Sentence Re-ranking

After each sentence has been scored, the maximal marginal relevance (MMR) (Carbonell and Goldstein, 1998) algorithm is used to re-rank and extract the best sentences to generate a 100-word summary. In our implementation, the MMR of a sentence s is computed as:

$$MMR(s) = Score(s) - R2(s, S)$$

where $Score(s)$ is the score predicted by the regression model, S is the set of sentences already selected to be in the summary from previous iterations, and $R2$ is the predicted ROUGE-2 score of the sentence under consideration (s) with respect to the selected sentences (S).

3.4 Post-Processing

There are many extraneous text fragments in the corpus that are uninformative. These include news agency headers and the reporting date of the articles, among others. These are removed automatically during post-processing from the summaries with the use of a modular post-processing system that matches regular expressions.

Table 3 provides the evaluation results of a baseline summarizer, `Generic`, when using only the above discussed generic features on the test dataset. We also provide the results of two

baseline systems commonly used in TAC for comparison. `FirstSent` returns the top sentences from the most recent article until the summary length (100 words) is reached, and `MEAD` is the output of *MEAD*, a popular open-source summarizer[1].

Configuration	ROUGE-2	ROUGE-SU4
`Generic`	0.13392	0.16513
`FirstSent`	0.06410	0.09934
`MEAD`	0.08682	0.11749

Table 3: ROUGE scores for baseline summarizer with generic features and common TAC baselines.

The ROUGE scores indicate that putting these generic features together surpassed the baseline systems by a huge margin, and is a competitive configuration used to compare with in the remaining parts of this paper.

4 Category Specific Information

In the guided summarization task, summaries are generated for each topic, where each topic belongs to one or more categories. The purpose for providing this manually-given classification is so that the summaries can focus on the content related to the aspects associated with the category. We want to leverage this knowledge of the category of a topic to improve generated summaries.

In this extractive summarization scenario, we formulate the summarization task as supervised regression, where the system learns to score the saliency of sentences. The idea behind CSI is to exploit information which is specific to a particular category, and use this as a guide to the saliency of sentences from the source documents. One such possible category-specific information could be how words are used within the category's topics. For a category such as *Accidents*, we may expect to see words like "died", "collision" in the associated source documents more commonly than we would in a general piece of English text. For multi-document summarization, we hypothesize that the word frequency statistics will be similar for document sets within the same category and will be different than those across document sets from different categories. For example, a set of news articles on "Borneo Ferry Sinking" may share similar word statistics with another set of news articles reporting "Minnesota Bridge Collapse" as these two sets belong to the category of *Accidents*. However, the word statistics will have a different distribution when compared to a set of news articles on "Pet Food Recall" (*Health*) as they are from different categories.

To find out if there is indeed a difference in word frequencies across each of the categories, we independently performed an analysis of the word usage in each category. To quantify this difference, we applied the log-likelihood ratio test (LLR) (Dunning, 1993). The LLR of a word w across two categories c_1 and c_2 is defined as:

$$LLR(w) = 2 \times \sum_{i \in c_1, c_2} \left(a_i \times \ log \left(\frac{a_i \times F}{b_i \times f(w)} \right) \right)$$

where a_i is the frequency of word w and b_i is the total frequency of all words in category c_i. F is the total frequency of all words, and $f(w)$ is the frequency of w across all categories. A

[1] `http://www.summarization.com/mead/`

word with a high LLR value implies that it co-occurs in both categories surprisingly often, or surprisingly rarely.

We obtained a list of words with high LLR value (99th percentile; 0.1% level; value = 6.63) for each category with respect to all other categories. For illustration, the top ten words for each of the five categories are shown in Table 4.

Category	Words
Accidents	bridge, bangladesh, crane, weather, spill, cyclone, survivor, earthquake, oil, crash
Attacks	attack, school, police, gunman, terrorist, shoot, condemn, fbi, molest, nuclear
Health	food, safety, children, recall, sleep, cancer, organ, heart, blood, risk
Endangered Resources	water, turtle, coral, ivory, global, conserve, warm, decline, poach, tuna
Investigations	charge, trial, guilty, investor, testify, plead, robbery, taylor, former, conspiracy

Table 4: Top ten words listed in decreasing order of LLR values in each of the TAC categories.

The table shows that almost all of the words are semantically related to their corresponding categories. For example, the first word for the category *Attacks* is actually "attack", while that for the category *Endangered Resources* is "water". We expect that a good summary will contain a fair amount of these category-specific words. To validate this, we examine the densities of these words in both the model summaries and all of the document sets that belong to a category. Here, density is computed as the ratio of the sum of the term frequencies of all the words found in the list to the total term frequency of the category. If a word is used more frequently in a model summary compared to a more general document set, we would expect a higher density value for the model summary.

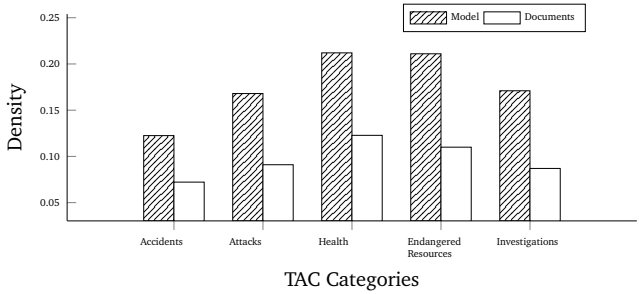


Figure 2: Comparison of density of category-specific words across model summaries and document sets.

The word densities for both the model summaries and document sets for each category are plotted in Figure 2. It shows that the words identified by the LLR criterion are indeed used more often in the model summaries than in the document sets. This shows that a good summary will

contain more category-specific words, and thus gives solid evidence for our intuition that the difference in word usage across each category is a useful guide in generating a good summary.

4.1 Category-Specific Features

Having determined the efficacy of category-specific word usage, we design two features, category relevance score (CRS), and category KL-divergence score (CKLD), to model and exploit this property.

Category Relevance Score (CRS) computes the importance of a word with respect to a category, using the frequency statistics of the word in constituent topics and topic documents of the category. As every topic in the category is related, the topic frequency of a word is directly proportional to its categorical relevance. Similarly, the larger the number of documents a word appears in within the category, the more relevant it is to the category. CRS is the linear interpolation of frequency scores at topic (TLF) and document level (DLF). The score of a sentence s in category c, is calculated as:

$$CRS_c(s) = \frac{\sum_{w \in s}(\beta \times TLF_c(w) + (1-\beta) \times DLF_c(w))}{|s|}$$

where $TLF_c(w)$ and $DLF_c(w)$ are computed as:

$$TLF_c(w) = \frac{|\{t : w \in t, \ \ \forall t \in c\}|}{|T_c|}$$

$$DLF_c(w) = \frac{|\{d : w \in d, \ \ \forall d \in c\}|}{|D_c|}$$

where t and d represent topic and document, respectively, and T_c and D_c are the sets of topics and documents in category c, respectively. The value of β was determined empirically, optimally set to 0.7. This setting highlights that topic-level influence is more important that of the document level.

Category KL-Divergence Score (CKLD) is a differential measure that calculates the importance of a word using KL Divergence. Also known as information divergence, it quantifies the information gain between two probability distributions. Category KLD (CKLD) measures the divergence of probability distribution of a word in the current category (c) to its distribution in the whole corpus (C). The greater the divergence from C, the more informative the word is for category c. The CKLD value of a sentence s in category c is given as:

$$CKLD_c(s) = \sum_{w \in s} \left(p_c(w) \times log \frac{p_c(w)}{p_C(w)} \right)$$

where $p_c(w)$ is the probability of word w in category c and $p_C(w)$ is the probability of word w in the corpus.

The key difference between CRS and CKLD is that CRS tries to promote words which are important to all the topics within a category, while CKLD seeks words which are unique in terms of word usage in a category. In other words, CRS is an *intra-category* measure, while CKLD is an *inter-category* measure. The distinction between these two is subtle but important. Table 5 shows the top five words in descending order of CRS and CKLD in each category.

Consider two words such as "report" and "Madoff" for the category of *Investigations*. The word "report" ranks top for CRS in this category and appears in three categories, while "Madoff" ranks top for CKLD and only appears in *Investigations*. CKLD will be able to detect if these two words are used differently from how they are used in the other categories, which explains the fact that most words in the list appear only in one category. In this example, the word "Madoff" is a person name which is likely important only in some topics in *Investigations* but not in other categories. On the other hand while "report" is important to the *Investigations* category (it appears in seven out of eight topics in this category), it is also found important in two other categories (*Accidents* and *Attacks*). We hypothesize that these intra- and inter-category aspects of CRS and CKLD will be complementary to each other, which we will validate in the experiment section.

Category	CRS	CKLD
Accidents	official, people, report, news, accident	crane, bridge, construction, java, people
Attacks	attack, report, killed, state, police	attack, pirate, police, school, israel
Health	product, research, company, increase, time	food, toy, sleep, vitamin, product
Resources	conserve, world, protect, manage, country	coral, water, tuna, elephant, turtle
Investigations	report, charge, people, killed, family	madoff, taylor, alvarez, prosecutor, charge

Table 5: Top five words listed in decreasing order of CRS and CKLD, for each category.

4.2 Experiments

To evaluate the efficacy of the proposed category-specific importance features (*i.e.*, CRS and CKLD), we add them to the baseline summarizer described earlier. Table 6 shows the ROUGE measures of the various summarizer configurations when tested on the TAC-2011 dataset. `Generic+CRS` uses the CRS feature alongside the generic features described in the previous section (*i.e.*, sentence position, sentence length, and INDF). Likewise `Generic+CKLD` uses the CKLD feature in addition to the generic features, and `SWING` which is essentially `Generic+CRS+CKLD` uses both CRS and CKLD. We also include the results achieved by two other top-performing systems, `CLASSY` (Conroy et al., 2011) and `POLYCOM` (Zhang et al., 2011), at TAC-2011 for comparative purposes.

Configuration	ROUGE-2	ROUGE-SU4
`SWING`	0.13796	0.16808
`Generic+CRS`	0.13702	0.16788
`Generic+CKLD`	0.13525	0.16649
`CLASSY`	0.12780	0.15812
`POLYCOM`	0.12269	0.15974

Table 6: ROUGE scores over TAC-2011 dataset. Results for `CLASSY` and `POLYCOM` are reported after the jackknifing procedure, as released by the shared task organizer.

The table shows that adding either one of the category-specific features to `Generic` outperforms the two top-performing summarizers on both ROUGE-2 and ROUGE-SU4. When comparing `Generic+CRS` and `Generic+CKLD`, `Generic+CRS` slightly outperforms `Generic+CKLD` with 0.00177 for ROUGE-2 and 0.00139 for ROUGE-SU4. This is explained by the fact that CRS

captures intra-category importance of words which focuses on word usage within a topic of a specific category. As TAC systems are to summarize a single topic (as opposed to a whole category), it is reasonable that CRS provides more improvement when we look at the ROUGE scores on the topics. We expect that if systems were asked to instead summarize categories, CKLD would yield a larger improvement as CKLD captures inter-category importance of words which would be more pertinent to this hypothetical task.

When both category-specific features are used (*i.e.*, SWING), the performance for both ROUGE-2 and ROUGE-SU4 are higher than that for Generic+CRS and Generic+CKLD. This validates our hypothesis that both features are complementary to each other as they measure word statistics from different angles (*i.e.*, intra- vs. inter-category). Two-tailed student's t-test verifies that SWING significantly outperforms Generic, CLASSY, and POLYCOM (p-value < 0.05).

4.2.1 Chunk-sensitive CSI Scoring

Up to this point, we have assigned sentence-wide CSI scores; the sentence score aggregates the CSI scores of all words in the sentence. However consider the word "bridge" from the category of *Accidents* — "bridge" can be part of a NP chunk (*e.g.*, *The bridge across the road...*), or part of a VP chunk (*e.g.*, *Let's bridge our differences...*). When found in a NP chunk, we can (casually) associate the use of the word with accidents. For example traffic accidents can happen on bridges, or bridges can collapse. When found in a VP chunk however, this association is lost. It is unfair then to regard a sentence as being more salient to the category *Accidents* if it contains the word "bridge" outside of a NP chunk.

We postulate that there is a need to first determine the word's role within a sentence, before deciding if it contributes to the saliency of the sentence. To verify this, we build variants of our scorer that ignores the CSI scores of word occurrences when they appear in chunks outside of a target chunk type.

To implement this, we parse all the input sentences from the source documents using the OpenNLP constituent grammar parser[2]. From the parses, we identify the constituent noun phrases (NP), verb phrases (VP) and prepositional phrases (PP). Instead of computing the CSI value of every word in the sentence, only the words found in a particular syntactic chunk (*i.e.*, one of NP, VP, and PP) are used to compute its score. The ROUGE evaluation results of the experiments are shown in Table 7.

Configuration	ROUGE-2	ROUGE-SU4
NP	0.13934	0.16836
VP	0.1354	0.16602
PP	0.13494	0.16592
All	0.13796	0.16808

Table 7: ROUGE scores of SWING when CSI computation is restricted to specific syntactic chunks. "All" denotes the non-chunk specific system, where results are repeated from Table 6.

By restricting CSI scores to word occurrences found only within NP chunks, we obtain a statistical significant improvement ($p < 0.05$) on the ROUGE-2 score. This result suggests that it is indeed useful to also consider the function of a word within a sentence.

[2] http://opennlp.sourceforge.net/projects.html

We note that restricting scoring to either just VP or PP chunks reduced performance significantly when compared to the baseline on the other hand. We suspect that word usage within VP and PP chunks could be more generic, and thus do not convey additional notions of saliency. It will be insightful to investigate this further in future work.

4.2.2 Clustering Accuracy

So far, we definitively demonstrated the utility of CSI features in guided summarization. However, the previous experiments made use of gold-standard, human-assigned categories for each topic, provided manually by the TAC organizers. In more typical multi-document summarization scenarios, such gold-standard categorization is unavailable. Might CSI features still be useful when such categorization is generated using less-than-perfect automatic categorization? To answer this, we set out to measure the effect that the quality of category assignments have on CSI feature efficacy.

We start by placing all the topics into one large cluster, ignoring the original human-assigned categories. Various automated clustering algorithms are then run to cluster the topics. The summarizer is then provided with these automatic clustering results to compute summaries as per the pipeline previously discussed.

Since our focus in this experiment is to measure the robustness of the CSI features, a simple clustering method suffices. We used a simple approach in which a bag-of-words feature is used for the clustering, considering only words from the first sentence of each document. This is reasonable as the first few sentences of a news article often give a good indication of the content to follow in the rest of the article.

We experiment with three clustering algorithms of K-Means, X-Means and Expectation Maximization (EM), using different numbers of clusters. All experiments were carried out using the *WEKA* (Hall et al., 2009) package and used only the simple bag-of-words feature to construct clusters. Evaluation results of the clustering algorithms are shown in Table 8 along with p-values from the two-tailed Student's t-test when compared with SWING that used the gold-standard clusters provided by TAC. Each configuration in the table uses the automatic clustering results assigned by the corresponding clustering algorithm while computing the relevant CSI scores.

Clustering Method	Size	ROUGE-2	p-value
EM	3	0.13547	0.156
	4	0.13659	0.158
	5	0.13647	0.154
X-Means	3	0.1364	0.101
	4	0.13603	0.146
	5	0.13546	0.117
K-Means	3	0.13574	0.173
	4	0.13696	0.311
	5	0.13569	0.365

Table 8: ROUGE scores of SWING when paired with different clustering schemes. *p-values* are with respect to results obtained when SWING is paired with human-assigned categories from the TAC datasets.

From Table 8, we see that while all automatic clustering algorithms report a drop in ROUGE compared to the use of gold-standard categories, the difference in the scores were generally not

statistically significant. This is a positive result as it shows that our CSI features can be useful even if perfect categorization results are not available; automatic clustering can be employed to create the necessary input to calculate CSI features.

The drop in performance is expected: since CSI features measure information specific to a category, noisy clusters produced by the automatic algorithms are more likely to be less well-defined than the human assigned gold-standard categories. Any category-specific information will be diluted, and thus features seeking to exploit this information will be adversely affected.

Results among the clustering methods were inconclusive. Variation in the methods employed and the number of clusters used led to mixed results that did not point towards a clear direction to favor.

5 Analysis

To gain insight on how category specific information affects our system, we manually examined the improvements SWING made over Generic. In the test topics, we found that the CSI version selected alternative sentences in 14 out of the 44 topics, roughly 1/3 of all summaries. The categories *Accidents*, *Attacks*, and *Investigations* have 3 replacements each while *Health* and *Endangered Resources* have 1 and 4 replacements, respectively. Less important a phenomenon is that the summary sentences were re-ordered in 10 instances, resulting in minor changes in ROUGE scores, as the last sentence is trimmed to keep the summary length to 100 words. The changes made by CSI in the selection are thus frequent, altering some summaries in a substantial way, made evident by the change in ROUGE score.

To illustrate the utility of leveraging on category specific information, differences between both systems for a topic in *Accidents* category are provided in Figure 3. The '—' sign represents that the sentence is excluded and '+' sign shows that the sentence is included in SWING. The first sentence that was replaced has more category specific words like "warning", "earthquake", "killed", "people". The original sentence only contains words such as "death", "buried". The new sentence thus offers more information content.

Generic:
— *The death toll could rise as thousands are still buried in debris and many are reported missing.*
— *Therefore, the relevant sectors and personnel should pay attention to disaster prevention.*
SWING:
\+ *Chinese authorities did not detect any warning signs ahead of Monday's earthquake that killed more than* ***8,600 people****.*
\+ *Xinhua said* ***8,533 people*** *had died in Sichuan alone, citing the local government.*

Figure 3: Difference in summaries for the topic "Earthquake in Sichuan", from the category *Accidents*.

When we compute CSI scores for sentences, one shortcoming is that we do not look at whether sentences have redundant category-specific information and whether all aspects of the category are covered by the selected sentences. For example, we observed that the second replaced sentence repeats the information already found in previous sentences of the summary. However it still gets selected into the summary due to the presence of more category specific words.

In the future, we plan to use category specific statistics in a more organized way to remove category-specific redundancy (akin to MMR) and to include all aspects of information in the summary.

Numerical information in a topic, such as casualties, temporal markers, monetary damages can also conflict within documents in a set on a topic, as they are compiled by different sources and at different points of time. For example, the number of casualties (**bolded** in summaries) is specified as 8,533 and more than 8,600 in different sentences from different sources. While any of these sentences could be selected into a summary due to similar content words, the corresponding model summary has only the most updated information (12,000 people). As a result the evaluation scores are dropped although the summarizer picks an informative sentence. This highlights the need to normalize such numerical information in the summaries which are important in categories like accidents and attacks where quantitative information is key.

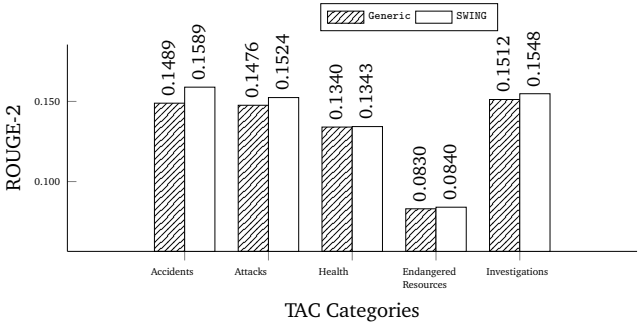


Figure 4: ROUGE-2 scores for each category for `Generic` and `SWING`.

We further observed that the difficulty in summarizing a topic may vary by category. We show ROUGE-2 performance by category in Figure 4, revealing that the topics in *Health* and *Endangered Resources* are the most difficult to summarize. We believe that the larger presence of subjective aspects (*How, Why, Threats*) in both of these categories increases the difficulty for automatic summarizers to recognize relevant information. The topics in the other three categories are easier to summarize: we note that the improvements on *Accidents* and *Attacks* with the CSI features are more pronounced than in the remaining categories. When we look at the aspects defined by TAC for both *Accidents* and *Attacks*, we notice that seven of their aspects overlap, as shown in Table 9. This suggests that the more general aspects a category has, the easier it is to compute its category-specific information. In our future work, we plan to look at how we can utilize general versus specific aspects to improve our model of CSI.

Conclusion

We have shown that using category-specific information (CSI) can significantly improve the performance of topic oriented summaries. We model CSI by creating two features: category relevance score (CRS), an intra-category measure; and category KL-divergence score (CKLD), an inter-category measure. Simple to compute and requiring no external knowledge or corpus, the combined use of both CRS and CKLD significantly improved automated ROUGE scores, leading to a basic extractive summarization system that leads the state-of-the-art.

Category	Aspects
Accidents and Natural Disasters	WHAT, WHEN, WHERE, WHY, WHO_AFFECTED, DAMAGES, COUNTERMEASURES
Attacks	WHAT, WHEN, WHERE, PERPETRATORS, WHY, WHO_AFFECTED, DAMAGES, COUNTERMEASURES

Table 9: Aspects for categories *Accidents* and *Attacks* defined in TAC. Seven aspects overlap in these two categories.

To probe more deeply, we assessed how to improve CSI features by limiting its calculation to word occurrences that occur within NP chunks. We also showed that automatically acquired category information (through clustering) still yields improved results, even when the artificially induced categories are noisy. Finally we performed a micro-analysis of the effect of CSI, studying the changes in sentence selection in the test dataset. This process showed that the incorporation of CSI changed selection selection significantly. The analysis also yielded insights about future directions for extractive sentence selection.

The use of CSI can be incorporated with sophisticated sentence post-processing that is a focus of current summarization research. As such, we see CSI as a foundational contribution that we urge other summarization platforms to adopt. To aid this adoption, we have open-sourced our package for the research community to use[3].

Acknowledgements

We like to thank Dr. Jian Su for her help with the data sets that we have used as part of our work. This research is supported by the Singapore National Research Foundation under its International Research Centre @ Singapore Funding Initiative and administered by the IDM Programme Office.

References

Aker, A. and Gaizauskas, R. (2009). Summary Generation for Toponym-referenced Images using Object Type Language Models. In *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP)*, pages 6–11.

Bysani, P., Reddy, V. B., and Varma, V. (2009). Modeling Novelty and Feature Combination using Support Vector Regression for Update Summarization. In *Proceedings of the 7th International Conference on Natural Language Processing (ICON)*.

Carbonell, J. and Goldstein, J. (1998). The Use of MMR, Diversity-based Reranking for Reordering Documents and Producing Summaries. In *Proceedings of the 21st Annual International ACM Conference on Research and Development in Information Retrieval (SIGIR)*, pages 335–336.

Conroy, J. M., Schlesinger, J. D., Kubina, J., Rankel, P. A., and O'Leary, D. P. (2011). CLASSY 2011 at TAC: Guided and Multi-lingual Summaries and Evaluation Metrics. In *Proceedings of the Text Analysis Conference (TAC)*.

Conroy, J. M., Schlesinger, J. D., Schlesinger, J. D., and O'Leary, D. P. (2010). CLASSY 2010: Summarization and Metrics. In *Proceedings of the Text Analysis Conference (TAC)*.

[3] `http://wing.comp.nus.edu.sg/downloads/swing/`

Daumé III, H. and Marcu, D. (2006). Bayesian query-focused summarization. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th Annual Meeting of the Association for Computational Linguistics*, pages 305–312.

Díaz, A. and Gervás, P. (2007). User-model Based Personalized Summarization. *Information Processing & Management*, 43(6):1715 – 1734.

Dunning, T. (1993). Accurate Methods for the Statistics of Surprise and Coincidence. *Computational Linguistics*, 19(1):61–74.

Edmundson, H. P. (1969). New Methods in Automatic Extracting. *Journal of ACM*, 16:264–285.

Gunn, S. (1998). Support Vector Machines for Classification and Regression. *ISIS Technical Report*, 14.

Haghighi, A. and Vanderwende, L. (2009). Exploring Content Models for Multi-document Summarization. In *Proceedings of Human Language Technologies: The Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL-HLT)*, pages 362–370.

Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., and Witten, I. H. (2009). The WEKA Data Mining Software: An Update. *ACM SIGKDD Explorations Newsletter*, 11:10–18.

Lin, C. (2004). Rouge: A Package for Automatic Evaluation of Summaries. In *Text Summarization Branches Out: Proceedings of the ACL-2004 Workshop*, pages 74–81.

Lin, C. and Hovy, E. (2003). Automatic Evaluation of Summaries Using N-gram Co-occurrence Statistics. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology (NAACL-HLT)*, pages 71–78.

Nenkova, A., Passonneau, R., and Mckeown, K. (2007). The Pyramid Method: Incorporating Human Content Selection Variation in Summarization Evaluation. *ACM Transactions on Speech and Language Processing (TSLP)*, 4(2).

Ng, J. P., Bysani, P., Lin, Z., Kan, M. Y., and Tan, C. L. (2011). SWING: Exploiting Category Specific Information for Guided Summarization. In *Proceedings of the Text Analysis Conference (TAC)*.

Owczarzak, K. and Dang, H. T. (2010). Overview of the TAC 2010 Summarization Track. In *Proceedings of the Text Analysis Conference (TAC)*.

Owczarzak, K. and Dang, H. T. (2011). Overview of the TAC 2011 Summarization Track: Guided Task and AESOP Task. In *Proceedings of the Text Analysis Conference (TAC)*.

Radev, D. R., Jing, H., Sty, M., and Tam, D. (2004). Centroid-based Summarization of Multiple Documents. *Information Processing & Management*, 40:919–938.

Schilder, F. and Kondadadi, R. (2008). FastSum: Fast and Accurate Query-based Multi-document Summarization. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, pages 205–208.

Steinberger, J., Tanev, H., Kabadjov, M., and Steinberger, R. (2010). JRC's Participation in the Guided Summarization Task at TAC 2010. In *Proceedings of the Text Analysis Conference (TAC)*.

Varma, V., Bysani, P., Reddy, K., Reddy, V. B., Reddy, V. B., Kovelamudi, S., Vaddepally, S. R., Nanduri, R., N, K. K., Gsk, S., and Pingali, P. (2010). IIIT Hyderabad in Guided Summarization and Knowledge Base Population. In *Proceedings of the Text Analysis Conference (TAC)*.

Zhang, R., Ouyang, Y., and Li, W. (2011). Guided Summarization with Aspect Recognition. In *Proceedings of the Text Analysis Conference (TAC)*.

Improved Temporal Relation Classification using Dependency Parses and Selective Crowdsourced Annotations

Jun − Ping NG Min − Yen KAN
National University of Singapore, 13 Computing Drive, Singapore 117417
{junping,kanmy}@comp.nus.edu.sg

ABSTRACT
We study the problem of classifying the temporal relationship between events and time expressions in text. In contrast to previous methods that require extensive feature engineering, our approach is simple, relying only on a measure of parse tree similarity. Our method generates such tree similarity values using dependency parses as input to a convolution kernel. The resulting system outperforms the current state-of-the-art.

To further improve classifier performance, we can obtain more annotated data. Rather than rely on expert annotation, we assess the feasibility of acquiring annotations through crowdsourcing. We show that quality temporal relationship annotation can be crowdsourced from novices. By leveraging the problem structure of temporal relation classification, we can selectively acquire annotations on problem instances that we assess as more difficult. Employing this annotation strategy allows us to achieve a classification accuracy of 73.2%, a statistically significant improvement of 8.6% over the previous state-of-the-art, while trimming annotation efforts by up to 37%.

Finally, as we believe that access to sufficient training data is a significant barrier to current temporal relationship classification, we plan to share our collected data with the research community to promote benchmarking and comparative studies.

KEYWORDS: temporal relations, information extraction, crowdsourcing, convolution kernels, clause structure, dependency parsing.

Proceedings of COLING 2012: Technical Papers, pages 2109–2124,
COLING 2012, Mumbai, December 2012.

1 Introduction

We study the problem of temporal information extraction; specifically, we seek to establish when an event has taken place relative to a time expression. We refer to this task as *event-temporal* relationship classification. We adopt the same definitions for event and time expressions used in TimeML[1]. Consider the sentence[2]:

> Two top aides to Netanyahu , political adviser Uzi Arad and Cabinet Secretary Danny Naveh, left for Europe on *Sunday* , apparently to investigate the Syrian issue , the newspaper said. (1)

In Sentence 1 "*Sunday*" is a time expression, while the various underlined words such as "said" are events. A pictorial summary of the timeline of the events as they are described in the sentence is shown in Figure 1. The time expression "*Sunday*" is marked on the timeline to represent an instance in time. The spans of the arrows representing each of the events denote the temporal relation between the three events and the time expression. In this case "left" happens in the same time span denoted by "*Sunday*". The other two events (i.e. "investigate" and "said") do not coincide with the "*Sunday*" marker and take place after the time span denoted by "*Sunday*".

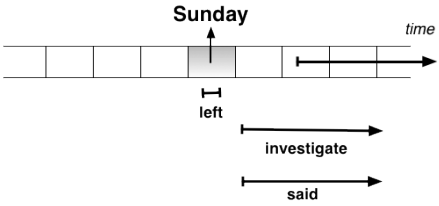


Figure 1: Temporal relationships between events and a time expression in Sentence 1.

Knowing such temporal relations are useful for downstream natural language processing applications. For example, Harabagiu and Bejan (2005) attempt to identify relationships between entities and time expressions to improve their question answering system. They combine knowledge of the relations between answer candidates and temporal expressions with semantic inference to derive exact answers to temporal questions. Temporal relations are also a recent focus in information extraction. In the temporal slot-filling task of the Knowledge Population Track featured in the Text Analysis Conference (Ji et al., 2011), the shared task required systems to add temporal information to extracted information. Such information would enable inference on such knowledge bases to be extended to event ordering and perhaps help to identify temporal contradictions.

Recognizing the value of temporal relations, the community has organized shared tasks to spur research efforts in this area. In our work, we adopt the community-standard TempEval-2 task definition and dataset (Verhagen et al., 2010). This dataset consists of 959 training instances for the *event-temporal* relationship classification task, and a testing set of 138 instances. A summary of the various *event-temporal* relationships annotated in the dataset is shown in Table 1.

[1] `http://www.timeml.org`.
[2] Extracted from document APW19980301.0720 of the TempEval-2 dataset.

Event-Temporal Relationship	Relative temporal ordering
OVERLAP	Event happens *within* time expression
BEFORE	Event happens *before* time expression
AFTER	Event happens *after* time expression
BEFORE-OR-OVERLAP	Event happens either *before* or *within* time expression
OVERLAP-OR-AFTER	Event happens either *within* or *after* time expression
VAGUE	Unable to assign a relation

Table 1: Event-temporal relationships in the TempEval-2 dataset.

State-of-the-art approaches to this problem make extensive use of careful feature engineering to model both surface lexical cues as well as information about the syntactic relations between parts of the sentences, focusing on the words that comprise the events and time expressions. Both feature types manifest a wide array of values, making data sparsity a concern. Current state-of-the-art systems achieve only about 65% accuracy, where the low performance has indeed been partially attributed to data sparsity (Yoshikawa et al., 2009).

There are several techniques to address sparse data. We adopt two in this paper, both of which lead to improved performance. First, we simplify the input features and classification methodology. Specifically, we use a support vector machine provisioned with only two features: dependency parse similarities of the time expression and the path from the event to time expressions. By limiting our input features, we obtain denser judgments which enhance the quality of the decision boundary. Second, we can enlarge our annotated dataset. To this end, we exploit annotations obtained from novices via crowdsourcing. We show that augmenting the TempEval-2 dataset with crowdsourced information improves classification performance slightly. However, we can further boost performance by selectively acquiring annotations for difficult and under-represented categories of instances. Pairing both techniques together allow us to achieve an 8.6% improvement in accuracy over the current state-of-the-art, which is statistically significant.

After first reviewing the related work, the remainder of this paper sequentially details these two key contributions of our temporal classification work. In the first half, we detail our approach for temporal relation classification. In the second half, we discuss our approach to selectively annotate important problem instances via crowdsourcing.

2 Related Work

There has been a good amount of research on temporal relationship classification, culminating in recent shared tasks in TempEval-1 (Verhagen et al., 2007) and TempEval-2 (Verhagen et al., 2010). In fact, the *event-temporal* relationship classification problem we focus on is exactly Task C in TempEval-2. Other temporal information extraction tasks include *event-event* relationship classification, which involves determining which of two events took place earlier. Top performing teams attempting Task C made use of supervised machine-learning, including conditional random fields (Kolya et al., 2010), Markov logic (UzZaman and Allen, 2010; Ha et al., 2010), and maximum entropy classification (Derczynski and Gaizauskas, 2010). Enlarging the problem, Yoshikawa et al. (2009) proposed building an inference model which jointly predicts *event-temporal*, *temporal-document creation time* and *event-event* temporal

relationships within a given sentence. While the system performed comparably to other systems in TempEval-1, such a joint model greatly enlarges the problem complexity as the search space is a Cartesian product of the three separate problems.

All of the above systems employed similar features which we categorize into three feature types: 1) lexical cues, such as signal words and part-of-speech tags, 2) context, including the attributes of the event and time expressions and 3) the grammatical structure of the sentences, obtained by processing automatic parses. Possibly since they use similar features, the top performing systems all perform similarly, achieving around 65% accuracy in their judgments. We note that the feature types employed can take on many different values and thus represent a large potential feature space. Given that the TempEval-2 dataset comprises of only 959 instances, we believe the learned models suffer from sparse data, and can improve from the incorporation of additional data.

Many researchers have looked at ways to reduce the annotation effort required to build a sizable temporal dataset. Setzer et al. (2003) proposed making use of a set of inference rules to identify useful unlabeled instances for annotation. If the equivalence of an unlabeled instance to an already labeled instance can be shown, the unlabeled instance is discarded. Similarly, we are proposing a systematic way in which unlabeled instances can be left out of the data acquisition process. But instead of making use of pre-identified inference rules, our proposal is fully automatic and exploits the dependency parse structure of unlabeled instances without prior curation of hand-written rules.

In a bid to balance the cost of data acquisition with the need for more annotated data, the organizers of TempEval-3 (UzZaman et al., 2012), to be held at SemEval-2013 propose augmenting a manually annotated dataset of 100K word tokens with around 500K semi-automatically annotated instances. At the time of this writing, the semi-automatically annotated instances have yet to be released.

In recent times, crowdsourcing has been a popular method employed to collect linguistic annotations. Munro et al. (2010), for example, lists some linguistic projects which have benefitted from crowdsourcing. To the best of our knowledge however, there is no existing literature on the adaptation of crowdsourcing for temporal datasets.

3 Convolution Kernels using Dependency Parse Trees

What features and model should we adopt to create a robust temporal relation classifier that takes full advantage of the small amount of annotated training data?

Given the sparsity of values in the features used by previous work, our approach is to reduce the dimensionality of the input feature space, to make the feature space more dense. While we concur with the prior work that the relation between the events and time expression are important, we feel that the current approaches to model this information can be simplified. Such structural relations are perhaps best captured by parse tree paths. Intuitively, if the parse tree of a testing instance is similar to one from a training instance, we may guess that the training instance's type of temporal relation would also hold for the previously unseen testing instance.

To compute the similarity between two parse trees, a typical approach is to engineer flat representations of the parse structure through feature engineering. This process is time consuming and often requires good knowledge of the problem structure to decide which

features are more discriminative than others. Convolution kernels (Collins and Duffy, 2001) on the other hand model this form of structure similarity well. Convolution tree kernels take as input tree structures and calculate a degree of similarity between the two trees. In its simplest form, similarity is computed by recursively counting the number of identical subtrees that appear in both input instances. With this structural similarity measure, we can do away with the need to "flatten" the structure with hand-devised representations. For these reasons, we use a support vector machine (SVM) as our supervised classification model, adopting a convolution kernel as its kernel function (Moschitti, 2006).

A second issue is then to decide the type of parse tree to employ. Most previous work centered around the use of constituent grammar parses. In fact, the only other work that also adopts a convolution kernel approach to temporal relation classification (Mirroshandel et al., 2011) also uses constituent grammar parses. However, as constituent parses create internal phrasal nodes for every semantic constituent, such parse trees are often deep and overly detailed. Paths in such trees are fine-grained, capturing nuances (*e.g.*, intervening finite verb phrase nodes), and as such, may not generalize well when used to compute tree or path similarities.

To remedy this, we employ dependency parses instead of constituent parses. Dependency parses are generally more compact than constituent grammar parses because they have no immediate phrasal nodes. This translates into a more compact feature space. Bearing in mind that one of the problems we are trying to overcome is that of data sparsity, we find that dependency parses are generally more useful than constituent parses for this task.

We compute two features based on dependency parses:

1. ***Dependency path from event to time expression.*** Starting from a full dependency parse of a sentence, we identify the vertices representing the event and time expressions. We locate the shortest path between these two vertices and use this sub-tree as a feature.
2. ***Dependency parse of the time expression.*** Time expressions can range from single word tokens to multi-word phrases, with vastly different semantics. For example, *Friday*, *last Friday* and *next Friday* convey very different meanings. To capture this, we extract a sub-tree from the full dependency parse consisting of all the vertices and edges related to the time expression and use this as a feature.

3.1 Evaluation

How does our simplified, two input feature SVM compare against the prior work on the TempEval-2 dataset?

Table 2 gives the performance of our classifier which we will refer to as `SVMConvoDep` vis-a-vis the top performing systems in TempEval-2. We adopt the same *accuracy* metric used in the TempEval-2 task, which is the number of correct answers divided by the number of answers. We also list macro-averaged precision, recall, and F_1 measures. We find macro-averaged measures more appropriate than micro-averaged ones for this task. As will be explained later, the dataset has a skewed distribution of labels, with `OVERLAP` forming the majority label. Micro-averaged measures give more weight to the most common label in such skewed datasets, but we feel that it is important for systems to be able to perform well across all the temporal labels.

To the best of our knowledge, our classifier outperforms all previous temporal relation classifiers. While this performance gain is probably not statistically significant (as we have no access to the participating systems' individual judgments, it is impossible to check for statistical significance),

we feel that these are impressive results as our classification input is decidedly simple (just two subtrees derived from dependency parses as features).

System	Accuracy (%)	Precision	Recall	F_1
SVMConvoDep	67.4	0.828	0.512	0.523
TRIOS	65.0	Not Available		
JU_CSE	63.0			
NCSU-indi	63.0			
NCSU-joint	63.0			
TRIPS	63.0			
USFD2	63.0			

Table 2: Performance on TempEval-2 testing set. Results for TempEval-2 systems are cited from Verhagen et al. (2010).

4 Enlarging the Dataset through Crowdsourcing

Our next proposal to solve the data sparsity problem is to enlarge the dataset available. Developing a large, suitably annotated dataset is expensive – both in terms of time and monetary cost. Recent work in natural language processing suggests that crowdsourcing annotations from the untrained public can provide annotated data at similar annotation quality as expert annotators, but for a fraction of the cost (Sheng et al., 2008; Hsueh et al., 2009). However we are unsure if these results are applicable to a temporal dataset which is inherently complex (Setzer and Gaizauskas, 2000) and require a better understanding of the target language.

To find out, we set up a crowdsourcing task on CrowdFlower. We chose CrowdFlower as our crowdsourcing platform, as it has access to a large user base (it uses Amazon Mechanical Turk to find workers), but adds an extra validation layer to attempt to address quality concerns, as this has been an issue in many applications of crowdsourcing in natural language annotation tasks (Mason and Watts, 2009; Callison-Burch and Dredze, 2010).

4.1 Task Setup

Each annotation instance consists of a single sentence. We pre-process the sentence to highlight one event expression and one time expression found within it. Annotators were tasked to choose from five (`OVERLAP`, `AFTER`, `BEFORE`, `NOT-RELATED`, `BAD-SENTENCE`) possible temporal relationships between the marked event and time expression. An additional choice for `BAD-SENTENCE` was included to allow annotators to indicate if there had been problems with our automatic pre-processing. Such instances (67 in our study) were discarded.

We required that at least 3 judgments be made for each annotation, and majority voting was then used to decide on a final label for each annotation. To ensure the quality of the judgments we had obtained, we made use of a validation facility provided by CrowdFlower. Pre-annotated gold instances were mixed together with unlabeled instances. These gold instances were used to validate the annotations made by each annotator. Annotators were not informed which were the gold instances. During the annotation process, annotators who failed to label these gold instances correctly were stopped from proceeding with the task, and the annotations they made were discarded.

Raw Data. To ready a set of unlabeled instances for annotation, we crawled news articles on several news web sites, including *Wall Street Journal*, *New York Times*, *CNN*, and *Channel News*

Asia from 2 June to 8 July 2012. We made use of the sentence splitting module from the Apache OpenNLP[3] library to obtain individual sentences from these news articles. We built a CRF-based event and time expression extractor (`CRFEventTimexExt`) which is able to automatically identify event and time expressions in each of the collected sentences. For event extraction, we made use of part-of-speech (POS) tags as a feature. For time expression extraction, we made use of a compiled lexicon of the days of a week and months of a year on top of POS tags. The performance of our extractor is compared against top participating systems in TempEval-2 in Table 3. We include only the results of systems which are able extract both event and time expressions as it is not required for systems to be able to do both in TempEval-2.

System	Event			Time		
	Precision	Recall	F_1	Precision	Recall	F_1
`CRFEventTimexExt`	0.81	0.82	0.82	0.76	0.62	0.68
Edinburgh	0.75	0.85	0.80	0.85	0.82	0.84
TIPSem	0.81	0.86	0.83	0.92	0.80	0.85
TRIPS	0.55	0.88	0.68	0.85	0.85	0.85
TRIOS	0.80	0.74	0.77	0.85	0.85	0.85

Table 3: Comparison of event and time expression identification with TempEval-2 systems. Performance of TempEval-2 systems are cited from Verhagen et al. (2010).

We are able to turn in competitive F_1 scores for event expression extraction, but less so for time expression extraction due mainly to poor recall. We find the precision scores of 0.81 and 0.76 for event and time expression identification respectively sufficiently high as this is part of pre-processing. Incorrectly identified event and expressions can be labeled as so by annotators, and these precision scores will minimize the occurrences of such mistakes. To address the problems with recall, we can always collect more sentences to increase the number of time expressions we have for annotation.

4.2 Experiments

We first want to test how a similarly sized crowdsourced annotation compares with the manual, expert annotations from TempEval-2. For this reason, we extracted an initial batch of 1,000 tuples (close in size to the TempEval-2's training data of 959 instances) of event and time expressions from the sentences we had crawled. We collected annotations for each of the tuples on CrowdFlower. We refer to this dataset as `d-1000`. We compared the distribution of the labels within this collected set with the TempEval-2 training and testing sets in Table 4.

The skewed distribution in the TempEval-2 datasets are similarly observed in the collected dataset, with the `OVERLAP` label taking up more than 50% of the whole dataset.

We trained a new classifier using the same features as `SVMConvoDep` with the crowdsourced annotations `d-1000`. The performance of this trained classifier, `CF-1000`, on the TempEval-2 testing set is shown in Table 5. In the table we have also included macro-averaged precision, recall and F_1 measures.

`CF-1000` did not do as well as `SVMConvoDep` in terms of accuracy. While its F_1 score is slightly higher, the difference is not statistically significant. This is not surprising for two reasons. First

[3] `http://opennlp.apache.org/`.

Dataset	Size	Distribution of label (%)			
		OVERLAP	BEFORE	AFTER	Others
TempEval-2 training set	959	53.8	18.0	23.0	5.2
TempEval-2 testing set	138	55.1	14.5	19.6	10.9
d-1000	1000	70.0	12.2	17.8	0.0

Table 4: Distribution of labels in different datasets.

the annotators recruited for the annotation task are not domain experts, and we can expect mistakes in the annotations obtained. Second, SVMConvoDep is trained on the TempEval-2 training set. This dataset is prepared in similar fashion together with the TempEval-2 testing set we are evaluating on. We will expect the two datasets to share more similar attributes and characteristics than the d-1000 set that we had collected. The content of the TempEval-2 datasets and ours span a different time period, are sourced from different sources, and possibly drawn from different domains and categories.

We tried to combine the TempEval-2 training set with d-1000. With this combined dataset, we trained another classifier CF-1000+TE. The performance of this classifier is also reported in Table 5. We get improved results in both accuracy and F_1 scores. This improvement is significant, with $p < 0.05$ when tested with the paired Student's t-test.

System	Accuracy (%)	Precision	Recall	F_1
SVMConvoDep	67.4	0.828	0.512	0.523
CF-1000	65.2	0.578	0.535	0.525
CF-1000+TE	71.7	0.726	0.598	0.615

Table 5: Classifier performance on TempEval-2 testing set.

There are two important conclusions that we can draw from these results. First, the difference in performance between SVMConvoDep and CF-1000 is slight. So while the the novices recruited via crowdsourcing may not have been domain experts, they are able to generate a dataset that is comparable to an expert-curated one.

Then, the improvement to the performance of CF-1000+TE shows us that despite the possible differences in time span, source, and categorization of d-1000 from the TempEval-2 dataset as we have suggested, there is value to the crowdsourced dataset. Doubling the amount of training data available by putting d-1000 and the TempEval-2 training set together rewards performance.

5 A Smarter Way

With crowdsourcing, the costs associated with generating temporal datasets are lowered. Yet we want to do this efficiently with minimal wastage. In this section, we explain how we can selectively acquire annotations to reduce the costs involved without affecting the efficacy of the data collected.

Motivation. We recognize that it is often not easy to decide on the relationship between an event and time expression. For example let us go back and take a look at Sentence 1.

Within the sentence there are several event expressions. For the sake of this discussion we focus on just two of them: 1) "left", and 2) "said". One immediate observation is that time

expressions are commonly adjuncts (in this case, a prepositional phrase (PP)) that attach to a verb phrase (VP). This implies that time expressions often directly modify only the head it is attached to. In such cases, it is usually easy to identify the temporal relationship between the event and time expression. Looking at "left" and "*Sunday*", it is quite straightforward to determine that they take place within the same time span.

However, it gets significantly more difficult when we look at "said" and "*Sunday*". We have to read through more of the sentence to build up an understanding of the relation between "left", "investigate", and "said" before we can conclude that "said" takes place after "*Sunday*".

Our key insight to more efficient data acquisition is to leverage this observation. We postulate that it is more useful if we can gather annotations for complex instances instead of easier ones.

Definitions. We define a few terms that we will use in the rest of this paper. For the *event-temporal* relationship classification problem, the input is a collection $\mathbb{S}$ of sentences. Each sentence s is composed of one or more word tokens, i.e. $s = w_1 w_2 \ldots$. Let s^* be the set of all possible subsequences of s. For each s, we can define a set of unigram word tokens $\epsilon_s = \{w, w \in s\}$ that are event expressions. We can also define a set $\Theta_s = \{\theta, \theta \in s^*\}$ which includes all the time expressions within s. The problem then is to define some function $f : s, e, t \rightarrow R, s \in \mathbb{S}, e \in \epsilon_s, t \in \Theta_s$ where R is the temporal relationship between the event expression e and time expression t.

Building on our observation, we want to be able to partition a set of unlabeled instances so that the more complex instances are separated from the easier ones. We propose building such a partitioning scheme around the ordering of the elements in ϵ_s given a time expression $t \in \Theta_s$.

To do this, we first build a dependency parse[4] of the input sentence s, where each $w_i \in s$ forms a vertex within the dependency parse tree. A portion of the dependency parse tree for Sentence 1 is shown in Figure 2.

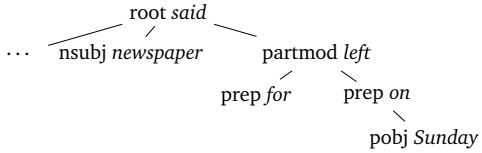


Figure 2: Excerpt of the dependency parse for Sentence 1.

In the figure, the governing dependency relation of a word is shown as a vertex. The associated word is also shown in *italics* to illustrate which part of the sentence this parse is about.

For a given dependency parse and a target time expression t, we can define a total order O_t on ϵ_s. O_t is defined by arranging each $w \in \epsilon_s$ in ascending order of their respective distances from the time expression vertex. Distance in this case is defined as the number of edges that needs to be traversed to reach the time expression vertex.

Imposing the total order O_t on ϵ_s gives us a totally ordered set, i.e. $(O_t, \epsilon_s) = \{e_0, e_1, \ldots\}$. From this, for every input sentence s and its associated event (e) and time expressions (t), we can place the tuple $< s, e, t >$ into a partition $\mathscr{P}_i$, where i is the index of e within (O_t, ϵ_s).

[4]We make use of Stanford dependencies (Klein and Manning, 2003).

Referring to Sentence 1 as an illustration, we thus place <"left", "*Sunday*"> in $\mathscr{P}_0$ because "left" is the nearest event expression to "*Sunday*" in the dependency parse in Figure 2. <"said","*Sunday*"> will be placed in $\mathscr{P}_1$ as "said" is the next nearest event expression. For convenience, we will also be referring to $\mathscr{P}_i$ as the set of Level-*i* instances.

This partitioning scheme is premised on the intuition that it requires more effort to understand the temporal relationship between event and time expressions which are both structurally and semantically further away from each other. Higher level instances thus should be more complex than their lower level peers.

Following this scheme, we separated the TempEval-2 testing set into different partitions based on our prescribed methodology. A breakdown of the performance of `SVMConvoDep` on each of the partitions $\mathscr{P}_i$ is shown in Table 6. Accuracy drops steadily from Level-0 instances to Level-2 instances. This provides support for our intuition that higher level instances are harder to classify accurately.

Given that there are only 10 Level-3 instances and 1 Level-4 instance, the variance in measurements can potentially be very wide. There are too few Level-3 and Level-4 instances for their results to be analyzed reliably.

System	Accuracy (%)				
	Level-0 (59)	Level-1 (47)	Level-2 (21)	Level-3 (10)	Level-4 (1)
`SVMConvoDep`	84.5	66.0	42.9	30.0	100.0

Table 6: Breakdown of performance on partitions of TempEval-2 testing data. The number of instances for each partition is indicated in parentheses.

Given the high prediction accuracy on Level-0 instances, we argue that it is not necessary to obtain more annotations for them. Instead we should focus on the higher level instances. By opting not to annotate additional Level-0 instances, substantial cost savings can be made, as seen from Table 7. In the table we present a breakdown of the relative size of each partition to its entire dataset. `d-full` is a set of 8,851 tuples of event and time expressions that we have extracted from the same set of sentences we had crawled earlier. As seen from the table, Level-0 instances consistently form a large part of the various datasets. Not annotating Level-0 instances will directly lead to a cost savings of at least 37%.

Dataset	Relative size of partition (%)			
	Level-0	Level-1	Level-2	Others
TempEval-2 Training Set	40.9	35.2	15.1	8.8
TempEval-2 Testing Set	41.4	34.3	15.7	8.6
`d-full`	37.0	34.3	17.5	11.2

Table 7: Breakdown of partition sizes of different datasets.

5.1 Experiments

To study the effectiveness of our recommendations, we collected annotations for all the tuples in `d-full` via CrowdFlower in a similar way to what we had done earlier. From this `d-full` collection of 8,851 annotations, we removed all Level-0 instances to create a subset of 5,576 annotations which we will call `d-nolevel0`.

Using d-full and d-nolevel0, we trained two new classifiers CF-Full and CF-NoLevel0 respectively. Table 8 shows the performance of these two new classifiers with that of SVMConvoDep when tested with the TempEval-2 testing set.

From the results, we note that CF-NoLevel0 is able to deliver a significant performance gain of about 8.6% over SVMConvoDep ($p < 0.05$) even though it only made use of part of the full dataset we had collected. Further, it is able to match the performance of CF-Full which was trained over all collected instances (d-full). The performances of CF-NoLevel0 and CF-Full are not significantly different.

System	Accuracy (%)	Precision	Recall	F_1
SVMConvoDep	67.4	0.828	0.512	0.523
CF-NoLevel0	73.2	0.659	0.643	0.639
CF-Full	73.2	0.660	0.647	0.641

Table 8: Accuracy on TempEval-2 testing set.

These results are illuminating. We see that our proposed partitioning scheme is able to reliably identify unlabeled instances that will not be able to contribute to better classifier performance. By focusing our annotation effort only on more complex instances, we bring down data acquisition costs by a large amount (37%) while retaining classifier performance.

6 Analysis and Discussion

6.1 Selective Data Acquisition

We try to probe deeper into the performance of both CF-NoLevel0 and CF-Full. We want insight into why the former works as well as the latter, despite the large reduction in training data.

Table 9 shows a more concise breakdown of the precision, recall and F_1 scores of both classifiers when tested with the TempEval-2 testing set. We see that both classifiers turn in similar performance across all three labels. Performance for the OVERLAP label is better, possibly because there are more training instances for it in the training dataset.

Classifier	OVERLAP			BEFORE			AFTER		
	P	R	F_1	P	R	F_1	P	R	F_1
CF-NoLevel0	0.72	0.96	0.82	0.56	0.45	0.5	0.70	0.52	0.60
CF-Full	0.72	0.95	0.81	0.57	0.40	0.47	0.70	0.60	0.64

Table 9: Performance measures on TempEval-2 testing set broken down by individual labels.

Table 10 illustrates the distribution of the AFTER and BEFORE labels in the first three partitions of the test data. The labels make up a larger part of the annotations at Level-1 and Level-2 than at Level-0.

Putting the pieces of the puzzle together, it is likely that the training instances in Level-0 are more useful for the OVERLAP label than the other two. CF-NoLevel0 is already able to classify these instances quite well, so not having access to Level-0 training instances does not affect it adversely.

We break down the performance of the classifiers on each partition of the test data in Figure 3. As expected, without access to Level-0 training instances, CF-NoLevel0 is slightly outperformed

Label	Distribution of labels (%)		
	Level-0	Level-1	Level-2
AFTER	10.1	21.2	23.6
BEFORE	5.1	13.7	16.1

Table 10: Distribution of labels in each partition.

(but not statistically so) by `CF-Full` for Level-0 test instances. Interestingly, the performance of `CF-NoLevel0` for Level-2 and Level-3 instances are also behind that for `CF-Full`. This suggests that having additional Level-0 training instances can have an effect on classifier performance for Level-2 and Level-3 test instances. In future work we will like to investigate why this is the case.

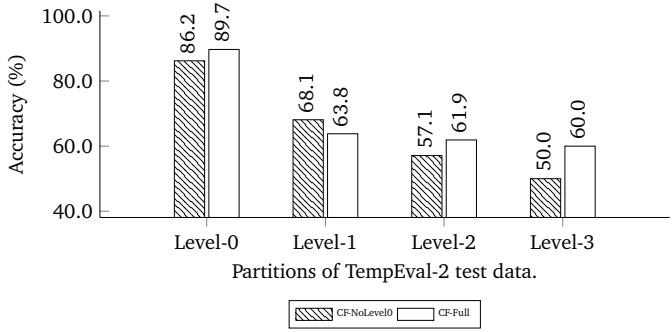


Figure 3: Breakdown of performance across different partitions.

6.2 Common Errors

We are also interested to investigate what more can be done to push system performance towards its upper-bound. Typically, such a theoretic upper-bound is derived from the inter-annotator agreement for the training set. The inter-annotator agreement for the annotations we have collected for `d-full` is 78.8%. Considering that the participants of the crowdsourcing exercise are not domain experts, the actual upper-bound could be higher. However this value is a good starting point for our analysis.

The best performing classifier we have presented in this work has an accuracy of 73.2%. There is still room for improvement towards our pessimistic upper-bound estimate. We study the misclassifications made, building a confusion matrix for `CF-NoLevel0`. Studying the matrix in Table 11, we observe two large clusters of errors.

Skewed dataset. First, `BEFORE` and `AFTER` instances are often mis-classified as `OVERLAP`. Reflecting on this, we believe the skewed training dataset where `OVERLAP` instances form a majority is one of the reasons why.

Considering the lower recall scores we get for `BEFORE` and `AFTER` labels, the mis-classifications are likely a direct result of a lack of suitable training instances within the training dataset to better identify `BEFORE` and `AFTER` instances.

Actual Label	Predicted label		
	OVERLAP	BEFORE	AFTER
OVERLAP	78	2	1
BEFORE	7	9	4
AFTER	13	0	14

Table 11: Confusion matrix for CF-NoLevel0.

Looking closer, we see that the performance on BEFORE instances is slightly lower than the performance for AFTER instances. We relate this to the smaller number of training instances available for the BEFORE label as seen from Table 10.

In future work it will be useful to verify if increasing the number of training instances available can help to improve the recall of the classifier for instances of these two labels, thereby eradicating this cluster of errors.

Feature design. The next major cause of mistakes is the misclassification of BEFORE instances as AFTER. Studying the mis-labeled instances, we realized this is because the features that we have used neglected to consider modal or copula modifications to the event expressions. For example, take a look at sentence 2[5]. The relationship between "added" and "*early November*" should have been BEFORE but was incorrectly classified as AFTER. The relevant portions of the dependency parse extracted as a feature for this instance is shown in Figure 4.

He added that final guidelines to be published in *early November* will determine whether the bank is in compliance . (2)

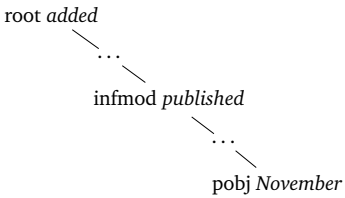


Figure 4: Dependency parse of Sentence 2.

The dependency parse feature that we extract is the shortest path between the vertices for "added" and "*November*". The key to interpreting the temporal relationship in this example however is the copula modifier "*to be*" in front of "published".

Without capturing these modifiers, the sentence will appear to read as "He added that final guidelines published in *early November* will determine . . . ". In this reading, "added" takes place AFTER the time span indicated by "*early November*", which explains the mis-classification by our classifier.

It will be useful following this analysis to study if including the auxiliary modifiers of event expressions into our parse features can help improve classifier performance.

[5]Extracted from document wsj_0527 of the TempEval-2 dataset.

Conclusion

In our study, we target the the main problem which we believe is hampering better performance for *event-temporal* relationship classification — lack of sufficient training data. We adopt a two-prong approach to tackle this problem: 1) by simplifying the feature space with the use of dependency parses as features to a convolution kernel support vector machine, and 2) by leveraging on crowdsourcing to get more data to support supervised machine learners.

We show that the classifier design we adopted is competitive when pitted against classifiers which make use of far more complex mechanics and features. With this as a starting point, we went on to expand the training data that is available for use via crowdsourcing. Despite the complexity of the annotation task, novice annotators are able to generate a dataset that helps improve classifier performance significantly.

Building on our insight of the clausal structure inherent to event and time expressions, we suggest an effective way to selectively acquire annotations. Our proposal reduces the amount of data to be annotated by up to 37% without sacrificing classifier performance. We achieved a classification accuracy of 73.2%, which represents a 8.6% improvement over our very competitive baseline.

Acknowledgments

This work is funded by the NExT Search Centre, which is supported by the Singapore National Research Foundation and Interactive Digital Media R&D Program Office, MDA under research grant (R-252-300-001-490).

References

Callison-Burch, C. and Dredze, M. (2010). Creating Speech and Language Data with Amazon's Mechanical Turk. In *Proceedings of the NAACL HLT 2010 Workshop on Creating Speech and Language Data with Amazon's Mechanical Turk*, pages 1–12.

Collins, M. and Duffy, N. (2001). Convolution Kernels for Natural Language. In *Proceedings of NIPS*, volume 14, pages 625–632.

Derczynski, L. and Gaizauskas, R. (2010). USFD2: Annotating Temporal Expressions and TLINKs for TempEval-2. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, pages 337–340.

Ha, E. Y., Baikadi, A., Licata, C., and Lester, J. C. (2010). NCSU: Modeling Temporal Relations with Markov Logic and Lexical Ontology. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, pages 341–344.

Harabagiu, S. and Bejan, C. A. (2005). Question Answering Based on Temporal Inference. In *Proceedings of the AAAI-2005 Workshop on Inference for Textual Question Answering*, pages 27–34.

Hsueh, P., Melville, P., and Sindhwani, V. (2009). Data Quality from Crowdsourcing: A study of Annotation Selection Criteria. In *Proceedings of the NAACL HLT 2009 Workshop on Active Learning for Natural Language Processing*, pages 27–35.

Ji, H., Grishman, R., and Dang, H. T. (2011). Overview of the TAC2011 Knowledge Base Population Track. In *Proceedings of the Text Analysis Conference (TAC)*.

Klein, D. and Manning, C. (2003). Accurate Unlexicalized Parsing. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics*, volume 1, pages 423–430.

Kolya, A. K., Ekbal, A., and Bandyopadhyay, S. (2010). JU_CSE_TEMP: A First Step towards Evaluating Events, Time Expressions and Temporal Relations. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, pages 345–350.

Mason, W. and Watts, D. (2009). Financial Incentives and the Performance of Crowds. In *Proceedings of the ACM SIGKDD Workshop on Human Computation*, pages 77–85.

Mirroshandel, S. A., Ghassem-Sani, G., and Khayyamian, M. (2011). Using Syntactic-Based Kernels for Classifying Temporal Relations. *Journal of Computer Science and Technology*, pages 68–80.

Moschitti, A. (2006). Making Tree Kernels Practical for Natural Language Learning. In *Proceedings of European Chapter of the Association for Computational Linguistics*, volume 6, pages 113–120.

Munro, R., Bethard, S., Kuperman, V., Lai, V., Melnick, R., Potts, C., Schnoebelen, T., and Tily, H. (2010). Crowdsourcing and Language Studies: The New Generation of Linguistic Data. In *Proceedings of the NAACL HLT 2010 Workshop on Creating Speech and Language Data with Amazon's Mechanical Turk*, pages 122–130.

Setzer, A. and Gaizauskas, R. (2000). Building a Temporally Annotated Corpus for Information Extraction. In *Proceedings of the LREC-2000 Workshop on Information Extraction Meets Corpus Linguistics*.

Setzer, A., Gaizauskas, R., and Hepple, M. (2003). Using Semantic Inferences for Temporal Annotation Comparison. In *Proceedings of the 4th International Workshop on Inference in Computational Semantics (ICoS-4)*.

Sheng, V., Provost, F., and Ipeirotis, P. (2008). Get another label? Improving data quality and data mining using multiple, noisy labelers. In *Proceeding of the 14th ACM International Conference on Knowledge Discovery and Data Mining*, pages 614–622.

UzZaman, N. and Allen, J. F. (2010). TRIPS and TRIOS System for TempEval-2: Extracting Temporal Information. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, pages 276–283.

UzZaman, N., Llorens, H., Allen, J., Derczynski, L., Verhagen, M., and Pustejovsky, J. (2012). TempEval-3: Evaluating Events, Time Expressions, and Temporal Relations. *ArXiv e-prints*.

Verhagen, M., Gaizauskas, R., Schilder, F., Hepple, M., Katz, G., and Pustejovsky, J. (2007). SemEval-2007 Task 15: TempEval Temporal Relation Identification. In *Proceedings of the 4th International Workshop on Semantic Evaluations*, pages 75–80.

Verhagen, M., Sauri, R., Caselli, T., and Pustejovsky, J. (2010). SemEval-2010 Task 13: TempEval-2. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, pages 57–62.

Yoshikawa, K., Riedel, S., Asahara, M., and Matsumoto, Y. (2009). Jointly Identifying Temporal Relations with Markov Logic. In *Proceedings of the 47th Annual Meeting of the Association for Computational Linguistics and the 4th International Joint Conference On Natural Language Processing of the AFNLP*, pages 405–413.

Accurate Unbounded Dependency Recovery using Generalized Categorial Grammars

Luan NGUYEN[1] *Marten VAN SCHIJNDEL*[2] *William SCHULER*[2]
(1) University of Minnesota, Dept. of Computer Science and Engineering, Minneapolis, MN
(2) The Ohio State University, Dept. of Linguistics, Columbus, OH
lnguyen@cs.umn.edu, vanschm@ling.osu.edu, schuler@ling.osu.edu

ABSTRACT
Accurate recovery of predicate-argument dependencies is vital for interpretation tasks like information extraction and question answering, and unbounded dependencies may account for a significant portion of the dependencies in any given text. This paper describes a categorial grammar which, like other categorial grammars, imposes a small, uniform, and easily learnable set of semantic composition operations based on functor-argument relations, but like HPSG, is generalized to limit the number of categories used to those needed to enforce grammatical constraints. The paper also describes a novel reannotation system used to map existing resources based on Government and Binding Theory, like the Penn Treebank, into this categorial representation. This grammar is evaluated on an existing unbounded dependency recovery task (Rimell et al., 2009; Nivre et al., 2010).

KEYWORDS: Grammar and formalisms, Semantics.

Proceedings of COLING 2012: Technical Papers, pages 2125–2140,
COLING 2012, Mumbai, December 2012.

1 Introduction

Accurate recovery of predicate-argument dependencies is vital for interpretation tasks like information extraction and question answering, and unbounded dependencies may account for a significant portion of the dependencies in any given text. Many current interpretation models are based on PCFGs, trained on syntactic annotations from the Penn Treebank (Marcus et al., 1993). These often recover dependencies as a post-process to parsing, and often are not able to retrieve unbounded dependencies if they are optimized on syntactic representations that leave these dependencies out.

Categorial grammars, on the other hand, have well-defined unbounded dependency representations based on functor-argument relations in a small and easily-learnable set of composition operations. Such grammars — in particular, Combinatory Categorial Grammar (CCG) (Steedman, 2000; Clark and Curran, 2007) — do well on unbounded dependency recovery tasks (Rimell et al., 2009) but not as well as models based on Head Driven Phrase-structure Grammar (HPSG) (Pollard and Sag, 1994; Miyao and Tsujii, 2005), given the same training. This may be attributed to implicit tradeoffs in many categorial frameworks that minimize the number of composition operations at the expense of large numbers of possible categories for each lexical item, which may lead to sparse data effects in training. HPSG models, in contrast, maintain a relatively large number of composition operations and a relatively small set of possible lexical categories, which are then used in a wider set of contexts.

Can categorial grammars, which have well-studied semantic representations and are well suited for interpretation, obtain better performance on a general unbounded dependency extraction task if it adopts an HPSG-like strategy of re-using types in various contexts? This paper describes a categorial grammar which, like HPSG, is generalized to limit the number of categories used to those needed to enforce grammatical constraints, but like other categorial grammars, imposes a small, uniform, and easily learnable set of semantic composition operations based on functor-argument relations. The paper also describes a novel reannotation system used to map existing resources based on Government and Binding Theory, like the Penn Treebank, into this categorial representation. This grammar is evaluated by training a state-of-the-art latent-variable parser on a version of the Penn Treebank reannotated into this generalized categorial grammar representation, and testing on an existing unbounded dependency recovery task (Rimell et al., 2009; Nivre et al., 2010).

The remainder of this paper is organized as follows. Section 2 describes related work on modeling unbounded dependencies and reannotation of syntax, Section 3 describes a generalized categorial grammar which provides transparent predicate-argument dependencies and generalizes similar categories across contexts, Section 4 describes how syntactically-annotated corpora can be reannotated into this framework, and Section 5 describes an evaluation of this system on an existing unbounded dependency recovery task.

2 Related Work

This section describes related work in unbounded dependencies and syntax reannotation.

2.1 Unbounded Dependencies

This paper describes a general-purpose reannotation system and its application to the specific task of reannotating local argument-passing treatments of long-distance dependencies in a generalized categorial grammar. Unbounded dependencies are dependencies between constituents

and points of attachment that have other constituents syntactically intervening. For example, the sentence *What does the First Amendment protect?* has a preposed constituent *what* that functions as a direct object of the transitive verb *protect*. The long distance between the source and destination of this type of dependency, paired with the relatively low probability of their occurrence in the language, and the fact that filler-gap annotations in syntactic resources such as the Penn Treebank are often stripped out, makes it very difficult for parsers to recognize this type of dependency correctly. While difficult to parse, this type of dependency is vital to the meaning of the sentence and of great importance in applications such as question answering and information extraction.

Rich grammar formalisms such as CCG (Steedman, 2000) have been shown to provide very accurate syntactic dependency extraction. This is thought to be because these formalisms explicitly model long-distance filler-gap dependencies inherent in relative clauses and interrogatives (Gildea and Hockenmaier, 2003). But formal systems like CCG, especially as reflected in reannotated Treebanks like CCGbank (Hockenmaier, 2003), comprise analyses of many phenomena: right node raising, light verbs, subject control, object control, coordinating conjunctions, etc., among which filler-gap constructions are just one component. Some of these analyses may be more easily automatically learnable (and thus more robust) than others. For example, CCG analyzes filler-gap constructions using essentially the same mechanism as right-side complementation, turning gaps into right-seeking functors using a special backward crossed composition combinator when gaps are not at the right periphery of constituents that contain them (Steedman, 2000). This distinction between peripheral and non-peripheral filler-gap constructions, and this merging of filler-gap constructions with right-side complementation, may produce different data densities in training corpora and make certain phenomena harder or easier to learn than, say, an HPSG-style analysis (Pollard and Sag, 1994), which treats all filler-gap constructions the same, but distinguishes constituents containing gaps from constituents awaiting complements.

Naturally, empirical questions about alternative syntactic analyses such as these cannot be resolved by fitting to syntactically annotated corpora, since the syntactic analyses do not agree. This paper therefore addresses the question of how to empirically distinguish between analyses of individual phenomena, or between entire formal systems, on the basis of learnability in a down-stream unbounded dependency recovery task. In particular, this paper describes a method for implementing syntactic analyses of various phenomena through automatic reannotation rules, which operate deterministically on a corpus like the Penn Treebank (Marcus et al., 1993) to produce a corpus with desired syntactic analyses. This reannotated corpus is then used to define a probabilistic grammar which is automatically annotated with additional latent variable values (Petrov and Klein, 2007) to introduce distinctions based on distributions of words and syntactic categories that increase the probability of the corpus (and improve the accuracy of parsing on held-out data), but do not affect the calculation of dependency structure. Dependency structures can then be extracted from parse trees produced by this grammar in a deterministic post-process, and mapped to the dependency representation used by Rimell et al. (2009) for evaluation.

2.2 Reannotation

The reannotation process described in this paper is similar in purpose to efforts to convert Penn Treebank annotations into other grammar formalisms, e.g. CCG (Hockenmaier, 2003) and HPSG (Miyao et al., 2004). These reannotation processes consist of multiple phases for

head-finding, binarization, and relabeling. However, unlike the Hockenmaier or Miyao et al. processes, the rule set described in this paper is intended to be modified and reused by researchers interested in manipulating analyses of individual phenomena (like distinguishing or consolidating filler-gap and right-node raising constructions) to test the generality and learnability of alternative syntactic representations on some down-stream task such as recovering unbounded dependencies. As such, the reannotation rules used in this process are defined to apply in a single top-down pass, pulling arguments and modifiers out of constituents until only a lexical head remains at the bottom. This single-pass architecture allows the rules for reannotating various linguistic phenomena to be relatively modular, so that they can be independently manipulated and evaluated.

The reannotation rules described in this paper are also similar to reannotation rules in the 'wsjsed' system (Blaheta, 2002), but that system was evaluated as an error-correction tool, rather than for making theoretically-motivated changes to syntactic analyses.

The reannotated grammar described in this paper exploits the rich traces and function labels in the Treebank that are typically ignored in parsing. These annotations are transformed into a categorial grammar representation of filler-gap constructions that makes use of 'gap' arguments, similar to the 'slash' arguments used in HPSG. Other approaches also exploit the trace and function labels of the Penn Treebank. Campbell (2004) proposed recovering trace information in a post-process following parsing. In contrast, the sample reannotation described in this paper incorporates filler-gap annotations into the corpus on which the parser is trained. Collins (1997), connected fillers with gaps and introduced a 'TRACE' category into the training data. Since this annotation does not change the constituent structure of the corpus, the addition of this gap annotation did not result in improved parsing results for Collins' gap-annotating model (model 3) as compared to the non-gap-annotating model (model 2). However, these are the kind of unbounded dependencies that need to be recovered in the evaluation described in this paper.

3 Generalized Categorial Grammar

This paper explores the use of a generalized categorial grammar which has the transparent predicate-argument dependencies of traditional categorial grammars (based on function application), but is generalized to allow arbitrary sets of type-constructing operators. An extended set of type-constructing operators and a corresponding set of inference rules are then used to group syntactically-interchangeable signs — for example, those with peripheral and non-peripheral gaps or those occurring in post-nominal and predicative contexts — into equivalent categories.

A generalized categorial grammar (Lambek, 1958; Bach, 1981; Oehrle, 1994) is a tuple $\langle U, O, R, X, M \rangle$ of a set U of primitive category types, a set O of type-constructing operators, a set R of inference rules, a set X of vocabulary items, and a mapping M from vocabulary items to complex types. The set of primitive category types U specify various linguistic forms for descriptions of entities or eventualities, corresponding to different clause types,[1] e.g.:

V: finite verbal (*they knew it*)
I: infinitive verbal (*them to know it*)
B: base-form verbal (*them know it*)
L: participial verbal (*them known it*)
A: adjectival/predicative (*them knowing it*)
R: adverbial (*them knowingly*)

[1]This system of categories may be viewed as a simplification of a tectogrammatical type system such as that of Mihalicek and Pollard (2010), weakened to be representable as a context-free grammar.

G: gerund (*them knowing it*)
N: nominal form (e.g. *their knowledge of it*)
D: determiner (*their knowledge of it's*)
O: genitive (*of their knowledge of it*)
E: embedded infinitive (*for them to know it*)
C: complementized finite (*that they know it*)
Q: interrogative (*did they know it*)
S: complete utterance (*know it*)

The set of type-constructing operators O specify various kinds of arguments:[2]

-a: initial argument
-b: final argument
-c: initial conjunct
-d: final conjunct
-g: filler-gap argument
-h: held argument for right node raising
-i: interrogative pronoun argument
-r: relative pronoun argument

Using this set of primitive category types U and type-constructing operators O, a set of complex categories C can be defined such that:

1. every U is in C
2. every $C \times O \times C$ is in C
3. nothing else is in C

Mapping M defines associations from vocabulary items $x \in X$ to meaning functions and associated categories of the form '$(\lambda ...) : u\varphi_1...\varphi_v\psi$', where '$(\lambda ...)$' is a meaning function and '$u\varphi_1...\varphi_v\psi$' is a category consisting of output category $u \in U$, a sequence of argument categories $\varphi_1, ..., \varphi_v \in \{\textbf{-a}, \textbf{-b}, \textbf{-c}, \textbf{-d}\} \times C$, and an optional non-local argument category $\psi \in (\{\textbf{-r}, \textbf{-i}\} \times C) \cup \{\epsilon\}$. Since this model will be used to generate predicate-argument relations but not scoping relations, these meaning functions are constrained to describe simple existentially-quantified variables over instances of entities or eventualities, connected by a set of numbered argument relations. These meaning functions map instances of entities or eventualities i, j, k to truth values based on whether the described argument relations hold between these referents. These argument relations are defined as numbered functions $(v\,i){=}j$ from eventuality or predicate instances i to argument instances j identified by the number of the function v. The '0' function identifies j as i's predicate concept (so '0' maps entity or eventuality instances to instances of concepts associated with words in X), the '1' function identifies j as i's first argument (e.g. its subject), the '2' function identifies j as i's second argument (e.g. its direct object), and so on.[3] A graphical representation of the predicate-argument relations generated by this system for the sentence *The person who officials say stole millions,* is shown in Figure 1. This is similar to the semantic dependency representations of Mel'čuk (1988) and Parsons (1990).

The meaning functions associated with most words specify just the predicate concept (which is here defined to match the word x):

$$x \;\mapsto_M\; (\lambda_i\,(0\,i){=}x) : u\varphi_1...\varphi_v \tag{1a}$$

[2] The **-a** and **-b** operators may be viewed as equivalent to the forward and backward slash, respectively, of Lambek (1958) or Bar-Hillel (1953) categorial grammars, except that they are not used to represent gap arguments or conjuncts. The **-g** operator is similar to the vertical or neutral slash of Kubota and Levine (2012), used to represent gap arguments. The **-c** and **-d** operators for conjuncts, the **-h** operator for rightward raising, and the **-r** and **-i** operators for relative and interrogative pronoun referents are novel extensions to the system.

[3] More sophisticated meaning functions are possible, but are not necessary for evaluating the accuracy of unbounded dependency recovery.

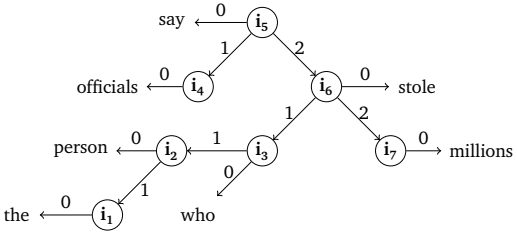


Figure 1: Graphical representation of predicate-argument dependencies for the sentence *The person who officials say stole millions.*

Meaning functions for relative pronouns (Equation 1b) and interrogative pronouns (Equation 1c) introduce additional arguments k, using operators **-r** or **-i** for the referent of the antecedent of a relative or interrogative pronoun respectively:

$$x \;\mapsto_M\; (\lambda_{k\,i}\,(0\,i){=}x \wedge (v\,i){=}k) : u\varphi_1 ... \varphi_{v-1}\text{-}\mathbf{r}c \tag{1b}$$

$$x \;\mapsto_M\; (\lambda_{k\,i}\,(0\,i){=}x \wedge (v\,i){=}k) : u\varphi_1 ... \varphi_{v-1}\text{-}\mathbf{i}c \tag{1c}$$

Inference rules are defined in terms of *composition functions* for arguments, modifiers, and conjuncts. These composition functions each take a meaning function g for an initial (left) child sign and a meaning function h for a final (right) child sign (each defining a set of entity or eventuality instances) and return a meaning function for the parent, which is itself a function from entity or eventuality instances i to truth values:

- Composition functions for arguments $f_{u\varphi_1 ... \varphi_v}$ connect the referent j of an initial (left) child function g as an argument of referent i of a final (right) child function h, or vice versa:

$$f_{u\varphi_1 ... \varphi_{v-1}\text{-}\mathbf{a}c} \stackrel{\text{def}}{=} \lambda_{g\,h\,i}\,\exists_j\,(v\,i){=}j \wedge (g\,j) \wedge (h\,i) \tag{2a}$$

$$f_{u\varphi_1 ... \varphi_{v-1}\text{-}\mathbf{b}c} \stackrel{\text{def}}{=} \lambda_{g\,h\,i}\,\exists_j\,(v\,i){=}j \wedge (g\,i) \wedge (h\,j) \tag{2b}$$

- Composition functions for initial and final modifiers (f_{IM} and f_{FM}) are category-independent and return the referent of the argument (j) rather than of the predicate (i):

$$f_{\text{IM}} \stackrel{\text{def}}{=} \lambda_{g\,h\,j}\,\exists_i\,(1\,i){=}j \wedge (g\,i) \wedge (h\,j) \tag{3a}$$

$$f_{\text{FM}} \stackrel{\text{def}}{=} \lambda_{g\,h\,j}\,\exists_i\,(1\,i){=}j \wedge (g\,j) \wedge (h\,i) \tag{3b}$$

- Composition functions for conjuncts are similar to composition functions for arguments,

except that they only count conjunct arguments, for $c, d \in C$:[4]

$$f_{c\text{-}\mathbf{c}d} \stackrel{\text{def}}{=} \lambda_{ghi}\, \exists_j\, (1\, i){=}j \wedge (g\, j) \wedge (h\, i) \tag{4a}$$

$$f_{c\text{-}\mathbf{d}d} \stackrel{\text{def}}{=} \lambda_{ghi}\, \exists_j\, (2\, i){=}j \wedge (g\, i) \wedge (h\, j) \tag{4b}$$

$$f_{\&} \stackrel{\text{def}}{=} \lambda_{ghi}\, \exists_{jk}\, j{=}(2\, i) \wedge (0\, i){=}(0\, j) \wedge (1\, j){=}k \wedge (g\, k) \wedge (h\, j) \tag{4c}$$

The set of *inference rules R* in the categorial grammar then apply these composition functions to compose and categorize super-lexical signs. These inference rules will use variables f, g, h over meaning functions, variables k over referents for possible values of gaps, variables $u \in U$ over primitive categories, variables $c, d, e \in U \times (\{\textbf{-a}, \textbf{-b}, \textbf{-c}, \textbf{-d}\} \times C)^*$ over categories with local arguments, and variables $\psi \in \{\textbf{-g}, \textbf{-h}, \textbf{-i}, \textbf{-r}\} \times C$ over non-local operators and argument categories:[5]

1. Inference rules for argument attachment apply functors of category c-**a**d or c-**b**d to initial or final arguments of category d. Non-local arguments k, using non-local operator and argument category ψ, are then propagated to the consequent from all possible combinations of antecedents, skipping over the composition function:

$$\frac{g{:}d \quad h{:}\,c\text{-}\mathbf{a}d}{(f_{c\text{-}\mathbf{a}d}\, g\, h){:}\,c} \quad \frac{g{:}d\psi \quad h{:}\,c\text{-}\mathbf{a}d}{\lambda_k\,(f_{c\text{-}\mathbf{a}d}\, (g\, k)\, h){:}\,c\psi} \quad \frac{g{:}d \quad h{:}\,c\text{-}\mathbf{a}d\psi}{\lambda_k\,(f_{c\text{-}\mathbf{a}d}\, g\, (h\, k)){:}\,c\psi} \quad \frac{g{:}d\psi \quad h{:}\,c\text{-}\mathbf{a}d\psi}{\lambda_k\,(f_{c\text{-}\mathbf{a}d}\, (g\, k)\, (h\, k)){:}\,c\psi} \quad \text{(Aa–d)}$$

$$\frac{g{:}\,c\text{-}\mathbf{b}d \quad h{:}d}{(f_{c\text{-}\mathbf{b}d}\, g\, h){:}\,c} \quad \frac{g{:}\,c\text{-}\mathbf{b}d\psi \quad h{:}d}{\lambda_k\,(f_{c\text{-}\mathbf{b}d}\, (g\, k)\, h){:}\,c\psi} \quad \frac{g{:}\,c\text{-}\mathbf{b}d \quad h{:}d\psi}{\lambda_k\,(f_{c\text{-}\mathbf{b}d}\, g\, (h\, k)){:}\,c\psi} \quad \frac{g{:}\,c\text{-}\mathbf{b}d\psi \quad h{:}d\psi}{\lambda_k\,(f_{c\text{-}\mathbf{b}d}\, (g\, k)\, (h\, k)){:}\,c\psi} \quad \text{(Ae–h)}$$

For example, to attach a verb to a direct object with or without a gap:

$$\frac{\dfrac{\text{read} \quad \text{a book about cars}}{\textbf{V-aN-bN} \qquad \textbf{N}}\ \text{Ae}}{\textbf{V-aN}} \qquad \frac{\dfrac{\text{read} \quad \text{a book about}}{\textbf{V-aN-bN} \qquad \textbf{N-gN}}\ \text{Ag}}{\textbf{V-aN-gN}}$$

2. Inference rules for modifier attachment apply initial or final modifiers of category u-**a**d to modificands of category c (again propagating non-local arguments ψ to the consequent from all combinations of antecedents, so as to skip over the composition function):

$$\frac{g{:}\,u\text{-}\mathbf{a}d \quad h{:}c}{(f_{\text{IM}}\, g\, h){:}\,c} \quad \frac{g{:}\,u\text{-}\mathbf{a}d\psi \quad h{:}c}{\lambda_k\,(f_{\text{IM}}\, (g\, k)\, h){:}\,c\psi} \quad \frac{g{:}\,u\text{-}\mathbf{a}d \quad h{:}c\psi}{\lambda_k\,(f_{\text{IM}}\, g\, (h\, k)){:}\,c\psi} \quad \frac{g{:}\,u\text{-}\mathbf{a}d\psi \quad h{:}c\psi}{\lambda_k\,(f_{\text{IM}}\, (g\, k)\, (h\, k)){:}\,c\psi} \quad \text{(Ma–d)}$$

$$\frac{g{:}c \quad h{:}\,u\text{-}\mathbf{a}d}{(f_{\text{FM}}\, g\, h){:}\,c} \quad \frac{g{:}c\psi \quad h{:}\,u\text{-}\mathbf{a}d}{\lambda_k\,(f_{\text{FM}}\, (g\, k)\, h){:}\,c\psi} \quad \frac{g{:}c \quad h{:}\,u\text{-}\mathbf{a}d\psi}{\lambda_k\,(f_{\text{FM}}\, g\, (h\, k)){:}\,c\psi} \quad \frac{g{:}c\psi \quad h{:}\,u\text{-}\mathbf{a}d\psi}{\lambda_k\,(f_{\text{FM}}\, (g\, k)\, (h\, k)){:}\,c\psi} \quad \text{(Me–h)}$$

For example, to attach an adverbial modifier with or without a gap:

$$\frac{\dfrac{\text{sleep} \quad \text{in Aix}}{\textbf{V-aN} \qquad \textbf{R-aN}}\ \text{Me}}{\textbf{V-aN}} \qquad \frac{\dfrac{\text{sleep} \quad \text{in}}{\textbf{V-aN} \qquad \textbf{R-aN-gN}}\ \text{Mg}}{\textbf{V-aN-gN}}$$

[4] The last of these (4c) introduces lexical and compositional relations for elided conjunctions in sequences of three or more conjuncts (e.g. between *creditors* and *investors* in the conjunction *creditors, investors, and employees*).

[5] A deductive system consists of inference rules of the form $\frac{P}{Q}R$, meaning premises or antecedents P entail conclusion or consequent Q according to rule or side condition R (Shieber et al., 1995). Additionally, this notation assumes adjacent premises arise from adjacent and similarly ordered sequences of observations.

3. Inference rules for conjunct attachment apply conjunctions of category c-**c**d or c-**d**d to conjuncts of category d (including repeated initial conjuncts):

$$\frac{g{:}d \quad h{:}\,c\text{-}\mathbf{c}d}{(f_{c\text{-}\mathbf{c}d}\ g\ h){:}\,c} \qquad \frac{g{:}d \quad h{:}\,c\text{-}\mathbf{c}d}{(f_{\&}\ g\ h){:}\,c\text{-}\mathbf{c}d} \qquad \frac{g{:}\,c\text{-}\mathbf{d}d \quad h{:}d}{(f_{c\text{-}\mathbf{d}d}\ g\ h){:}\,c} \qquad \text{(Ca–c)}$$

$$\frac{g{:}d\psi \quad h{:}\,c\text{-}\mathbf{c}d\psi}{\lambda_k(f_{c\text{-}\mathbf{c}d}\ (g\ k)\ (h\ k)){:}\,c} \qquad \frac{g{:}d\psi \quad h{:}\,c\text{-}\mathbf{c}d\psi}{\lambda_k(f_{\&}\ (g\ k)\ (h\ k)){:}\,c\text{-}\mathbf{c}d\psi} \qquad \frac{g{:}\,c\text{-}\mathbf{d}d\psi \quad h{:}d\psi}{\lambda_k(f_{c\text{-}\mathbf{d}d}\ g\ (h\ k)){:}\,c} \qquad \text{(Cd–f)}$$

For example, to combine three noun phrase conjuncts:

$$\dfrac{\text{creditors}\ \ \dfrac{\text{investors}\ \ \dfrac{\text{and}\ \ \text{employees}}{\mathbf{N\text{-}cN\text{-}dN}\quad\ \mathbf{N}}\ \text{Cc}}{\mathbf{N}\qquad\qquad\mathbf{N\text{-}cN}}\ \text{Cb}}{\mathbf{N}\qquad\qquad\mathbf{N\text{-}cN}}\ \text{Ca}$$

4. Inference rules for gap attachment hypothesize gaps as initial arguments, final arguments, or modifiers:[6]

$$\frac{g{:}\,c\text{-}\mathbf{a}d}{\lambda_k\,(f_{c\text{-}\mathbf{a}d}\ \{k\}\ g){:}\,c\text{-}\mathbf{g}d} \qquad \frac{g{:}\,c\text{-}\mathbf{b}d}{\lambda_k\,(f_{c\text{-}\mathbf{b}d}\ g\ \{k\}){:}\,c\text{-}\mathbf{g}d} \qquad \frac{g{:}c}{\lambda_k\,(f_{\mathrm{IM}}\ \{k\}\ g){:}\,c\text{-}\mathbf{g}d} \qquad \text{(Ga–c)}$$

For example:

$$\dfrac{\text{is sleeping}}{\mathbf{V\text{-}aN}}\ \text{Ga}\qquad \dfrac{\text{we}\quad\dfrac{\text{drove}}{\mathbf{V\text{-}aN\text{-}bN}}}{\mathbf{N}\qquad\mathbf{V\text{-}aN\text{-}gN}}\ \begin{matrix}\text{Gb}\\ \text{Ac}\end{matrix}\qquad \dfrac{\text{is sleeping}}{\mathbf{V\text{-}aN}}\ \text{Gc}$$

5. Inference rules for filler attachment apply gapped clauses to modificands or relative or interrogative phrases as fillers:

$$\frac{g{:}e \quad h{:}\,c\text{-}\mathbf{g}d}{\lambda_i\,\exists_j\,(g\ i)\wedge(h\ i\ j){:}e} \qquad \frac{g{:}d\text{-}\mathbf{r}e \quad h{:}\,c\text{-}\mathbf{g}d}{\lambda_{kj}\,\exists_i\,(g\ k\ i)\wedge(h\ i\ j){:}\,c\text{-}\mathbf{r}e} \qquad \frac{g{:}d\text{-}\mathbf{i}e \quad h{:}\,c\text{-}\mathbf{g}d}{\lambda_{kj}\,\exists_i\,(g\ k\ i)\wedge(h\ i\ j){:}\,c\text{-}\mathbf{i}e} \qquad \text{(Fa–c)}$$

For example:

$$\dfrac{\text{the car}\quad\text{we drove}}{\mathbf{N}\qquad\mathbf{V\text{-}gN}}\ \text{Fa}\qquad \dfrac{\text{which}\quad\text{we drove}}{\mathbf{N\text{-}rN}\qquad\mathbf{V\text{-}gN}}\ \text{Fb}\qquad \dfrac{\text{what}\quad\text{do we drive}}{\mathbf{N\text{-}iN}\qquad\mathbf{Q\text{-}gN}}\ \text{Fc}$$

6. Inference rules for relative pronoun attachment apply pronominal relative clauses of category c-**r**d to modificands of category e:

$$\frac{g{:}e \quad h{:}c\text{-}\mathbf{r}d}{\lambda_i\,\exists_j\,(g\ i)\wedge(h\ i\ j){:}e} \qquad \text{(R)}$$

For example:

$$\dfrac{\text{the car}\quad\text{which we drove}}{\mathbf{N}\qquad\mathbf{V\text{-}rN}}\ \text{R}$$

[6]Here, set notation is used in order to save space: $\{k\} = (\lambda_i\ i{=}k)$.

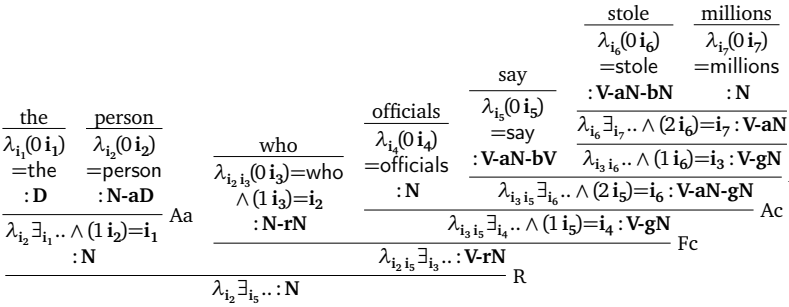


Figure 2: Example categorization of the noun phrase *the person who officials say stole millions*. This derivation yields the following lexical relations: $(0\,\mathbf{i}_1)$=the, $(0\,\mathbf{i}_2)$=person, $(0\,\mathbf{i}_3)$=who, $(0\,\mathbf{i}_4)$=officials, $(0\,\mathbf{i}_5)$=say, $(0\,\mathbf{i}_6)$=stole, $(0\,\mathbf{i}_7)$=millions, and the following argument relations: $(1\,\mathbf{i}_2)$=$\mathbf{i}_1$, $(1\,\mathbf{i}_3)$=$\mathbf{i}_2$, $(1\,\mathbf{i}_5)$=$\mathbf{i}_4$, $(2\,\mathbf{i}_5)$=$\mathbf{i}_6$, $(1\,\mathbf{i}_6)$=$\mathbf{i}_3$, $(2\,\mathbf{i}_6)$=$\mathbf{i}_7$. The semantic dependency relations for this sentence are represented graphically in Figure 1.

7. Inference rules for argument elision (including determiners of plural nouns) simply leave these arguments unspecified in the resulting meaning function:

$$\frac{g\colon c\text{-}\mathbf{a}d}{g\colon c} \qquad \frac{g\colon c\text{-}\mathbf{b}d}{g\colon c} \qquad \frac{g\colon c\text{-}\mathbf{a}d\psi}{g\colon c\psi} \qquad \frac{g\colon c\text{-}\mathbf{b}d\psi}{g\colon c\psi} \qquad \text{(Ea–d)}$$

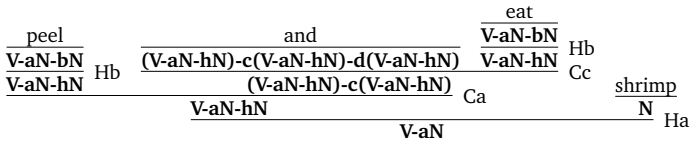
   For example, to elide determiners of plural nouns or optional direct objects:

$$\dfrac{\dfrac{\text{cars}}{\textbf{N-aD}}\,\text{Ea}}{\textbf{N}} \qquad \dfrac{\dfrac{\text{drive}}{\textbf{V-aN-bN}}\,\text{Eb}}{\textbf{V-aN}}$$

8. Inference rules for right node raising introduction and attachment treat right-node raising as a type of non-local argument using operator **-h**:

$$\frac{g\colon c\text{-}\mathbf{h}d \quad h\colon d}{\lambda_i\,\exists_j\,(g\;j\;i)\wedge(h\;j)\colon c} \qquad \frac{g\colon c\text{-}\mathbf{b}d}{\lambda_k\,(f_{c\text{-}\mathbf{b}d}\;g\;\{k\})\colon c\text{-}\mathbf{h}d} \qquad \text{(Ha–b)}$$

   For example:



An example derivation of the noun phrase *the person who officials say stole millions*, exemplifying F, G, and R rules, is shown in Figure 2; and an example derivation of the noun phrase *creditors, investors and employees of the company*, exemplifying C, E, and H rules, is shown in Figure 3. After all lambda expressions are applied to arguments in a derivation, each word is associated

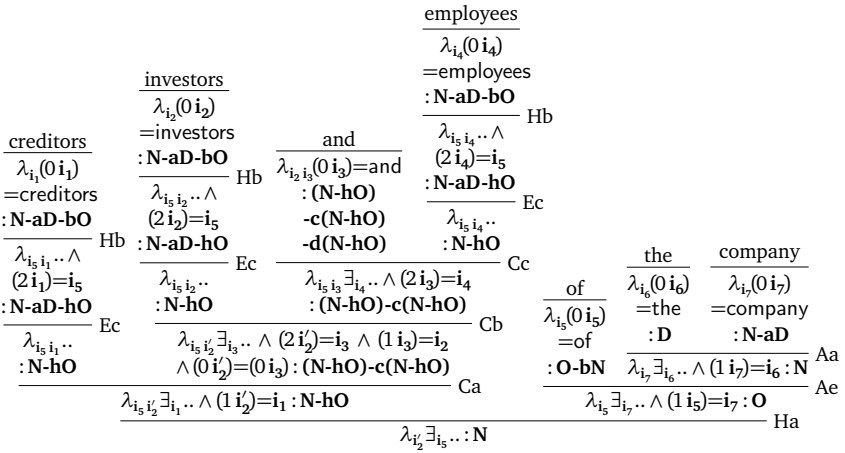


Figure 3: Example categorization of the noun phrase *creditors investors and employees of the company*. This derivation yields the following lexical relations: $(0\,\mathbf{i}_1)$=creditors, $(0\,\mathbf{i}_2)$=investors, $(0\,\mathbf{i}'_2)=(0\,\mathbf{i}_3)$=and, $(0\,\mathbf{i}_4)$=employees, $(0\,\mathbf{i}_5)$=of, $(0\,\mathbf{i}_6)$=the, $(0\,\mathbf{i}_7)$=company, and the following argument relations: $(2\,\mathbf{i}_1)=\mathbf{i}_5$, $(2\,\mathbf{i}_2)=\mathbf{i}_5$, $(2\,\mathbf{i}_4)=\mathbf{i}_5$, $(1\,\mathbf{i}'_2)=\mathbf{i}_1$, $(2\,\mathbf{i}'_2)=\mathbf{i}_3$, $(1\,\mathbf{i}_3)=\mathbf{i}_2$, $(2\,\mathbf{i}_3)=\mathbf{i}_4$, $(1\,\mathbf{i}_5)=\mathbf{i}_7$, $(1\,\mathbf{i}_7)=\mathbf{i}_6$.

with the variable of an existential quantifier. These existentially quantified variables can then be uniquely identified using numerical indices of words, and the numbered functions in lambda expressions $(v\,i)=j$ are interpreted as dependency relations assigning the v^{th} argument of i to be j.

This system has the attractive property that the same syntactic constraints can be assigned the same category in every context. This property is not shared by most categorial grammars: e.g. post-nominal and post-copular prepositional phrases often have different categories.

4 Mapping Treebank to a Generalized Categorial Grammar

The reannotation system described in this paper defines its target grammar in terms of a set of reannotation rules. These reannotation rules work within a script that traverses each bracketed sentence in a corpus by selecting each pair of matching brackets from the top of the tree to the bottom, then running a sed-like pattern substitution rule on each selection (see Figure 4). Such rules can implement local syntactic transformations, as well as certain non-local transformations like adding gap arguments to constituents containing a particular trace marker, for example. Reannotation of the categorial grammar evaluated in this paper requires about 150 such rules. These rules are modular and can be reused or modified to experiment with different syntactic analyses.

In order to make the trace and function labels in the Treebank accessible to a PCFG-based latent variable annotator, they must be incorporated into the syntactic categories of each tree and propagated from filler to gap constituents, creating a categorial grammar with local associations between parents and immediate children. First all trace annotations for interrogatives, relative

a) s/{(V-rN) <(WHNP)(-[0-9]+)([^ >]*)> (.*[-NONE- *T*\3].*)}/{\1 <N-rN\4> <V-gN\3 \5>}/;

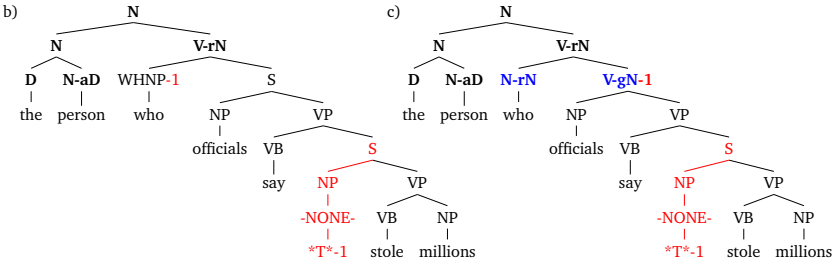


Figure 4: Sample sed-like reannotation rule introducing a gap tag at the top of a relative clause (a), and an application of this rule to the movement-based notation in the Penn Treebank (b) to produce a binary-branching categorial grammar derivation using gap arguments (c). Rules are applied to every constituent from the top of the tree down, using parentheses to delimit constituents above the current constituent, curly braces to delimit the current constituent, angle brackets to delimit child constituents, and square brackets to delimit constituents below children. Delimiters are then updated at every iteration.

clauses, and topicalizations are transformed into gap arguments: **-gN** (for noun phrase gaps), **-g(R-aN)** (for adverbial phrase gaps), and **-gS** (for e.g. topicalized sentential gaps), which follow the category label for each constituent. Similar transformations localize it-clefts (*it seems that...*), tough constructions (*tough to cut*), parentheticals (*he/she said*), and certain types of inversion (*'it rained,' she said*), also using the gap operator **-g**. This is similar to the treatment of filler-gap constructions used in HPSG (Pollard and Sag, 1994).

Then, specifiers of head projections are annotated as initial arguments, e.g. **-aN** for nominal subjects, and complements of head projections are annotated as final arguments: e.g. **-bN** for transitive verbs, prepositions, and certain adjectives, **-bV** for sentential complements, **-bN-bN** for ditransitive verbs, etc. The **-h** operator is then used to propagate directional dependencies in right node raising (*they peeled and ate shrimp*). These are similar to the 'subcat' feature in HPSG, or to the left and right slash in categorial grammar accounts of specification, complementation and right node raising.

Then, interrogative and relative pronouns (e.g. *what* and *which*) are distinguished with **-iN** and **-rN** arguments, respectively, in order to regularize typical contexts for filler-gap traces.

Conjuncts are assigned **-c** and **-d** arguments to distinguish composition functions for conjuncts from composition functions for ordinary arguments. This distinction makes it possible for ordinary arguments to be shared among conjuncts.

The transform rules are defined as recursive rewrites that progress down the Treebank trees, propagating **-a**, **-b**, **-c**, **-d**, **-g**, **-h**, **-i**, and **-r** arguments as they go. Figure 4 shows a single step in a reannotation of the noun phrase *the person who officials say stole millions.* This set of recursive rules obtains about 94% coverage of the training set, consisting of sections 2–21 of the Penn Treebank. Trees not completely transformed by these rules are excluded from training. Since the system is evaluated on unbounded dependency annotations as described by Rimell et al. (2009), it is not necessary to apply this transformation to the test set.

In many cases the conversion to this categorial grammar replaces Treebank category labels with more general specifications. For example, in most contexts, adjective phrases, prepositional phrases, and progressive and passive verb phrases are replaced with a predicative category **A-aN**. This allows an HPSG-, GPSG- or CCG-like treatment of conjunction of any of these kinds of phrases with any other, following Sag et al. (1985), Gazdar et al. (1985), and Komagata (2002). This generalized predicative category is also used for noun phrase modifiers, and for noun phrases following the copular *be*. This annotation is compatible with additional specification of categories for adjective phrases (following words like *become*), or gerund phrases (following words like *start*), but these contexts cannot be reliably identified by the current reannotation.

In other cases, the conversion to categorial grammar distinguishes categories that are conflated in the Treebank. For example, the Treebank 'SBAR' category is distinguished into adjectival relative clauses **V-rN**, embedded questions **V-iN**, embedded inflected sentences **C**, embedded infinitival sentences **E**, nominal clauses **N**, adjectival modifier phrases **A-aN**, and adverbial modifier phrases **R-aN** (e.g. *because* . . .). Treebank 'S' categories are similarly distinguished into inflected sentences **V**, infinitival sentences **I**, base form sentences **B**, adjectival sentences (small clauses) **A**, and participial sentences **L**.[7]

Also, like other categorial grammars, this transform leaves trees in binary branching (Chomsky normal) form.

5 Evaluation

To evaluate the reannotation system described in this paper, a version of the Penn Treebank was annotated with a generalized categorial grammar as described in Section 3. This required about 150 reannotation rules, with about 20 rules for each of noun phrases, verb phrases, prepositional phrases, adjectival phrases, and adverbial phrases, and about 50 rules for various sentence types. This reannotation was applied to Sections 02 to 21 of the WSJ portion of the Penn Treebank (Marcus et al., 1993). The resulting corpus then underwent three iterations of latent variable annotation, using the split-merge algorithm of Petrov et al. (2006).[8] Each iteration of this algorithm splits the categories in the training corpus randomly in two, runs the inside-outside algorithm (expectation maximization) on the resulting trees to maximize the probability of the training set, then merges those categories that contributed the least to the maximization. The categorial grammar type-constructing operators were then used to extract semantic dependency relations for each parsed sentence.

This evaluation used development and test sets from Rimell et al. (2009) for tuning the annotation mappings and testing, respectively. The de Marneffe et al. (2006) dependencies used in these sets are deterministically mapped to the comparable numeric relations used by the current system.[9] Thus, the dependencies 'nsubj' and 'nsubjpass' are mapped to a '1' relation, and 'dobj', 'pobj', and 'obj2' are mapped to a '2' relation. The dependencies 'advmod', 'prep', and 'nn' are also mapped to a '1' relation with the direction of the dependency reversed.[10]

[7] It is important to note that these rules allow the word *that* to be treated as a relativizer rather than a relative pronoun, as it is in other categorial grammars, such as CCG.

[8] Under the current system, this number of split-merge iterations was empirically shown to yield the same results on development data as five iterations of split-merge, which is the recommended number for maximum accuracy without overfitting according to Petrov and Klein (2007).

[9] This simplification of dependency labels to numbers can be losslessly reversed by looking at the categories of the involved predicates.

[10] One anonymous reviewer points out that this reversal of direction for modifier dependencies is similar to that described in dependency accounts of Tree Adjoining Grammars (Joshi, 1985; Candito and Kahane, 1998).

	Obj RC	Obj Red	Sbj RC	Free	Obj Q	RNR	Sbj Embed	Total
Enju	47.3	65.9	82.1	**76.2**	32.5	47.1	32.9	54.4
C&C	**59.3**	62.6	80.0	72.6	27.5	**49.4**	22.4	53.6
Malt	40.7	50.5	**84.2**	70.2	16.2	39.7	23.5	46.4
MST	34.1	47.3	78.9	65.5	18.8	45.4	37.6	46.1
Stanford	22.0	1.1	74.7	64.3	41.2	45.4	10.6	38.1
DCU	23.1	41.8	56.8	46.4	27.5	40.8	5.9	35.7
Rasp	16.5	1.1	53.7	17.9	27.5	34.5	15.3	25.3
This system	52.7	**69.2**	68.4	69.0	**57.5**	26.4	**38.8**	**54.6**

Table 1: Unbounded dependency results compared to those of other systems studied by Rimell et al. (2009) and Nivre et al. (2010) over a variety of constructions: object extraction from relative clauses (Obj RC), object extraction from reduced relative clauses (Obj Red), subject extraction from relative clauses (Sbj RC), free relatives (Free), object wh-questions (Obj Q), right node raising (RNR), and subject extraction from embedded clauses (Sbj Embed). Evaluated parsers are C&C (Clark and Curran, 2007), Enju (Miyao and Tsujii, 2005), DCU (Cahill et al., 2004), Rasp (Briscoe et al., 2006), Stanford (Klein and Manning, 2003), MST (McDonald, 2006), Malt (Nivre et al., 2006a,b). This system used the Berkley parser (Petrov and Klein, 2007) run on the reannotated categorial grammar.

Due to differences between the de Marneffe et al. (2006) dependency representation and that of the current system, some deterministic modifications were required for evaluation against the Rimell et al. (2009) corpus.[11]

1. If the hypothesized target of a dependency is a conjunction, the dependencies to each of its conjuncts are hypothesized instead;

2. If the target of a dependency is a relativizer or a relative pronoun, the predicate it modifies is used in its place; and

3. If the source predicate of a dependency has a category of **O**, the predicate that depends on the hypothesized target is hypothesized as the target.

For example, in this categorial grammar analysis of the phrase *levels of health*, the word *of* heads a phrase of category **O**, with *health* as an argument, and *levels* depends directly on *health*, so the dependency that is evaluated is from *of* to *levels*. Finally, following Rimell et al. (2009), the current system counts both dependencies in a conjunction as having been captured if the first is correct and the conjunction is modelled correctly. If the source has the same type of relation to the target in the gold standard as was output by the current system, the dependency is counted as correct. The final results are tabulated in comparison to those in Rimell et al. (2009) and Nivre et al. (2010) in Table 1.[12]

[11] This automated scoring makes the evaluation less generous than the manual output interpretations given in Rimell et al. (2009) and Nivre et al. (2010), but has the advantage of being easily reproducible.

[12] Note that one of the other systems was a variant of C&C which was given additional training on QuestionBank (Judge et al., 2006). This extra training and the results obtained from it were not used in the experimental description in this paper because i) other systems were not trained on this data and ii) extra training examples may distort the prior model to reduce probability of non-questions, which may have an adverse effect on overall parsing accuracy.

6 Conclusion

This paper has presented a system for automatically reannotating syntactically-annotated corpora for the purpose of refining linguistically-informed phrase structure analyses of various phenomena. The paper has also presented a generalized categorial grammar reannotation developed using this system which combines the transparent predicate-argument structure of a categorial grammar with the categorial generality of an HPSG-like system. The system achieves unbounded dependency parsing accuracy favorably comparable to all the systems recently studied by Rimell et al. (2009) and Nivre et al. (2010) on this same task.

This system and the generalized categorial grammar reannotation rules (and other scripts needed to produce the reannotated corpus) are available at:
http://sourceforge.net/projects/modelblocks/

Acknowledgements

Thanks to Laura Rimell for providing the unbounded dependency corpus and for her clarifications regarding the unbounded dependency recovery task. Thanks also to Bob Levine and Carl Pollard for insightful discussions related to this project, and to the anonymous reviewers for their comments. This work was funded by a Department of Linguistics Targeted Investment for Excellence (TIE) grant for collaborative interdisciplinary projects conducted during the academic year 2012-13.

References

Bach, E. (1981). Discontinuous constituents in generalized categorial grammars. *Proceedings of the Annual Meeting of the Northeast Linguistic Society (NELS)*, 11:1–12.

Bar-Hillel, Y. (1953). A quasi-arithmetical notation for syntactic description. *Language*, 29:47–58.

Blaheta, D. (2002). Handling noisy training and testing data. In *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing*.

Briscoe, T., Carroll, J., and Watson, R. (2006). The second release of the RASP system. In *Proceedings of the Interactive Demo Session of COLING/ACL-06*, Sydney, Australia.

Cahill, A., Burke, M., Genabith, J. V., and Way, A. (2004). Long-distance dependency resolution in automatically acquired wide-coverage PCFG-based lfg approximations. In *Proceedings of the 42nd Meeting of the ACL*, pages 320–327, Barcelona, Spain.

Campbell, R. (2004). Using linguistic principles to recover empty categories. In *In Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics (ACL-04)*, pages 645–652, Barcelona, Spain.

Candito, M.-H. and Kahane, S. (1998). Can the TAG derivation tree represent a semantic graph? In *Proceedings of the TAG+4 Workshop*, University of Pennsylvania.

Clark, S. and Curran, J. R. (2007). Wide-coverage efficient statistical parsing with CCG and log-linear models. *Computational Linguistics*, 33:493–552.

Collins, M. (1997). Three generative, lexicalised models for statistical parsing. In *Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics (ACL'97)*.

de Marneffe, M.-C., MacCartney, B., and Manning, C. D. (2006). Generating typed dependency parses from phrase structure parses. In *Proceedings of LREC 2006*.

Gazdar, G., Klein, E., Pullum, G., and Sag, I. (1985). *Generalized Phrase Structure Grammar*. Harvard University Press, Cambridge, MA.

Gildea, D. and Hockenmaier, J. (2003). Identifying semantic roles using combinatory categorial grammar. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Sapporo, Japan.

Hockenmaier, J. (2003). *Data and Models for Statistical Parsing with Combinatory Categorial Grammar*. PhD thesis, University of Edinburgh.

Joshi, A. K. (1985). How much context sensitivity is necessary for characterizing structural descriptions: Tree adjoining grammars. In D. Dowty, L. K. and Zwicky, A., editors, *Natural language parsing: Psychological, computational and theoretical perspectives*, pages 206–250. Cambridge University Press, Cambridge, U.K.

Judge, J., Cahill, A., and van Genabith, J. (2006). Questionbank: Creating a corpus of parse-annotated questions. In *Proceedings of the Joint Conference of the International Committee on Computational Linguistics and the Association for Computational Linguistics*.

Klein, D. and Manning, C. D. (2003). Accurate unlexicalized parsing. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, pages 423–430, Sapporo, Japan.

Komagata, N. (2002). Coordination of unlike (?) categories: How not to distinguish categories.

Kubota, Y. and Levine, R. (2012). Gapping as like-category coordination. *Logical Aspects of Computational Linguistics*, (7351):Al135–150.

Lambek, J. (1958). The mathematics of sentence structure. *American mathematical monthly*, pages 154–170.

Marcus, M. P., Santorini, B., and Marcinkiewicz, M. A. (1993). Building a large annotated corpus of English: the Penn Treebank. *Computational Linguistics*, 19(2):313–330.

McDonald, R. (2006). *Discriminative Learning and Spanning Tree Algorithms for Dependency Parsing*. PhD thesis, University of Pennsylvania.

Mel'čuk, I. (1988). *Dependency syntax: theory and practice*. State University of NY Press, Albany.

Mihalicek, V. and Pollard, C. (2010). Distinguishing phenogrammar from tectogrammar simplifies the analysis of interrogatives. In *Formal Grammar*, pages 130–145.

Miyao, Y., Ninomiya, T., and Tsujii, J. (2004). Corpus-oriented grammar development for acquiring a head-driven phrase structure grammar from the Penn Treebank. In *Proceedings of the First International Joint Conference on Natural Language Processing (IJCNLP-04)*, pages 684—693, Hainan Island, China.

Miyao, Y. and Tsujii, J. (2005). Probabilistic disambiguation models for wide-coverage HPSG parsing. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 83–90, Michigan, Ann Arbor.

Nivre, J., Hall, J., and Nilsson, J. (2006a). Maltparser: A data-driven parser-generator for dependency parsing. In *In Proc. of LREC*, pages 2216–2219.

Nivre, J., Hall, J., Nilsson, J., Eryiğit, G., and Marinov, S. (2006b). Labeled pseudo-projective dependency parsing with support vector machines. In *Proceedings of the Tenth Conference on Computational Natural Language Learning*, CoNLL-X '06, pages 221–225, Stroudsburg, PA, USA. Association for Computational Linguistics.

Nivre, J., Rimell, L., McDonald, R., and Gómez-Rodríguez, C. (2010). Evaluation of dependency parsers on unbounded dependencies. In *Proceedings of the 23rd International Conference on Computational Linguistics (COLING'10)*, pages 833–841.

Oehrle, R. T. (1994). Term-labeled categorial type systems. *Linguistics and Philosophy*, 17(6):633–678.

Parsons, T. (1990). *Events in the Semantics of English*. MIT Press.

Petrov, S., Barrett, L., Thibaux, R., and Klein, D. (2006). Learning accurate, compact, and interpretable tree annotation. In *Proceedings of the 44th Annual Meeting of the Association for Computational Linguistics (COLING/ACL'06)*.

Petrov, S. and Klein, D. (2007). Improved inference for unlexicalized parsing. In *Proceedings of NAACL HLT 2007*, pages 404–411, Rochester, New York. Association for Computational Linguistics.

Pollard, C. and Sag, I. (1994). *Head-driven Phrase Structure Grammar*. University of Chicago Press, Chicago.

Rimell, L., Clark, S., and Steedman, M. (2009). Unbounded dependency recovery for parser evaluation. In *Proceedings of EMNLP 2009*, volume 2, pages 813–821.

Sag, I. A., Gazdar, G., Wasow, T., and Weisler, S. (1985). Coordination and how to distinguish categories. *Natural Language & Linguistic Theory*, 3:117–171.

Shieber, S. M., Schabes, Y., and Pereira, F. C. (1995). Principles and implementation of deductive parsing. *Journal of Logic Programming*, 24:3–36.

Steedman, M. (2000). *The syntactic process*. MIT Press/Bradford Books, Cambridge, MA.

Tibetan Base Noun Phrase Identification Framework Based on Chinese-Tibetan Sentence Aligned Corpus

NUO Ming Hua[1,2] LIU Hui Dan[1,2] ZHAO Wei Na[1,3] MA Long Long[1] WU Jian[1] DING Zhi Ming[1]
(1) Institute of Software, Chinese Academy of Sciences, Beijing, China
(2) Graduate University of the Chinese Academy of Sciences, Beijing, China
(3) Qinghai Normal University, Xining, China
Minghua,huidan,weina,longlong,wujian,zhiming@iscas.ac.cn

ABSTRACT

This paper presents an identification framework for extracting Tibetan base noun phrase (NP). The framework includes two phases. In the first phase, Chinese base NPs are extracted from all Chinese sentences in the sentence aligned Chinese-Tibetan corpus using Stanford Chinese parser. In the second phase, the Tibetan translations of those Chinese NPs are identified using four different methods, that is, word alignment, iterative re-evaluation, dictionary and word alignment, and sequence intersection method. We implemented and tested these methods on Chinese-Tibetan sentence aligned unlabelled corpus without Tibetan POS tagger and Treebank. The experimental results demonstrate these methods can get satisfactory results, and the best performance with 0.5283 precision is got using sequence intersection identification method. The identification framework can also be extended to extract Tibetan verb phrase.

Title and abstract in Chinese

基于汉藏句子对齐语料的藏文BaseNP识别框架

本文提出藏文BaseNP识别框架，它分两步完成。先通过句法分析得到汉语BaseNP。再为这些汉语BaseNP从汉藏句子对齐语料中识别出藏文对应短语。我们应用四种方法识别藏文BaseNP，分别是词对齐、迭代重估算法、词典和词对齐相结合的方法以及基于序列相交的方法。评价实验表明，没有藏文词性标注和树库的前提下，基于序列相交的方法性能最好。本文提出的框架可以用于藏文动词短语识别任务中。

KEYWORDS : Tibetan information processing; base noun phrase; head-phrase;
CHINESE KEYWORDS :藏文信息处理,基本名词短语,中心语块

Proceedings of COLING 2012: Technical Papers, pages 2141–2158,
COLING 2012, Mumbai, December 2012.

1 Introduction

Shallow parsing identifies the non-recursive cores of various phrase types in text, possibly as a precursor to full parsing or information extraction (Abney, 1991). The paradigmatic shallow parsing problem is NP chunking, which finds the non-recursive cores of noun phrases called BaseNPs. It can help to solve many natural language processing tasks, such as information extraction, named entity extraction, machine translation, and text summarization and so on.

In general, researchers consider chunking as a kind of tagging problem or a sequence labelling task. Machine learning techniques are often applied to chunking. In Tibetan information processing, the shortage of Tibetan language resource leads to the fact that most of the techniques related text processing are still developing. Since 2003, research on Tibetan corpus, Tibetan word segmentation are reported. Although there is no public available Tibetan annotated corpus and Tibetan Treebank, we intend to extract Tibetan BaseNP using machine translation techniques based on our Chinese-Tibetan sentence aligned corpus. The research on Tibetan BaseNP is still in the initial stage. So far, there is no related report. In this paper, we will propose several methods for automatic identification of Tibetan base noun phrases.

The concept of BaseNP is initially put forward by (Church, 1998). In English, BaseNP is simple and non-recursive noun phrase which does not contain other noun phrase descendants. It cannot meet the needs in Tibetan information processing. Presently, different definitions of Chinese BaseNP are used on the basis of research field. According to our Chinese-Tibetan corpus, restrictive attribute phrase is in the scope of our BaseNP extraction. Observing that the Tibetan BaseNP is different from English, BaseNP in Tibetan can be recursively defined as follows, which is in accordance with the definition of Chinese BaseNP in (Zhao and Huang, 1998).

Definition 1: Tibetan base noun phrase (abbreviated as *BaseNP*)
BaseNP ::= *BaseNP* + *BaseNP*
BaseNP ::= *BaseNP* + *Noun*
BaseNP ::= *Determinative modifier* + *BaseNP*
BaseNP ::= *Determinative modifier* + *Noun*
Determinative modifier ::= *Adjective* | *Distinctive Adjectives (DA)* |*Nominalized Verb*| *Noun* |*Location* |*Numeral* + *Quantifier*

The Determinative modifiers have agglutinative relation with the heads.

The rest of this paper is organized as follows. Section 2 introduces related work of BaseNP chunking. Section 3 describes the outline of our framework. In Section 4, we propose four methods to automatically identify the Tibetan BaseNP which are very convenient to manual proofreading. In Section 5, we make an experiment to evaluate the four methods by three metrics, namely coverage, quasi-precision, and precision, then concludes this paper.

2 Related work

2.1 English BaseNP chunking

In 1991, Abney proposed to approach parsing by starting with finding correlated chunks of words (Abney, 1991). The pioneering work of Ramshaw and Marcus (1995) introduced NP chunking as a machine-learning problem, with standard datasets and evaluation metrics. Their work has inspired many others to study the application of learning methods to noun phrase chunking. Other chunk types have not received the same attention as NP chunks. At home and abroad, many statistical and machine learning methods are applied to the English BaseNP identification, and have achieved good recognition performance.

The task was extended to additional phrase types for the CoNLL-2000 shared task (Sang and Buchholz, 2000), which is the standard evaluation task for shallow parsing now. In this conference, many systems used the Machine learning methods, and among them, the most representative and effective one is Support Vector Machine (SVM) based method (Kudo and Matsumoto, 2000). Recently, some new statistical techniques, such as CRF (Lafferty et al. 2001), Winnow algorithm (Zhang, 2001) and structural learning methods (Ando and Zhang, 2005) have been applied to the BaseNP chunking task. Sha and Pereira (2003) considered chunking as a sequence labelling task and achieved good performance by an improved training method of CRF. Ando and Zhang (2005) presented a novel semi-supervised learning method on chunking and produced higher performance than the previous best results.

2.2 Chinese BaseNP chunking

Researchers apply similar methods of English BaseNP chunking to Chinese. Zhao and Huang (1998) made a strict definition of Chinese BaseNP in terms of combination of determinative modifier and head noun and put forward a quasi-dependency model to analyse the structure of Chinese BaseNP. There are some other methods to deal with Chinese phrase (not only BaseNP) chunking, such as HMM (Li et al., 2003), Maximum Entropy (Zhou et al., 2003), Memory-Based Learning (Zhang and Zhou, 2002) etc. Xu et al. (2006) propose a hybrid error-driven combination approach to chunking Chinese BaseNP, which combines TBL (Transformation-based Learning) model and CRF. In order to analyse the results respectively from the two (TBL-based and CRF-based) classifiers and improve the performance of the BaseNP chunker, an error-driven SVM based classifier is trained from the classification errors of the two classifiers. The hybrid method outperforms the previous works.

In general, the flexible structure of Chinese noun phrase often results in the ambiguities during the recognition procedure. Compared with English, internal grammatical structure of phrases is not rigorous; long noun phrase in Chinese is richer. Usage of Chinese word may serve with multi POS (Part-of-Speech) tags. Therefore, the chunker is puzzled by those multi-used words. Furthermore, there are no standard datasets and evaluation systems for Chinese BaseNP chunking as the CoNLL-2000 shared task, which makes it difficult to compare and evaluate different Chinese BaseNP chunking systems.

2.3 Tibetan chunking

In Tibetan information processing, the shortage of Tibetan language resource leads to the fact that most of the techniques related text processing are still developing. Recently, the focus of Tibetan information processing is gradually transferred from word processing to text processing. The Tibetan text processing started in the early 1990s, mainly analyse statically at the beginning. Since 2003, research on Tibetan syntactic chunks is reported.

Jiang (2003a) describes the basic types of syntactic chunks and their formal markers in modern Tibetan, and propose a scheme of automatic word-segmentation based on chunks according to the features of Tibetan syntactic structures. This paper finds the left and right boundaries of each chunk on the basis of pre-processing, setting up small tables of formal markers of each chunk, the verbal paradigm, special tables of homographs, etc, and goes on segmenting words with a dictionary and tagging within chunk. In (Jiang, 2003b), they discuss the automatic recognition strategies of nominalization markers in the modern Tibetan language. The purpose of identifying the nominalization markers is to make automatic word-segmentation within non-finite VP chunks. Due to the complexity of the formal features as well as distribution of the markers, the paper proposes the major recognition approach of the nominalization markers which distinguish between nominal markers and their homographic words. Huang et al. (2005) defines the nominal chunks of Tibetan according to the structures and syntax. Their identification strategy of nominal chunks depends mainly on the markers on the right boundary of the chunks. These previous works define the basic types of syntactic chunks and their formal markers. Identification of chunk is rule-based, including word order rule and syntactic rules of chunk. These papers just illustrate chunking result of several example sentences without experimental data.

The research on Tibetan BaseNP Chunking is, however, still at its initial stage. There is no public available Tibetan Treebank, even a POS tagger at present. In addition, there is no annotated Tibetan corpus available which contain specific information about dividing sentences into chunks of words of arbitrary types. Since we have large-scale Chinese-Tibetan sentence aligned corpus, public available Chinese parser, and word segmentation software etc., we can identify Tibetan BaseNP using these existing resources. Therefore, a Tibetan BaseNP identification framework based on Chinese-Tibetan sentence aligned corpus is proposed in the following.

3 Brief description of Tibetan BaseNP identification framework

The proposed Tibetan BaseNP identification framework consists of three main steps: pre-processing step, Chinese BaseNP extraction step, and Tibetan BaseNP identification step, which are in boldface in FIGURE 1(A). Chinese BaseNP extraction step and the Tibetan BaseNP identification step are the core of the identification framework.

In pre-processing step, sentence aligned Chinese and Tibetan corpus are word segmented and stored separately to the two documents, one sentence per line. Then words in sentence pairs are aligned using Giza++ toolbox (Och and Ney, 2003). FIGURE 1(B) shows the data flowchart of pre-processing.

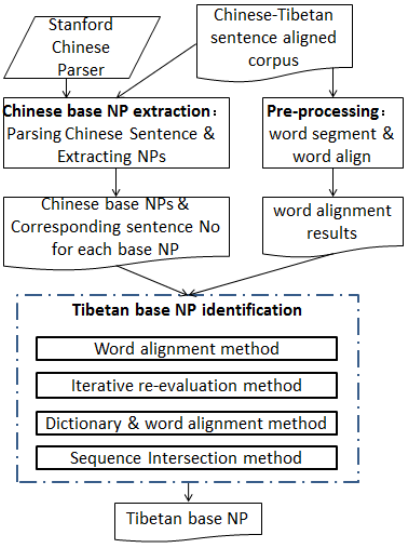


FIGURE 1 (A) – Flow chart of Tibetan BaseNP identification

(B) – Flow chart of pre-processing

In Chinese BaseNP extraction step, we use Stanford Chinese parser to parse all sentences in Chinese corpus from step 1; Extract all NP phrases from parsing results and note the sentence number in which there is a Chinese BaseNP.

The final step is Tibetan BaseNP identification. Aligned sentence pairs, their word alignment results and Chinese BaseNP extracted in step 2 is the input of Tibetan BaseNP identification. To determine the alignment for each Chinese BaseNP from step 2, different approaches within the dotted line in FIGURE 1(A) are proposed.

In the condition that there are no annotated Tibetan corpus, Treebank and Tibetan POS tagger, the framework regards the Tibetan BaseNP identification as translation problem, which confirms correct correspondence for those extracted Chinese BaseNP based on Chinese-Tibetan sentence aligned corpus. We assume that when a phrase in a source language is a BaseNP, its translation in target language is BaseNP too. The term "correspondence" is used here to signify a mapping between words in two aligned sentences. Specify that symbol '↔' is used to represent alignment. Any sentence-pair is symbolized by *SP*, and notated as $SP = CS \leftrightarrow TS$, where *CS* and *TS* stand for Chinese and Tibetan sentence respectively. A word sequence in *CS* is defined here as the correspondence of another sequence in *TS* if the words of one sequence are considered to represent the words in the other. In other words, Chinese-Tibetan BaseNP correspondence is an alignment at phrase-level.

Definition 2: Chinese-Tibetan BaseNP correspondence

$< C_{mi}, C_{mi+1}, \cdots, C_{mi+p} > \leftrightarrow < T_{nj}, T_{nj+1}, \cdots, T_{nj+q} >.$

On the left of alignment symbol is a Chinese BaseNP. Definition of Chinese BaseNP is the same to definition 1. On the right is its translation in Tibetan sentence of aligned sentence pair where Chinese BaseNP located, it conforms to definition 1 too. In this paper, Tibetan BaseNP consist of two or more words are taken into consideration, because of the task objectives.

In next section, we describe in detail how to identify Tibetan BaseNP correspondence to the extracted Chinese BaseNP. Different methods are evaluated, and we will select the method with best performance to generate referable Tibetan BaseNP, which is fully or partial correct, for further manual proofreading.

4 Tibetan BaseNP identification methods

Four different methods, that is, word alignment, iterative re-evaluation, dictionary and word alignment, and sequence intersection method are proposed to determine Tibetan correspondences for Chinese BaseNP. In the following, we elaborate the four methods.

4.1 Word alignment method

This subsection presents word alignment (WA, hereafter) method. In this method, phrase-level alignments are obtained on the basis of the optimal two-way word alignment results from Giza++. Outline of WA method is given below.

Step 1: Run Giza++ to get word alignment results of aligned sentence pair in Chinese-Tibetan corpus.

Step 2: For each Chinese BaseNP, get the word alignment results of located sentence pair and corresponding Tibetan sentence based on the sentence number it located.

Step 3: For each word in Chinese BaseNP, obtain the aligned Tibetan word according to word alignments using a heuristic. These Tibetan words constitute a Tibetan BaseNP for current Chinese BaseNP.

A number of different word alignment heuristics are implemented; in the end, grow-diag-final heuristic shows better performance than others on our aligned corpus. Consequently WA method with grow-diag-final heuristic is regarded as a baseline method.

4.2 Iterative re-evaluation method

This subsection describes iterative re-evaluation (IRE, hereafter) method, which is based on correlations of Chinese phrase and Tibetan words, to complete the identification of Tibetan BaseNP. It is an instance of a general approach to statistical estimation, represented by the EM algorithm (Dempster et al., 1977). Iterative re-evaluation algorithm assumes that a Chinese phrase has a corresponding probability with every Tibetan word; we call it relevancy (R). First, we assign an initial value to R, and then iteratively update the value of R based on correlations between Tibetan word

and Chinese phrase. If the absolute value of the difference between the latest updated two R value, the iterative process stops. Eventually we obtain satisfactory correlations of Chinese phrase and Tibetan words. We will describe the details of iterative re-evaluation algorithm in this section.

Relevancy of Chinese phrase and Tibetan words are used to identify the potential Tibetan BaseNP translation for each Chinese phrase. Tibetan words with higher relevancy should be within the translation of Chinese phrase. If we denote the relevancy of a Chinese phrase *c* and a Tibetan word *t* by $R(c,t)$, then we can calculate it with the following formula.

$$R(c,t) = \frac{W(c,t)}{\sum_{q \in V_F} W(c,q)} \qquad \text{satisfied} \quad \sum_{q \in V_F} R(c,q) = 1 \tag{4.1}$$

In formula(4.1), $W(c,t)$ represents weighted frequency (product of co-occurrence frequency and relevancy) of *c* and *t*. Every Chinese phrase *c* and every Tibetan word *t* has a weighted frequency. V_F indicates words set of Tibetan corpus.

Weighted frequency of Chinese phrase and Tibetan word is the key to the calculation of relevancy. Weighted frequency of *c* and *t* is defined as follow:

$$W(c,t) = \sum_{i=1}^{N} F(i,c,t) R(c,t) \tag{4.2}$$

In which *N* represents the number of sentence pairs in Chinese-Tibetan corpus. $F(i,c,t)$ indicates the number of simultaneous occurrence of *c* and *t* in i^{th} sentence pair. Equation (4.3) assumes that each Chinese BaseNP is initially equally likely to correspond to each Tibetan BaseNP. The weights $W_0(c,t)$ can be interpreted as the mean number of times that *c* corresponds to *t* given the corpus and the initial assumption of equivalent correspondences.

$$W_0(c,t) = \sum_{i=1}^{N} F(i,c,t) \frac{1}{\phi(i)} \tag{4.3}$$

Where $\phi(i)$ indicates the number of Tibetan words in i^{th} sentence pair.

Let *r* be the number of iterations, $P(c,t)$ and $W(c,t)$ for iteration *r* is formulated as in formula (4.4) and (4.5).

$$R_r(c,t) = \frac{W_{r-1}(c,t)}{\sum_{q \in V_F} W_{r-1}(c,q)} \tag{4.4}$$

$$W_r(c,t) = \sum_{i=1}^{N} F(i,c,t) R_{r-1}(c,t) \tag{4.5}$$

The procedure is then iterated using Equations (4.4) and (4.5) to obtain successively refined, convergent estimates of the probability that *c* corresponds to *t*. Experiment results shows that it works well when the iterative threshold is 0.001. It means that we can find Tibetan word *t* with highest corresponding probability to each Chinese phrase,

after several iterations. If $|R_r(c,t)-R_{r-1}(c,t)|<0.001$ then stop iterating.

IRE method can globally calculate the corresponding information, while Mutual Information is used to measure the relevance between Chinese phrase and Tibetan word in isolation without the information of other Tibetan words. For instance, if a few Tibetan words are correct translation for a certain Chinese phrase in a sentence pair, and its located sentence is long, it will lead to lower initial relevancy. However, after iteration, the weighted frequency will be increased due to the high co-occurrence frequency; and the proportion become greater in the sum of weighted frequency of all words, which makes the relevancy increased. Meanwhile, the proportion of weighted frequency of other words will decrease. In other words, relevancy of the error Tibetan correspondence will decrease. After each iteration, the difference between the correct and error Tibetan correspondence words gets bigger and bigger. This is the reason why we use IRE method.

4.3 Dictionary and word alignment method

To increase the overall performance of identification of Tibetan BaseNP, Dictionary and word alignment based (D&WA, hereafter) method are presented. Zhang et al. (2006) proposed a phrase alignment method. Their work obtains the translation head-phrase according to dictionary-based word alignment，and statistical translation boundary is determined based on the translation extending confidence.

Inspired by this research, we use the basic idea of head-phrase extension. The key problem is how to get the head-phrase and how to extend it to a correct Tibetan BaseNP. In other words, we decompose the identification of Tibetan BaseNP into head-phrase extraction and head-phrase extension steps. D&WA method use different way from (Zhang et al., 2006) in both steps.

4.3.1 Tibetan head-phrase extraction

Bilingual dictionary with 135,000 word pairs is used as an additional resource. Dictionary provides reliable alignments, but its coverage rate is low. Hence, we have modified the head-phrase extraction step in (Zhang et al., 2006). Our head-phrase extraction step is no longer solely dependent on Chinese-Tibetan dictionary. When there are no corresponding entries in bilingual dictionary for a Chinese word, we will use the word alignment result from intersect heuristic for current word, because intersect heuristic can achieve higher precision without any interference. Description of modified head-phrase extraction is as follows.

Firstly, for each word in a Chinese BaseNP, we search the bilingual dictionary and get the translation words list (TWL).

- If TWL is not null, judge whether one or more member of TWL occur in the corresponding Tibetan sentence of Chinese BaseNP, and mark the positions in Tibetan sentence.
- If TWL is null, directly use the word alignment correspondence from intersect heuristic, and mark the position in Tibetan sentence.

Then, continuous words between the most left and most right position from previous step constitute the head-phrase.

4.3.2 Tibetan head-phrase extension

The next step is the determination process of statistical translation boundary called head-phrase extension. Unlike (Zhang et al., 2006), D&WA method use commonly used Mutual information (MI) and t-value to determine left and right boundary of Tibetan BaseNP in the extension step. The formula is as follows:

$$MI(c,t) = \log \frac{P_r(c,t)}{P_r(c) \times P_r(t)} \tag{4.6}$$

$$t(c,t) \approx \frac{P_r(c,t) - P_r(c) \times P_r(t)}{\sqrt{\frac{1}{N} P_r(c,t)}} \tag{4.7}$$

Where *N* indicates the total number of sentences in bilingual corpus; *c* indicates Chinese phrase, *t* indicates Tibetan word; $P_r(c,t)$ denotes the co-occurrence probability of *c* and *t*. $P_r(c)$ and $P_r(t)$ denotes the occurrence probability of *c* and *t* respectively. For each Chinese BaseNP, we calculate MI and t-value between Tibetan words in corresponding sentence and reserve these values.

Suppose T is full or partial correspondence of Chinese BaseNP, it can symbolized as formula(4.8).

$$T = w_1 w_2 \cdots w_i \cdots w_n \tag{4.8}$$

Average Mutual Information and Average T-score between Chinese BaseNP c and T are based on formula(4.9) and formula(4.10).

$$AMI(c,T) = \frac{1}{n} \sum_{i=1}^{n} MI(c,W_i) \tag{4.9}$$

$$AT(c,T) = \frac{1}{n} \sum_{i=1}^{n} t(c,W_i) \tag{4.10}$$

Definition 4: head-phrase extension confidence.

For Chinese phrase Ph_C, the head-phrase of its translation in Tibetan sentence is denoted by $Ph_T(n)$, where *n* indicates the length of head-phrase; Extend to an adjacent Tibetan word of $Ph_T(n)$ and get $Ph_T(n+1)$, so the head-phrase extension confidence C_n defined as:

$$\begin{aligned} C_n = {} & \lambda_1 \left| AMI[Ph_C(n), Ph_T(n)] - AMI[Ph_C(n), Ph_T(n+1)] \right| \\ & + \lambda_2 \left| AT[Ph_C(n), Ph_T(n)] - AT[Ph_C(n), Ph_T(n+1)] \right| \end{aligned} \tag{4.11}$$

In which AMI and AT indicates the mean of *MI* and t-value in the scope of extended Tibetan BaseNP respectively.

In the extension step, word by word calculation of extension confidence in Tibetan sentence will be held, to both sides of head-phrase. For each extension-ready Tibetan word, note the head-phrase extension confidence C_n; if C_n is greater than the threshold, current Tibetan word is accepted as a member of Tibetan BaseNP, and extension continues; when C_n is less than the threshold extension stops. Statistical translation boundary for Chinese phrase Ph_C is obtained at the end of extension under head-phrase. FIGURE 2 shows the extension process in detail.

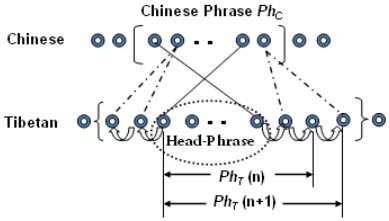


FIGURE 2 –Concept of head-phrase extension

In FIGURE 2, Chinese phrase Ph_C is in brackets; its final translation is in brace in Tibetan sentence. The extended $Ph_T(n+\omega), 0 \le \omega \le L-n$ is Tibetan BaseNP.

4.4 Sequence intersection identification method

Sequence intersection identification (SII, hereafter) method we proposed in this subsection is in accordance with the characteristics of Tibetan BaseNP. It uses intersection operation between Tibetan sentences and head-phrase extension strategy.

Analysis to the structure of Tibetan BaseNP indicates that, context based free translation style causes one Chinese phrase has different correct translation in Chinese-Tibetan sentence aligned corpus. Let's analyse the structure of following Example. "进出口货物收发货人" is "Shipper & Consignee as Declarant" in English. Three versions of Tibetan translation of "进出口货物收发货人" in Chinese-Tibetan sentence aligned corpus are given in TS1, TS2 and TS3.

TS1: ཕྱིར་གཏོང་ནང་འདེན་བྱ་རྒྱུའི་ཚོང་ཞོན་གཏོང་ལེན་བྱེད་མཁན

TS2: ཕྱིར་གཏོང་ནང་འདེན་བྱ་རྒྱུའི་ཚོང་ཞོན་ཐོབ་སྟོངས་ལེན་བྱེད་མཁན

TS3: ཕྱིར་གཏོང་ནང་འདེན་ཚོང་ཞོན་ཐོབ་སྟོངས་ལེན་མི་སྣ

During the intersection operation, "ཕྱིར་གཏོང་ནང་འདེན་" and "ཚོང་ཞོན་" is common string of TS1, TS2 and TS3. We can get "ཕྱིར་གཏོང་ནང་འདེན་བྱ་རྒྱུའི་ཚོང་ཞོན་" or "ཕྱིར་གཏོང་ནང་འདེན་ཚོང་ཞོན་" from different Tibetan sentences as head-phrase. Then we use head-phrase extension strategy to Tibetan sentence of sentence pair where "进出口货物收发货人" located and get TS1, TS2 or TS3. These different translations co-occur in our bilingual corpus. Moreover, the case is more common.

The extension step is the same as that we described in section4.3.2. The identification of Tibetan head-phrase in SII method is presented in this subsection.

4.4.1 Definition of sentence sequence intersection

Chinese-Tibetan bilingual corpus *CTBC* is composed of numerous aligned sentence pairs. Any sentence pair is notated as $SP = CS \leftrightarrow TS$, where *CS* and *TS* represent Chinese and Tibetan sentence respectively. Formula (4.12) and (4.13) give the expression of Chinese and Tibetan sentence as a word sequence.

$$CS = < C_1, C_2, \cdots, C_n > \tag{4.12}$$

$$TS = < T_1, T_2, \cdots, T_m > \tag{4.13}$$

Thus, SP can be expressed as words sequence form in formula (4.14) as below:

$$SP = CS \leftrightarrow TS = < C_1, C_2, \cdots, C_n > \leftrightarrow < T_1, T_2, \cdots, T_m > \tag{4.14}$$

Then, let's define sentence sequence intersection. Set $SP_r, SP_t \in CTBC$ are any two aligned sentence pairs. Representation of SP_r and SP_t in word sequence form is in formula (4.15) and (4.16).

$$SP_r = CS_r \leftrightarrow TS_r = < C_r, C_{r+1}, \cdots, C_{r+nr} > \leftrightarrow < T_r, T_{r+1}, \cdots, T_{r+mr} > \tag{4.15}$$

$$SP_t = CS_t \leftrightarrow TS_t = < C_t, C_{t+1}, \cdots, C_{t+nt} > \leftrightarrow < T_t, T_{t+1}, \cdots, T_{t+mt} > \tag{4.16}$$

Definition 3: Intersection of TS_r and TS_t

$$\begin{aligned} TS_r \cap TS_t = \{ & < T_{r+r1}, T_{r+r2}, \cdots, T_{r+rq} > = < T_{t+t1}, T_{t+t2}, \cdots, T_{t+tp} > \\ & | 0 \le r_1 < r_2 < \cdots < r_q \le n_r, 0 \le t_1 < t_2 < \cdots < t_p \le m_t \} \end{aligned} \tag{4.17}$$

In formula(4.17), the result of $TS_r \cap TS_t$ is a set of common substring of TS_r and TS_t. New subscripts $r_1, r_2 \cdots\cdots r_q$ and $t_1, t_2 \cdots\cdots t_p$ monotonously increase, they must be within the scope of original subscript n_r and m_t.

4.4.2 Identification of Tibetan BaseNP

SII method uses head-phrase extension like in the D&WA method. In head-phrase extraction step, SII method uses the idea of sentence sequence intersection, which is different from D&WA method.

Intuitively, if a Chinese BaseNP occurs in more than one sentence, denotes S_C, its Tibetan correspondence must occur in the aligned Tibetan sentences of S_C, denotes S_T; and sentences in S_T must have common substrings, denotes *TCS*, which is a set of multiword units including full or partial Tibetan correspondence. After sentence intersection, it is likely to get part of Tibetan BaseNP correspondence in terms of Tibetan tense, verb-endings, auxiliary word etc. Hence, one of the intersection parts must be regarded as head-phrase. It is to say, the preferred way to obtain translation of a Chinese BaseNP Q_i is searching for common substring of Tibetan sentences.

From above analysis, another form of sequence intersection for formula (4.17) is in formula(4.18).

$$TS_r \cap TS_t = T = \{T_1, T_2, \cdots, T_g\} \tag{4.18}$$

In(4.18), T is including T_j which is full or partial correspondence of Chinese BaseNP. Among these multiword units in T, we use selection function Ψ_j to determine the candidate. Commonly used mutual information (MI) and t-value statistical information are used to determine the candidate. Definition of Ψ_j for $T_j\ (1 \le j \le g)$ is given in formula(4.19).

$$\Psi_j = \lambda_1 \cdot AMI(Q_i, T_j) + \lambda_2 \cdot AT(Q_i, T_j) \tag{4.19}$$

Where *AMI* and *AT* are the means of *MI* and t-value between Chinese BaseNP Q_i and T_j respectively. $T_j\ (1 \le j \le g)$ with the highest Ψ_j is head-phrase of candidate Tibetan BaseNP. In the end, the Tibetan head-phrase is extended to BaseNP using the same extension process described in section4.3.2.

5 Experiments

5.1 Experimental corpus

The corpus used in this experiments is a domain-specific Chinese-Tibetan bilingual corpus in laws, regulations and official documents, which is the input of the Tibetan BaseNP identification. The original corpus is used to Chinese BaseNP extraction; we call it corpus1. It consists of 256,880 bilingual aligned sentence pairs including both long and short sentences. The size of Chinese corpus is 18,244 kilobytes; and that of Tibetan corpus is 58,650 kilobytes. We generate test corpus in TABLE 1 under random selection, to evaluate the proposed methods. TABLE 1 shows the basic information about the corpora. In TABLE 1, CS denotes Chinese sentence, TS denotes Tibetan sentence and SP denotes sentence pair.

First of all, Chinese sentences in corpus1 are parsed by Stanford parser, and NPs are extracted. The number of Chinese BaseNP in corpus1 is 422,146 without any pre-processing. After duplicate removal, it decreases to 255,249 which is shown in the last column of TABLE 1, where CBNP denotes Chinese BaseNP. The size of Chinese BaseNP from corpus1 is too large. In order to quantify the result, Tibetan BaseNP identification is tested on the small size of test corpus, because we need the manual reference for those Chinese BaseNPs at present. The number of Chinese BaseNP in test corpus is 394 before pre-processing. We take no account of one word NP. After filtration of one word NP and deletion of error parsed NP, we get 212 Chinese BaseNP from the test corpus.

Corpora	CS(KB)	TS(KB)	Number of SP	Number of CBNP
corpus1	18,244	58,650	256,880	255,249
Test corpus	37	149	378	212

TABLE 1 – Information about corpora

5.2 Evaluation

We define the Coverage, quasi-precision and precision to evaluate the experimental results.

$$Coverage = \frac{N_1}{N} \times 100\% \tag{5.1}$$

$$Quasi - Precision = \frac{N_2 + N_3}{N_1} \times 100\% \tag{5.2}$$

$$Precision = \frac{N_3}{N_1} \times 100\% \tag{5.3}$$

Where, N denotes the number of Chinese BaseNP for test. N_1 denotes the total number of Chinese BaseNP for which we obtain its correspondence. N_2 denotes the number of Chinese BaseNP which obtained partial correct correspondence . N_3 denotes the number of Chinese BaseNP which obtained full correct correspondence . We ask Tibetan scholar to provide us reference for Chinese BaseNP in size 212 for test, then automatically judge N_1, N_2, and N_3.

5.2.1 Different word alignment heuristics for WA method

A number of different word alignment heuristics are used in WA method. The options are:

- intersect
- union
- grow-diag-final-and
- grow-diag-final
- grow-diag

Different heuristic may show better performance for a specific language pair or corpus, the experimental results of Tibetan BaseNPs are shown in TABLE 2.

heuristic	Coverage	Quasi-precision	Precision
intersect	0.9292	0.7716	0.4975
union	1.0	0.7972	0.5165
grow-diag-final-and	0.9811	0.7692	0.5096
grow-diag-final	1.0	0.7972	0.5212
grow-diag	1.0	0.7972	0.5142

TABLE 2 –Results of WA method using different word alignment heuristics

TABLE 2 shows that, from the overall perspective, grow-diag-final heuristic outperforms others, and intersect heuristic gets the lowest performance.

On analysis, the reason for lower precision is as follows.

- Some of determinative modifier in Chinese noun phrases is verb in test set. The target language is Tibetan, which is morphologically rich language with ample variations in terms of tense, verb-endings, auxiliary word etc. These lead to discontinuous Tibetan translation.
- Chinese and Tibetan word segmentation is in different granularity.
- The quasi-precision of five heuristic is more even, however, there are some boundary interference like stop words in word alignment result all but intersect heuristic.
- Giza++is a tool based on statistics; therefore, it does not work so well on low frequency phrases.

The finding by analysis is that the word alignment result from heuristics except intersect heuristic interfered by tense, verb-endings, auxiliary word or stop word at the boundary. Yet the results from intersect heuristic consist of one or more Tibetan words without any interference. Hence we select the word alignment results of intersect heuristic in D&WA method to supplement to bilingual dictionary. It is proved that modification to automatic Tibetan BaseNP candidate under partial correspondence turns out to be effective than pure manual translation. Consequently, the best overall performance will be regarded as baseline method in next subsection.

5.2.2 Different methods for Tibetan BaseNP identification

The motivation of this paper is produce referable Tibetan BaseNP with best overall performance. To compare the four proposed Tibetan BaseNP identification method, WA methods with grow-diag-final heuristic is selected as baseline. TABLE 3 shows the results of four methods which are proposed in this paper for Tibetan BaseNP identification.

Methods	Coverage	Quasi-precision	Precision
WA method(Baseline)	1.0	0.7972	0.2453
IRE method	0.9764	0.8261	0.4203
D&WA method	0.9670	0.8732	0.4976
SII method	1.0	0.8821	0.5283

TABLE 3 –Results of different methods

The precision of IRE method is higher than baseline, because IRE method is able to filter some interference. However, precision of IRE method is influenced by some of the high frequency words and different granularity of Chinese and Tibetan words segmentation. Its coverage is medium due to correct correspondences to the low

frequency phrases. In D&WA method, Chinese-Tibetan word dictionary as auxiliary resource significantly increases the precision. Combination of dictionary and intersect heuristic word alignment improves the coverage of D&WA method. In SII method, we use intersection of Tibetan sentences to improve the coverage and quasi-precision. During intersection, some interference on boundary like verb-endings, auxiliary word are filtered; meanwhile, the step for head-phrase identification does not rely on statistics, so it works even on low frequency phrases. The overall result of SII method outperforms other proposed methods. Obviously, our test corpus is in small size, we are working on the further verification of the framework on large-scale Tibetan BaseNP identification.

Conclusion and perspectives

We are in the initial stage of identification of Tibetan base noun phrase. At present, you know, Tibetan POS tagger, Tibetan Treebank or annotated corpus is not available. On the basis of the existing resources of our group, we take the BaseNP identification as a translation problem. Four methods, namely word alignment, iterative re-evaluation, dictionary and word alignment, and sequence intersection method are applied to identify Tibetan BaseNP. We define the Coverage, quasi-precision and precision as metrics to evaluate the experimental results. As a result, the best one (SII method) achieves 1.0, 0.8821 and 0.5283 respectively on the test corpus. Compared with English or Chinese BaseNP identification work, the proposed methods doesn't get the best score, but the approach is very novel to Tibetan BaseNP identification. Due to the lack of resources like POS tagger and previous technology, the result is acceptable.

In the future, on one hand, we will improve the coverage to identify more potential BaseNP. Methods proposed in this paper need further validation in large-scale corpus. On the other hand, we will make more research on the Tibetan BaseNP templates using grammatical rules to produce a high quality results. It means that Tibetan parts-of-speech tagging is one of our future direction too.

Acknowledgments

We are grateful to three anonymous reviewers for their insightful comments that helped us improve the quality of the paper. We are thankful for Guancho Dorjie providing reference of Tibetan BaseNP in experiment part. The research is partially supported by National Science and Technology Major Project (No.2010ZX01036-001-002, No.2010ZX01037-001-002), National Science Foundation (No.61202219, No.61202220), Major Science and Technology Projects in Press and Publishing (No.0610-1041BJNF2328/23, No.0610-1041BJNF2328/26), and CAS Action Plan for the Development of Western China (No.KGCX2-YW-512).

References

Abney, S.P. (1991). Parsing By Chunks. In (Robert Berwick, Steven Abney and Carol Tenny, 1991) *Principle-Based Parsing*, Kluwer Academic Publishers.

Ando, R.K. and Zhang, T. (2005). A High-Performance Semi-Supervised Learning Method for Text Chunking. Kevin Knight. In *Proceedings of the 43rd Annual Meeting of ACL*, pages 1-9. Ann Arbor, Michigan.

Church, K.W. (1998). A stochastic parts program and noun phrase parser for unrestricted text. In *Proceedings of 2nd Conference on Applied Natural Language* Processing (ANLC '88), pages 136-143.

Dempster, A.P., Laird, N.M., and Rubin, D.B. (1977). Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society*, B39(1):1-38.

Fei, S. and Pereira, F. (2003). Shallow Parsing with Conditional Random Fields. In (*Eduard Hovy, 2003*) *Proceedings of HLT-NAACL*, pages 134-141. Edmonton, Alberta.

Huang, X., Sun, H.K., Jiang, D., Zhang, J.C., and Tang, L.M. 2005. The Types and Formal Markers of Nominal Chunks in Contemporary Tibetan. In *proceedings of the 8th Joint Conference on computational linguistics* (JSCL'2005).

Jiang, D. (2003a). On syntactic chunks and formal markers of Tibetan. *Minority Languages of China*,(3):30-39.

Jiang, D. and Long, C.J. (2003b). The Markers of Non-finite VP of Tibetan and its Automatic Recognizing Strategies. In *Proceedings of the 20th International Conference on Computer Processing of Oriental Languages* (ICCPOL'2003).

Kudo, T. and Matsumoto, Y. (2001). Chunking with support vector machine. In *Proceedings of the second meeting of the North American Chapter of the Association for Computational Linguistics on Language technologies*(NAACL '01),pages 1-8.

Lafferty, J., McCallum, A., and Pereira, F. (2001). Conditional random Fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of ICML 2001*, pages 282-289.

Li, H., Webster, J.J., Kit, C.Y., and Yao, T.S. (2003). Transductive HMM based Chinese Text Chunking. *IEEE NLP-KE 2003*, pages 257-262. Beijing.

Liu, D.M. (2004). Chinese-English bilingual parallel corpus alignment method. Graduate dissertation. China, Shanxi University.

Och, F.J. and Ney, H. (2003). A Systematic Comparison of Various Statistical Alignment Models. *Computational Linguistics* 29(1): 19-51.

Ramshaw, L.A. and Marcus, M.P. (1995). Text Chunking using Transformation-Based Learning. In *Proceedings of the Third ACL Workshop on Very Large Corpora*, pages 82–94. Cambridge MA, USA.

Sang, E.F.T.K. and Buchholz, S. (2000). Introduction to the CoNLL-2000 shared task: Chunking. In *Proc. CoNLL-2000*, pages 127-132.

Xu, F., Zong, C.Q., and Zhao, J. (2006). A Hybrid Approach to Chinese Base Noun Phrase Chunking. In *Proceedings of the Fifth SIGHAN Workshop on Chinese Language Processing*, pages 87–93, Sydney.

Zhang, C.X., Li, S., and Zhao, T.J. (2006). Phrase Alignment Based on Head-Phrase Extending. *Journal of Computer Research and Development*, 43(9):1658-1665.

Zhang, T., Damerau, F., and Johnson, D. (2001).Text Chunking using Regularized Winnow. In *Proceedings of the 39rd Annual Meeting of ACL* (ACL '01), pages 539-546. Morgan Kaufmann Publishers.

Zhang, Y.Q. and Zhou, Q. (2002). Chinese Base-Phrases Chunking. In *Proceedings of the First SIGHAN Workshop on Chinese Language Processing*, (SIGHAN '02), pages 1-5.

Zhao, J. and Huang, C.L. (1998). A Quasi-Dependency Model for Structural Analysis of Chinese BaseNPs. In *36th Annual Meeting of the ACL and 17th International Conference on Computational Linguistics*, pages 1-7.

Zhao, J. and Huang, C.N. (1999), The model for Chinese baseNP structure analysis, *Chinese Journal of Computers*, 22(2):141-146.

Zhou, Y.Q., Guo, Y.K., Huang, X.L., and Wu, L.D. (2003). Chinese and English Base NP Recognition on a Maximum Entropy Model. *Journal of Computer Research and Development*. 140(13):440-446.

A Pipeline Arabic Named Entity Recognition Using a Hybrid Approach

Mai Mohamed Oudah[1] *Khaled Shaalan*[1,2]
(1) The British University in Dubai, UAE
(2) (Fellow) School of Informatics University of Edinburgh, UK

Oudah.Mai@gmail.com, Khaled.Shaalan@buid.ac.ae

ABSTRACT

Most Arabic Named Entity Recognition (NER) systems have been developed using either of two approaches: a rule-based or Machine Learning (ML) based approach, with their strengths and weaknesses. In this paper, the problem of Arabic NER is tackled through integrating the two approaches together in a pipelined process to create a hybrid system with the aim of enhancing the overall performance of NER tasks. The proposed system is capable of recognizing 11 different types of named entities (NEs): Person, Location, Organization, Date, Time, Price, Measurement, Percent, Phone Number, ISBN and File Name. Extensive experiments are conducted using three different ML classifiers to evaluate the overall performance of the hybrid system. The empirical results indicate that the hybrid approach outperforms both the rule-based and the ML-based approaches. Moreover, our system outperforms the state-of-the-art of Arabic NER in terms of accuracy when applied to ANERcorp dataset, with f-measures 94.4% for Person, 90.1% for Location, and 88.2% for Organization.

KEYWORDS : Natural Language Processing, Named Entity Recognition, Machine Learning.

Title in Arabic

تنسيقٌ متتالٍ في التعرف على أنماط الأسماء العربية من خلال استخدام المنهج الهجين

Abstract in Arabic

تم بناء معظم أنظمة التعرف على أنماط الأسماء العربية من خلال تبني منهجية القواعد أو تبني المنهجية المبنية على تعلم الآلة، بما فيهما من نقاط قوة وضعف. في هذه الورقة، عملية التعرف على أنماط الأسماء في اللغة العربية يتم معالجتها من خلال دمج المنهجيتين معًا في تنسيقٍ متتالٍ لتشكيل المنهج الهجين في محاولة لتحسين أداء مهام التعرف على أنماط الأسماء. النظام المُقترح قادر على التعرف على 11 نوعًا مختلفًا من أنماط الأسماء بما في ذلك أسماء الأشخاص، والأماكن، والمنظمات، والتواريخ، والأوقات، والأسعار (الأموال)، والمقاييس (المقادير القياسية)، والنسب المئوية، وأرقام الهواتف، وردمك (الرقم الدولي المعياري للكتاب)، وأسماء الملفات. وقد تم إجراء تجارب مكثفة باستخدام ثلاث مصنفات مختلفة تُطَبّق تعلم الآلة لتقييم أداء النظام الهجين. تُظهر النتائج التجريبية تفوق المنهج الهجين على كلٍ من المنهج المبني على القواعد والمنهج المبني على تعلم الآلة. يتفوق نظامنا الهجين على أفضل الأنظمة المنشورة في الدوريات العلمية في مجال التعرف على أنماط الأسماء العربية من حيث الدقة عند تطبيق نظامنا على مجموعة البيانات "أنيركورب" بنتيجة معدلات توافقية قدرها: 94.4% في حالة أسماء الأشخاص، 90.1% في حالة أسماء الأماكن، و 88.2% في حالة أسماء المنظمات.

KEYWORDS in Arabic

معالجة اللغات الطبيعية، التعرف على أنماط الأسماء، تعلم الآلة

Proceedings of COLING 2012: Technical Papers, pages 2159–2176,
COLING 2012, Mumbai, December 2012.

1 Introduction

Named Entity Recognition (NER) is the task of detecting and classifying proper names within texts into predefined types, such as Person, Location and Organization names (Nadeau and Sekine, 2007), in addition to the detection of numerical expressions, such as date, time, price and phone number. Machine Translation, Information Retrieval and Question Answering are good examples of Natural Language Processing (NLP) applications that employ NER as an important preprocessing step to enhance the overall performance. In the literature, three types of approaches are used to develop NER systems: rule-based approach, machine learning (ML) based approach and hybrid approach. The rule-based approach relies on handcrafted local grammatical rules, while ML-based approach takes advantage of the ML algorithms that utilize sets of features extracted from datasets annotated with NEs for building NER systems. The hybrid approach combines rule-based approach with ML-based approach together in a pipelined process to improve the overall performance of the system.

Arabic is the official language in the Arab world where more than 300 million people speak Arabic as their native language (Shaalan, 2010). Arabic is a Semitic language and one of the richest natural languages in the world in terms of morphological inflection and derivation. Interest in Arabic NLP has been gaining momentum in the past decade, and some of the tasks have proven to be challenging especially when it comes to Information Extraction due to the language's complex and rich morphology. NER for Arabic has received some attention recently, yet opportunities for improvement in performance are still available. A number of Arabic NER systems have been developed using two types of approaches: the rule-based approach, notably NERA system (Shaalan and Raza, 2008), and the ML-based approach, notably ANERsys 2.0 (Benajiba and Rosso, 2007). Rule-based NER systems rely on handcrafted grammatical rules written by linguists. Therefore, any maintenance applied to rule-based systems is labour-intensive and time consuming especially if linguists with the required knowledge and background are not available. On the other hand, ML-based NER systems utilize ML techniques that require large tagged datasets for training and testing. An advantage of the ML-based NER systems is that they are updatable with minimal time and effort as long as sufficiently large datasets are available. The lack of linguistic resources creates a critical obstacle when it comes to Arabic NLP in general and Arabic NER in particular.

In this paper, the problem of Arabic NER is tackled through integrating the ML-based approach with the rule-based approach to develop a hybrid system in an attempt to enhance the overall performance. To the best of our knowledge, only one recent Arabic NER system (Abdallah, Shaalan and Shoaib, 2012) has adopted the hybrid approach in order to recognize three types of named entities (NEs) including Person, Location and Organization. Abdallah et al. (2012) have used only one ML technique (i.e. Decision Trees) within their system. Our research aims to develop an Arabic hybrid NER system that has the ability to extract 11 different types of NEs including Person, Location, Organization, Date, Time, Price, Measurement, Percent, Phone Number, ISBN and File Name. We extend the ML feature space to include morphological and contextual information. We test three ML algorithms (Decision Trees, Support Vector Machines, and Logistic Regression), and our results show significant performance gains over the state of the art.

The proposed system is composed of two main components: a rule-based component and a ML-based component. The rule-based component is a reproduction of an Arabic rule-based NER

system (Shaalan and Raza, 2008) with modifications and additions in order to enhance the performance. The ML-based component utilizes the ML techniques that have been used successfully in similar NER for other languages to generate a classification model for Arabic NER trained on annotated datasets. The annotated datasets are presented to the ML-based component through a set of features. The feature set is selected to optimize the performance of the ML-based component as much as possible. Two types of linguistic resources are collected and acquired: gazetteers (i.e. predefined lists of NEs or keywords) and corpora (i.e. datasets). Extensive experiments are conducted to evaluate the proposed hybrid system on different dimensions.

The structure of the remainder of this paper is as follows. Section 2 provides some background on NER. Section 3 gives a literature review of NER. Section 4 describes the process followed for data collection. Section 5 illustrates the architecture of the proposed NER system and then describes in details the main components. The evaluation experiments and the results are reported and discussed in Section 6. Finally, a conclusion and proposed future work extension are provided.

2 Background

2.1 NER and NLP Applications

In the 1990s, at the Message Understanding Conferences (MUC) in particular, the task of NER was firstly introduced and given attention by the community of research. Three main NER subtasks were defined at the 6th MUC: ENAMEX (i.e. Person, Location and Organization), TIMEX (i.e. temporal expressions), and NUMEX (i.e. numerical expressions). Customized NER system may require more sub-divisions in one or more of the NER subtasks to fulfil the system goals and objectives, e.g. Location NEs may have sub-types as City, Country, River, Road, etc.

The role of NER within NLP applications differs from one application to another. Examples of NLP applications which find the functionalities of NER useful for their purposes are Information Retrieval, Machine Translation, Question Answering and Text Clustering (Cowie and Wilks, 1996).

- **Information Retrieval (IR).** IR is the task of identifying and retrieving relevant documents out of a database of documents according to an input query (Benajiba, Diab and Rosso, 2009a). There are two possible ways that IR can benefit from NER: 1) recognizing the NEs within the query, 2) recognizing the NEs within the documents to extract the relevant documents taking into consideration their classified NEs. For example, if the input query has the word "مايكروسوفت" maAykruwsuwft[1] "Microsoft", an Organization NE, any documents that include Microsoft is considered relevant and retrieved.
- **Machine Translation (MT).** MT is the task of translating a text into another natural language. NEs need special handling in order to be translated correctly. Hence, the quality of the NE translation component would become an integral part that enhances the performance of the overall MT system (Babych and Hartley, 2003). In the translation from Arabic to Latin languages, such as English, Person names (NEs) can also be found as regular words (non-NEs) in the language without any distinguishing

[1] We used Habash-Soudi-Buckwalter transliteration scheme (Habash, Soudi and Buckwalter, 2007)

orthographic characteristics between the two surface forms. For example, the surface word "وفاء" wafaA' can be used as an adjective that means trustfulness and loyalty, and also as a Person name.

- **Question Answering (QA).** QA application is closely related to IR but with more sophisticated results. A QA system takes questions as input and gives in return concise and precise answers. NER can be exploited in recognizing NEs within the questions to help identifying the relevant documents and then extracting the correct answers (Hamadene, Shaheen and Badawy, 2011; Molla, Zaanen and Smith, 2006). For instance, the NE "الشرق الأوسط" Alšarq AlÂwsaT "Middle East" may be classified as an Organization (i.e. Newspaper) or as a Location according to the context. Hence, the proper classification for the NE will help targeting the relevant group of documents that answer the given query.
- **Text Clustering (TC).** TC may exploit NER in ranking the resulted clusters based on a ratio of entities that is associated with each cluster (Benajiba et al., 2009a). This is reflected in enhancing the process of analyzing the nature of the clusters and also improving the clustering approach in terms of the selected features. For example, Time expressions along with Location NEs can be utilized as factors that give an indication of *when* and *where* the events mentioned in a cluster of documents have happened.

2.2 Arabic Language Characteristics

Applying NLP tasks in general and NER task in particular is very challenging when it comes to Arabic because of its particularities and unique nature. The main characteristics of Arabic that pose non-trivial challenges for NER task are as follows:

- **No Capitalization:** Capitalization is not a feature of Arabic script unlike the European languages where an NE usually begins with a capital letter. Therefore, the usage of the capitalization feature is not an option in Arabic NER. However, the English translation of Arabic words may be exploited in this respect (Farber, Freitag, Habash and Rambow, 2008).
- **The Agglutinative Nature:** Arabic language has a high agglutinative nature in which a word may consist of prefixes, lemma and suffixes in different combination, and that results in a very complicated morphology (AbdelRahman, Elarnaoty, Magdy and Fahmy, 2010).
- **No Short Vowels:** Short vowels, or diacritics, are needed for pronunciation and disambiguation. However, most modern Arabic texts do not include diacritics, and therefore, a word form in Arabic may refer to two or more different words or meanings according to the context they appear, creating a one-to-many ambiguity.
- **Spelling Variants:** In Arabic script, the word may be spelled differently and still refers to the same word with the same meaning, creating a many-to-one ambiguity. For example, the word جرام jrAm 'Gram' can also be written as غرام grAm with the same meaning.
- **Lack of Linguistic Resources:** There is a limitation in the number of available Arabic linguistic resources that are free for research purposes, and many of those available are not suitable for Arabic NER tasks due to the absence of NEs annotations in the datasets or the size of the datasets which may not be sufficiently large. The Arabic gazetteers are rare as well and limited in size. Therefore, researchers tend to build their own Arabic linguistic resources in order to train and evaluate Arabic NER systems.

3 Literature Review

NER revolves around two main goals: 1) the detection of NEs 2) the extraction of those NEs in the form of different predefined types. Three main approaches are used to fulfill those two goals: the rule-based approach, the ML-based approach and the hybrid approach.

3.1 Rule-Based NER

Rule-based NER systems depend on handcrafted linguistic rules to identify NEs within texts using linguistic and contextual clues and indicators (Shaalan and Raza, 2007). Such systems exploit gazetteers/dictionaries as auxiliary clues to the rules. The rules are usually implemented in the form of regular expressions or finite state transducers (Mesfar, 2007). The maintenance of rule-based systems is not a straightforward process since experienced linguists need to be available to provide the system with the proper adjustments (Petasis et al., 2001). Thus, any adjustment to such systems is labour intensive and time consuming.

Maloney and Niv (1998) have presented TAGARAB system which is one of the early attempts to tackle Arabic NER. It is a rule-based system where a pattern matching engine is combined with a morphological tokenizer to recognize Person, Organization, Location, Number and Time. The empirical results show that combining NE finder with a morphological tokenizer outperforms the individual NE finder in terms of accuracy when applied to random datasets from AI-Hayat.

Mesfar (2007) has developed an Arabic component under NooJ linguistic environment to enable Arabic text processing and NER. The component consists of a tokenizer, morphological analyzer and NE finder. The NE finder exploits a set of gazetteers and indicator lists to support rules construction. The system identifies NEs of types: Person, Location, Organization, Currency, and Temporal expressions. The system utilizes the morphological information to extract unclassified proper nouns and thereby enhance the overall performance of the system.

Another work adopting the rule-based approach for NER is the one developed by Shaalan and Raza called PERA (2007). PERA is a grammar-based system which is built for identifying Person names in Arabic scripts with high degree of accuracy. PERA is composed of three components: gazetteers, grammars and filtration mechanism. Whitelists of complete Person names are provided in the gazetteer component in order to extract the matching names regardless of the grammars. Afterwards, the input text is presented to the grammar, which is in the form of regular expressions, to identify the rest of Person NEs. Finally, the filtration mechanism is applied on NEs detected through certain grammatical rules in order to exclude invalid NEs. PERA achieved satisfactory results when applied to the ACE and Treebank Arabic datasets.

As a continuation of Shaalan and Raza (2007) research work, NERA system was introduced in Shaalan and Raza (2008; 2009). NERA is a rule-based system that is capable of recognizing NEs of 10 different types: Person, Location, Organization, Date, Time, ISBN, Price, Measurement, Phone Numbers and Filenames. The implementation of the system was in the FAST ESP framework, where the system has three components as the PERA system with the same functionalities to cover the 10 NE types. The Authors have constructed their own corpora from different resources in order to have a representative number of instances for each NE type.

Elsebai et al. (2009) have proposed a rule-based NER system that integrates pattern matching with morphological analysis to extract Person names from Arabic text. The pattern matching engine utilizes lists of keywords without using predefined lists of Person names. Zaghouani

(2012) has also introduced a rule-based system for Arabic NER (RENAR) to extract Person, Location and Organization NEs. The system is composed of three phases: 1) morphological preprocessing, 2) looking up known NEs and 3) using local grammar to extract unknown NEs. According to the empirical results, RENAR outperforms ANERsys 1.0 (Benajiba et al., 2007), ANERsys 2.0 (Benajiba and Rosso, 2007) and LingPipe[2] in extracting Location NEs when applied to ANERcorp dataset, while LingPipe outperforms RENAR in extracting Person and Organization NEs.

3.2 Machine Learning Based NER

ML-based NER systems take advantage of the ML algorithms in order to learn NE tagging decisions from annotated texts. The most common ML techniques used for NER are Supervised Learning (SL) techniques which represent the NER problem as a classification task and require the availability of large annotated datasets. Among the most common SL techniques utilized for NER are Support Vector Machines (SVM), Conditional Random Fields (CRF), Maximum Entropy (ME), Hidden Markov Models (HMM) and Decision Trees (Nadeau and Sekine, 2007).

Benajiba et al. (2007) have developed an Arabic NER system, ANERsys 1.0, which uses ME. The authors have built their own linguistic resources: ANERcorp (i.e. an annotated corpus) and ANERgazet (i.e. gazetteers). The features used by the system are lexical, contextual and gazetteers features. The system can recognize four types of NEs: Person, Location, Organization and Miscellaneous. The ANERsys 1.0 system used to have difficulties with detecting NEs that are composed of more than one token/word; hence Benajiba and Rosso (2007) developed ANERsys 2.0, which employs a 2-step mechanism for NER: 1) detecting the start and the end points of each NE, 2) classifying the detected NEs. Benajiba and Rosso (2008) have applied CRF instead of ME as an attempt to improve the performance. The feature set used in ANERsys 2.0 was used in the CRF-based system. The features are POS tags and base phrase chunks (BPC), gazetteers and nationality. The CRF-based system achieves higher results in terms of accuracy.

Benajiba et al., (2008a) have developed another NER system based on SVM. The features used are contextual, lexical, morphological, gazetteers, POS-tags and BPC, nationality and the corresponding English capitalization. The system has been evaluated using ACE Corpora and ANERcorp. The best results are achieved when all the features are considered.

A simplified feature set has been proposed by Abdul-Hamid and Darwish (2010) to be utilized in Arabic NER. They proposed a NER system based on CRF to recognize three types of NEs: Person, Location and Organization. The system considers only surface features (i.e. leading and trailing character n-gram, word position, word length, word unigram probability, the preceding and succeeding words n-gram and character n-gram probability) without taking into consideration any other type of features. The system is evaluated using ANERcorp and ACE2005 dataset. The results show that the system outperforms the CRF-based NER system of Benajiba and Rosso (2008).

Benajiba et al, (2008b) investigated the sensitivity of different NE types to various types of features, i.e. in Benajiba et al., (2008a). They build multiple classifiers for each NE type adopting SVM and CRF approaches. ACE datasets are used in the evaluation process. According to their results, it cannot be stated whether CRF is better than SVM or vice versa in Arabic NER. Each

[2] LingPipe is available on http://alias-i.com/lingpipe/

NE type is sensitive to different features and each feature plays a role in recognizing the NE in different degrees. Further studies (i.e. Benajiba et al., 2009a; 2009b) have confirmed as well the importance of considering language independent and language specific features in Arabic NER.

AbdelRahman et al. (2010) integrated two ML approaches to handle Arabic NER including CRF and bootstrapping pattern recognition. The feature set used with the CRF classifier includes word-level features, POS tag, BPC, gazetteers and morphological features. The system is developed to extract 10 types of NEs: Person, Location, Organization, Job, Device, Car, Cell Phone, Currency, Date and Time. The results show that the system outperforms LingPipe NE recognizer when both are applied to ANERcorp dataset.

3.3 Hybrid NER

The hybrid approach integrates the rule-based approach with the ML-based approach in order to optimize the overall performance (Petasis et al., 2001). The direction of the processing flow may be from the rule-based system to the ML-based system or vice versa.

To the best of our knowledge, there is only one hybrid NER system for Arabic which has been recently developed by Abdallah, et al. (2012). The hybrid system is capable of identifying Person, Location and Organization NEs. The rule-based component is a re-implementation of the NERA system (Shaalan and Raza 2008) using the GATE tool, while the ML-based component utilizes decision trees to build the NE classifier. Each token/word is represented with a vector of features including the rule-based decisions as a feature. The other features considered are word's length, POS tag, Noun flag (i.e. a binary feature to indicate whether POS tag is Noun or not), gazetteers, statement-end flag, prefix and suffix features. The experimental results show that the hybrid system outperforms the CRF-based NER system built by Benajiba and Rosso (2008) when applied to ANERcorp dataset.

The hybrid NER proves to be feasible and requires further investigations to enhance the scope and improve the overall performance. In this paper, we contribute to hybrid NER for Arabic both in width and depth. We handle the recognition of 11 types of NEs including Person, Location, Organization, Date, Time, Price, Percent, Phone Number, Measurement, ISBN and File Name with high degree of accuracy. We investigate three different ML approaches including Decision Trees (Orphanos, Kalles, Papagelis and Christodoulakis, 1999), SVM (Vapnik, 1995) and Logistic Regression (Hastie, Tibshirani and Friedman, 2009) along with different types of features (including contextual and morphological information) in different combinations to find the feature sets with the optimal performance.

4 Data Collection

Various linguistic resources are necessary in order to develop the proposed Arabic NER system with scope of 11 different categories of NEs. The linguistic resources are of two main categories: corpora and gazetteers. The corpora used in this research are a combination of licensed and free linguistics resources. The licensed linguistics resources[3] are Automatic Content Extraction (ACE) corpora and Arabic Treebank (ATB) Part1 v 2.0 dataset. While the free linguistic resource is: ANERcorp[4] dataset which is freely available for research purposes. In the literature, these

[3] Available for us under license agreement from the Linguistic Data Consortium (LDC)
[4] Available to download on http://www1.ccls.columbia.edu/~ybenajiba/downloads.html

linguistics resources are commonly used for evaluation and comparing with existing systems. We have also built our own corpus for training and evaluating certain types of NEs that were not sufficiently covered, including file names, phone numbers and ISBN numbers. The dataset files have been prepared and annotated using our tag schema and in XML format. Our tag schema includes 11 named entity tags; one for each NE type.

The ACE training datasets covered are Newswire (NW) and Broadcast News (BN). ANERcorp is an annotated dataset built by Yassine Benajiba (Benajiba et al., 2007). Arabic Treebank Part1 v. 2.0 dataset (Maamouri et al., 2003) has no NE annotations and originally designed to support POS tagging in Arabic NLP. Therefore in this research, the ATB dataset has been manually annotated in order to support the Arabic NER task. Our study indicates that the previously listed datasets indicate that they do not include annotation for NEs of types Phone Number, ISBN and File Name. In order to have a dataset with a representative number of NEs of certain types including Phone Number, ISBN and File Name, we acquired our own corpus from different internet resources and did the manual tagging ourselves. The total number of NEs in all datasets (i.e. the number of NE annotations that are used for training and testing purposes) is 23,929 as demonstrated in Table 1.

Dataset \ NE type		Per.	Loc.	Org.	Date	Time	Price	Measure.	Percent	Phone No.	File Name	ISBN
ACE 2003	BN	711	1292	493	58	15	17	28	35			
	NW	517	1073	181	20	1	3	14	3			
ACE 2004	BN	1865	3449	1313	357	28	105	51	54			
	NW				67	4	36	30	32			
ACE 2005	BN				154	20	163	60	42			
	NW				37	7	9	22	5			
ANERcorp		3602	4425	2025								
ATB Part1 v 2.0					431	80	168	330	75			
Our own corpus										136	160	126
Total		6695	10239	4012	1124	155	501	535	246	136	160	126

TABLE 1 – The Number of Named Entities in each Reference Dataset

Another type of linguistic resources used is the gazetteers, or dictionaries. The gazetteers for Person, Location and Organization are collected from Shaalan and Raza, (2008), while the gazetteers for the rest of the NE are prepared as part of this research. The total number of NEs/keywords in all gazetteers is 19,328.

5 The System Architecture

The Rule-based and ML-based NER approaches have their own strengths and weaknesses. In this paper, we propose a hybrid architecture that is significantly better than the rule-based or machine-learning systems individually. Figure 1 illustrates the architecture of the hybrid NER system for Arabic. The system consists of two pipelined components: rule-based and ML-based Arabic NER components. The processing goes through three main phases: 1) The rule-based NER phase, 2) The feature engineering phase, i.e. the feature selection and extraction, and 3) the ML-based NER phase.

5.1 The Rule-based Component

The rule-based component in our hybrid system is a reproduction of the NERA system (Shaalan and Raza, 2008) using GATE framework[5]. The rule-based component is built with the capability of recognizing the aforementioned 11 NEs. The percent NE type is introduced in this research and some rules are improved. The rule-based system consists of three main modules: Whitelists (or gazetteers), Grammar Rules (as a set of regular expressions), and a Filtration mechanism (blacklists of invalid NEs).

The GATE environment is used to build the rule-based component. The corpus with its documents is processed using different processing tools and resources such as a tokenizer, gazetteers and grammatical rules. Table 2 illustrates the number of gazetteers and rules implemented within each NE type. The system contains a total of 73 rules and 90 gazetteers.

	Per.	Loc.	Org.	Date	Time	Price	Measure	Percent	Phone No.	File Name	ISBN	Total
# of Gazetteers	11	20	8	12	10	8	7	3	7	3	1	**90**
# of Rules	9	20	9	7	8	4	3	1	7	3	2	**73**

TABLE 2 – The Number of Gazetteers and Rules in each NE Extractor

5.2 The ML-based Component

The ML-based component depends on two main aspects: feature engineering and selection of ML classifiers. The first aspect is the feature engineering which involves the selection and extraction of classification features. The features explored are divided into various categories: rule-based features (i.e. derived from the rule-based component's decisions), morphological features, POS features, Gazetteer features, contextual features, and word-level features. Exploring different types of features and arranging them in sets allow studying the effect of each feature set on the overall performance of the proposed system along different dimensions, including NE type and ML technique.

The second aspect concerns the ML classifier, or function, to be used in the training, testing and prediction phases. Three ML techniques have been explored and examined individually in order to reach a conclusion with regards to the best approach to work with in our hybrid NER system for Arabic. The three techniques are Decision Trees, SVM, and Logistic Regression. The first two techniques were chosen for their high performance in NER in general and Arabic NER in particular; whereas, the third technique is a new investigation that has never been used before in evaluating Arabic NER performance. In this research, WEKA[6], a comprehensive and efficient workbench with support for a large number of ML algorithms, is utilized as the environment of the ML task. The decision tree algorithm is applied using the J48 classifier, SVM with the LibSVM classifier, and Logistic Regression with the Logistic classifier.

[5] Available for free download on http://gate.ac.uk/
[6] The official website of WEKA : www.cs.waikato.ac.nz/ml/weka/

The 11 types of NEs are distributed among three groups according to their nature in which each group has a distinct feature set:

- 1st group: Person, Location and Organization NEs (aka ENAMEX)
- 2nd group: Date, Time, Price, Measurement and Percent NEs (aka TIMEX and NUMEX)
- 3rd group: Phone Number, ISBN and File Name NEs. Notice that the first two types of NE can be considered as NUMEX but they have been moved to this group intentionally because of the nature of their rules and patterns which is specific and limited.

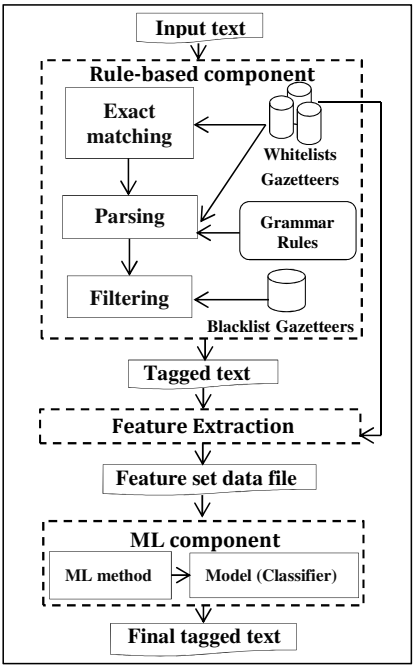


FIGURE 1 – The Architecture of the Hybrid NER System

The three groups of NEs have a generic set of classification features which are common among them, i.e. the following features are used across all three groups.

- *Rule-based features*: The NE type predicted by the rule-based component for the targeted word as well as the NE types for the two immediate left and right neighbors of the candidate word, i.e. NE type for a sliding window of size 5.
- *Morphological Features*: The set of 13 features generated by MADA[7] (Habash and Rambow, 2005).
- *POS tag*: part-of-speech tag of the targeted word estimated by MADA.

[7] MADA is Available for free download on http://www1.ccls.columbia.edu/MADA/MADA_download.html

- *Word length flag*: A binary feature to indicate whether the word length ≥ 3.
- *Dot flag*: A binary feature to indicate whether the word has adjacent dot.
- *Capitalization flag*: A binary feature to indicate the existence of capitalization information on the English gloss (translation) corresponding to the Arabic word.
- *NE type*: NE tag of the word is used along with other features for training the classification model. It is also used as a reference when calculating the accuracy scores. In the prediction phase, this feature (i.e. the NE type itself) is excluded from the selected feature set.

Besides, there are two distinct features that are used in the 1st group:

- *Nominal flag*: A binary feature to indicate whether POS tag is Noun (or Proper Noun).
- *Check Person/Location/Organization Gazetteers feature flags*: A binary feature to indicate whether the word (or left/right neighbour of targeted word) belongs to Person/Location/Organization Gazetteer(s).

Similarly, there are two distinct feature used with the 2nd group:

- *Check POS feature flags*: A binary feature to indicate whether POS tag is Noun_num (i.e. literal number word) (or Proper Noun).
- *Check Date/Time/Price/Measurement/Percent Gazetteers feature flags*: A binary feature to indicate whether the word (or left/right neighbour of targeted word) belongs to Date/Time/Price/Measurement/Percent Gazetteer(s).

Likewise, two distinct features are used with the 3rd group:

- *Nominal flag*: as described in the 1st group feature set.
- *Check Phone Number/ISBN/File Name Gazetteers feature flags*: A binary feature to represent indicate the word (or left/right neighbour of targeted word) belongs to Phone Number/ISBN/File Name Gazetteer(s).

6 Experimental Analysis

6.1 Experimental Setup

We conduct testing and evaluation experiments to test the rule-based component and compare it to the hybrid system. At the level of the hybrid system, experiments are subdivided at three dimensions: the NE type, the ML classifier used, and the inclusion/exclusion of feature groups, with the rule-based decision included as one of the feature groups as will be detailed in the following subsection. Each experiment includes a reference dataset, and an annotated dataset. The reference datasets are the initial datasets described with their tagging details in Section 4 including ACE corpora, ATB part1 v 2.0, ANERcorp and our own corpus. The reference datasets are fed into the rule-based component so that the outputs represent the annotated datasets which are exploited in the feature extraction phase to generate the feature set data files in order to be utilized by the ML-based component.

The performance of the rule-based component is evaluated using GATE built-in evaluation tool, so-called *AnnotationDiff*. This tool enables the comparison of two sets of annotations and the results are presented with the Information Extraction standard measures (i.e. precision, recall and f-measure). On the other hand, the ML approach uses three different functions (or classifiers) to

be applied to the annotated dataset, including decision trees, SVM and logistic regression approaches which are available in WEKA workbench via J48, LibSVM and Logistic classifiers respectively. In this research, 10-fold cross validation is chosen to avoid overfitting. The WEKA tool provides the functionality of applying the conventional k-fold cross-validation for evaluation with each classifier and then having the results represented in the aforementioned standard measures.

6.2 Experiments and Results

A number of experiments have been conducted to evaluate the performance of the proposed hybrid NER system when applied to different datasets in order to extract the various types of NEs applying each of the three different ML techniques. The experiments setting study the performance of the system when the contribution of all features is considered, contribution of pure ML-based features is considered, and after excluding the morphological features generated by MADA (Habash and Rambow, 2005; Roth et al., 2008), i.e. *asp, cas, enc0, gen, mod, num, per, prc0, prc1, prc2, prc3, stt, vox,* and *gloss*. In this way, the following three settings on the level of feature groups are examined:

1. All Features: all features are considered.
2. W/O RB: excluding the rule-based features (pure ML-based mode).
3. W/O MF: excluding the morphological features.

It should be noted that the baseline in all experiments is the performance of the pure rule-based component.

According to the empirical results illustrated in Table 3, the highest performance of our system in terms of Average F-measures when applied on ACE (2003-2004) NW and ANERcorp datasets to extract NEs of the 1st group (i.e. Person, Location and Organization) is achieved by J48 classifier when the 1st feature setting is used, while using J48 classifier with the 3rd setting leads to the highest performance in extracting NEs of the same group from ACE2003 BN dataset.

		ACE2003 NW	**ACE2003 BN**	**ACE2004 NW**	**ANERcorp**
		Avg. F-measure	Avg. F-measure	Avg. F-measure	Avg. F-measure
Rule-based (baseline)		0.6365	0.6087	0.4671	0.6745
J48	All Features	**0.8517**	0.8077	**0.7613**	**0.9090**
	W/O RB	0.8173	0.7633	0.7350	0.8357
	W/O MF	0.8487	**0.8203**	0.7447	0.9047
Libsvm	All Features	0.7953	**0.7653**	**0.7190**	**0.9007**
	W/O RB	0.7453	0.6307	0.6590	0.8100
	W/O MF	**0.7937**	0.7667	0.7117	0.8967
Logistic	All Features	**0.7953**	**0.7693**	**0.7170**	**0.8980**
	W/O RB	0.7577	0.6703	0.6447	0.7753
	W/O MF	0.7827	0.7620	0.7077	0.8857

TABLE 3 – The results of applying the proposed hybrid system on ACE2003 (NW & BN), ACE2004 (NW), and ANERcorp datasets in order to extract NEs of the 1st group

The results illustrated in Table 4 show that the highest performance in terms of Average F-measures when applied on ACE2003 BN, ACE2004 NW & BN, ACE2005 NW & BN and ATB Part1 v 2.0 datasets to extract NEs of the 2nd group (i.e. Date, Time, Price, Measurement and Percent) is achieved by J48 classifier when either the 1st or the 3rd feature setting is utilized, while using Logistic classifier with the 3rd feature setting leads to the highest performance in extracting NEs of the same group from ACE2003 NW dataset. The highest performance of our system in terms of Average F-measures when applied on our own corpus to extract NEs of the 3rd group (i.e. Phone Number, ISBN and File Name) is achieved by either the J48 classifier or the Logistic classifier when the 1st or the 3rd feature setting is utilized as shown in Table 5.

The experimental results show that the adaptation of the hybrid approach leads to the highest performance. It is worth noting that the results of the proposed hybrid system is very close to the results of the rule-based component when it comes to the numerical and temporal expressions, and the two approaches achieve the same results in recognizing NEs of the 3rd group. Therefore, the hybrid approach proves its suitability for the recognition of the three groups of NEs. Also, the decision trees function has proved its comparatively higher efficiency as a classifier in our Arabic hybrid NER system.

		ACE2003 NW	**ACE2003 BN**	**ACE2004 NW**	**ACE2004 BN**	**ACE2005 NW**	**ACE2005 BN**	**ATB**
		Avg. F-measure	Avg. F-measure	Avg. F-measure	Avg. F-measure	Avg. F-measure	Avg. F-measure	Avg. F-measure
Rule-based (baseline)		0.9790	1.0000	0.9766	0.9911	0.9580	0.9839	0.9812
J48	All Features	0.9842	**1.0000**	**0.9874**	**0.9962**	**0.9794**	**0.9870**	**0.9908**
	W/O RB	0.7350	0.6345	0.5742	0.6330	0.6722	0.5703	0.8240
	W/O MF	**0.9864**	**1.0000**	**0.9874**	**0.9962**	**0.9794**	**0.9870**	**0.9908**
Libsvm	All Features	0.9626	0.6985	**0.9818**	0.9090	0.9714	0.9246	**0.9862**
	W/O RB	0.8650	0.2860	0.5840	0.5077	0.8447	0.3610	0.7546
	W/O MF	**0.9672**	**0.7753**	0.9774	**0.9166**	**0.9732**	**0.9344**	**0.9862**
Logistic	All Features	0.9752	0.9280	0.9762	0.9854	0.9702	0.9666	0.9866
	W/O RB	0.6520	0.4123	0.5208	0.5418	0.6642	0.4104	0.7248
	W/O MF	**0.9908**	**0.9334**	**0.9788**	**0.9890**	**0.9774**	**0.9864**	**0.9872**

TABLE 4 – The results of applying our hybrid system on ACE2003, 2004 & 2005 (NW & BN) and ATB Part1 v 2.0 datasets when the 2nd group is the targeted group

		Phone Number	**ISBN**	**File Name**	
		F-measure	F-measure	F-measure	Avg. F-measure
Rule-based (baseline)		1	1	1	1.0000
J48	All Features	**1**	**1**	**1**	**1.0000**
	W/O RB	0.453	0.437	0.899	0.5963
	W/O MF	**1**	**1**	**1**	**1.0000**
Libsvm	All Features	0.996	1	1	0.9987
	W/O RB	0.4	0.148	0.891	0.4797
	W/O MF	0.996	1	1	0.9987
Logistic	All Features	**1**	**1**	**1**	**1.0000**
	W/O RB	0.447	0.518	0.879	0.6147
	W/O MF	**1**	**1**	**1**	**1.0000**

TABLE 5 – The results of applying our hybrid system on our own corpus when the 3rd group is the targeted group

In comparison with the results achieved by ANERsys 1.0 (Benajiba et al., 2007), ANERsys 2.0 (Benajiba and Rosso, 2007), Arabic ML-based NER system using CRF (Benajiba and Rosso, 2008) and the hybrid NER system for Arabic developed by Abdallah et al. (2012) when applied on ANERcorp, our system performs demonstrably better as illustrated by Table 6. As it can be noticed, our hybrid system outperforms the other systems in terms of F-measure in extracting Person, Location and Organization NEs from ANERcorp dataset.

System \ NE Type	Person	Location	Organization
	F-measure	F-measure	F-measure
ANERsys 1.0	0.4669	0.8025	0.3679
ANERsys 2.0	0.5213	0.8671	0.4643
CRF-based system	0.7335	0.8974	0.6576
Abdallah et al. (2012)	0.928	0.8739	0.8612
Our Hybrid System (J48)	**0.944**	**0.901**	**0.882**

TABLE 6 – The results of ANERsys 1.0, ANERsys 2.0, CRF-based system (Benajiba and Rosso, 2008) and Abdallah et al. (2012)'s system compared to our hybrid system's highest performance when applied to ANERcorp dataset

Conclusion and Future Work

The hybrid approach is most recent which integrates rule-based with ML approaches. The integration is more intuitive and linguistically motivated as it conducts an Arabic NER pipeline that combines rule-based features with other features used in machine learning. The proposed hybrid system has achieved an overall improvement of the Arabic NER performance. It is capable of recognizing 11 different types of named entities including Person, Location, Organization, Date, Time, Price, Measurement, Percent, Phone Number, ISBN and File Name. A number of extensive experiments are conducted on three different dimensions including the named entity types, the feature set (divided into groups) and the ML technique to evaluate the performance of our Arabic NER system when applied on different datasets. The experimental results show that the hybrid approach outperforms the pure Rule-based approach and the pure ML-based approach. Our hybrid NER system for Arabic outperforms the state-of-the-art of the Arabic NER in terms of f-measure when applied to ANERcorp dataset with f-measure of 94.4% for Person named entities, f-measure of 90.1% for Location named entities, and f-measure of 88.2% for Organization named entities.

In future work, we intend to enhance the gazetteers and explore the possibility of improving the system with adding more lists. There is also a space for improving the grammatical rules implemented within the rule-based component through analyzing the hybrid system's output in a way to automate the enhancement process. We are also considering the possibility of using different ML techniques other than decision trees, SVM and logistic regression and how this will impact on the overall performance of the system.

Acknowledgments

This research was funded by the British University in Dubai (Grant No. INF004-Using machine learning to improve Arabic named entity recognition).

References

Abdallah, S., Shaalan, K. and Shoaib, M. (2012). Integrating Rule-based System with Classification for Arabic Named Entity Recognition. *Proceedings of the 13th International Conference on Intelligent Text Processing and Computational Linguistics (CICLing),* Springer-Verlag, Berlin Heidelberg, pages 311-322.

AbdelRahman, S., Elarnaoty, M., Magdy, M. and Fahmy, A. (2010). Integrated Machine Learning Techniques for Arabic Named Entity Recognition. *International Journal of Computer Science Issues (IJCSI)*, Vol. 7, Issue 4, No 3, pages 27-36.

Abdul-Hamid, A. and Darwish, K. (2010). Simplified Feature Set for Arabic Named Entity Recognition. *Proceedings of the 2010 Named Entities Workshop, (ACL 2010),* pages 110-115.

Babych, B. and Hartley, A. (2003). Improving Machine Translation Quality with Automatic Named Entity Recognition. *Proceedings of the 7th International EAMT workshop on MT and other Language Technology Tools, Improving MT through other Language Technology Tools: Resources and Tools for Building MT (EAMT 2003)*, pages 1-8.

Benajiba, Y., Rosso, P. and Bened´i, J. M. (2007). ANERsys: An Arabic Named Entity Recognition system based on Maximum Entropy. *Proceedings of the 8th International Conference on Computational Linguistics and Intelligent Text Processing (CICLing-2007)*, Springer-Verlag, Berlin, Heidelberg, pages 143-153.

Benajiba, Y. and Rosso, P. (2007). ANERsys 2.0: Conquering the NER task for the Arabic language by combining the Maximum Entropy with POS-tag information. *Proceedings of Workshop on Natural Language-Independent Engineering, 3rd Indian International Conference on Artificial Intelligence (IICAI-2007)*, pages 1814-1823.

Benajiba, Y. and Rosso, P. (2008). Arabic Named Entity Recognition using Conditional Random Fields. *Proceedings of Workshop on HLT & NLP within the Arabic World (LREC 2008).*

Benajiba, Y., Diab, M. and Rosso, P. (2008a). Arabic Named Entity Recognition: An SVM-Based Approach. *Proceedings of Arab International Conference on Information Technology (ACIT 2008),* pages 16-18.

Benajiba, Y., Diab, M. and Rosso, P. (2008b). Arabic Named Entity Recognition Using Optimized Feature Sets. *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP 2008)*, pages 284-293.

Benajiba, Y., Diab, M. and Rosso, P. (2009a). Arabic Named Entity Recognition: A Feature-Driven Study. *IEEE Transactions On Audio, Speech, And Language Processing*, 17(5), pages 926-934.

Benajiba, Y., Diab, M. and Rosso, P. (2009b). Using Language Independent and Language Specific Features to Enhance Arabic Named Entity Recognition. *The International Arab Journal of Information Technology,* 6(5), pages 464- 473.

Cowie, J. and Wilks, Y. (1996). Information Extraction. *Communications of the ACM*, 39(1), pages 80-91.

Elsebai, A., Meziane, F. and BelKredim, F. Z. (2009). A Rule Based Persons Names Arabic Extraction System. *Communications of the IBIMA,* pages 53-59.

Farber, B., Freitag, D., Habash, N. and Rambow, O. (2008). Improving NER in Arabic Using a Morphological Tagger. *Proceedings of Workshop on HLT & NLP within the Arabic World (LREC 2008)*, pages 2509- 2514.

Habash, N. and Rambow, O. (2005). Arabic Tokenization, Part-of-Speech Tagging and Morphological Disambiguation in One Fell Swoop. *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 573-580.

Habash, N., Soudi, A. and Buckwalter, T. (2007). On Arabic Transliteration. *Arabic Computational Morphology: Knowledge-based and Empirical Methods,* Springer, pages 15-22.

Hamadene, A., Shaheen, M. and Badawy, O. (2011). ARQA: An Intelligent Arabic Question Answering System. *Proceedings of Arabic Language Technology International Conference (ALTIC 2011).*

Hastie, T., Tibshirani, R. and Friedman, J. (2009). *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. (2nd ed.). Springer.

Maamouri, M., Bies, A., Jin, H. and Buckwalter, T. (2003). Arabic Treebank: Part 1 v 2.0. *LDC2003T06: Linguistic Data Consortium*, Philadelphia.

Mesfar, S. (2007). Named Entity Recognition for Arabic Using Syntactic Grammars. *Proceedings of the 12th International Conference on Application of Natural Language to Information Systems*, Springer-Verlag, Berlin, Heidelberg, pages 305-316.

Mitchell, A., Strassel, S., Huang, S., and Zakhary, R. (2005). ACE 2004 Multilingual Training Corpus. *Ldc2005t09: Linguistic Data Consortium*, Philadelphia.

Mitchell, A., Strassel, S., Przybocki, M., Davis, J., Doddington, G., Grishman, R., Meyers, A., Brunstein, A., Ferro, L. and Sundheim, B. (2003). Tides Extraction (ACE) 2003 Multilingual Training Data. *Ldc2004t09: Linguistic Data Consortium*, Philadelphia.

Molla, D., Zaanen, M. and Smith, D. (2006). Named Entity Recognition for Question Answering. In *Proceedings of the 2006 Australasian Language Technology Workshop (ALTW2006),* pages 51-58.

Nadeau, D. and Sekine, S. (2007). A Survey of Named Entity Recognition and Classification. *Lingvisticae Investigationes*, 30(1), pages 3-26.

Orphanos, G., Kalles, D., Papagelis, T. and Christodoulakis, D. (1999). Decision Trees and NLP: A Case Study in POS Tagging. *Proceedings of Annual Conference on Artificial Intelligence (ACAI).*

Petasis, G., Vichot, F., Wolinski, F., Paliouras, G., Karkaletsis, V. and Spyropoulos, C. D. (2001). Using Machine Learning to Maintain Rule-based Named-Entity Recognition and Classification Systems. *Proceeding Conference of Association for Computational Linguistics*, pages 426-433.

Roth, R., Rambow, O., Habash, N., Diab, M. and Rudin, C. (2008). Arabic Morphological Tagging, Diacritization, and Lemmatization Using Lexeme Models and Feature Ranking.

Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics on Human Language Technologies: Short Papers (HLT-Short '08), pages 117-120.

Shaalan, K. (2010). Rule-based Approach in Arabic Natural Language Processing. *The International Journal on Information and Communication Technologies (IJICT),* 3(3), pages 11-19.

Shaalan, K. and Raza, H. (2007). Person Name Entity Recognition for Arabic. *Proceedings of the 5th Workshop on Important Unresolved Matters*, pages 17-24.

Shaalan, K. and Raza, H. (2008). Arabic Named Entity Recognition from Diverse Text Types. *Proceedings of the 6th International Conference on Natural Language Processing (GoTAL 2008)*, Springer-Verlag, Berlin, Heidel-berg, pages 440-451.

Shaalan, K. and Raza, H. (2009). NERA: Named Entity Recognition for Arabic. *Journal of the American Society for Information Science and Technology,* 60(8), pages 1652–1663.

Vapnik, V. N. (1995). *The Nature of Statistical Learning Theory*. Springer-Verlag New York, Inc.

Walker, C., Strassel, S., Medero, J., and Maeda, K. (2006). ACE 2005 Multilingual Training Corpus. *Ldc2006t06: Linguistic Data Consortium*, Philadelphia.

Zaghouani, W. (2012). RENAR: A Rule-Based Arabic Named Entity Recognition System. *ACM Transactions on Asian Language Information Processing,* 11(1), Article no. 2, pages 1-13.

Attribute Extraction from Conjectural Queries

Marius Paşca
Google Inc.
Mountain View, California
mars@google.com

ABSTRACT
Conjectural search queries (*is python case sensitive, is millennium stadium heated*) embody attempts by Web users to verify whether a particular property (*soluble in water?, case sensitive?, heated?*) does or does not apply to a particular instance (*iodine, python, millennium stadium*). This paper considers such queries to be a data source of attributes of open-domain classes. Conjectural attributes complement attributes encoded in human-compiled knowledge resources or automatically acquired from text by previous methods. They correspond to properties of interest to Web users, which are not necessarily stated in nominal form. Relevant properties of *Chemical elements*, *Programming languages* and *Stadiums* include whether they are *soluble in water*, *flammable* or *ductile*; *case sensitive, platform independent*, or *interpreted*; or *air conditioned, roof retractable* or *heated*, respectively. Experimental results show that relevant, conjectural attributes can be extracted from inherently-noisy queries, for a variety of open-domain classes of interest.

KEYWORDS: Class attributes, open-domain information extraction, Web search queries.

Proceedings of COLING 2012: Technical Papers, pages 2177–2190,
COLING 2012, Mumbai, December 2012.

Examples of Attributes Available in or Extracted from Various Sources, for a Sample of Classes
Food ingredients:
W: energy, dietary fiber, solubility in water
F: energy per 100g, availability, scientific name, solubility in water
D_1: species, pounds, cup, kinds, lbs, bowl
D_2: quality, part, taste, value, portion
Q_1: nutritional value, health benefits, glycemic index, varieties, calories
Q_C: gluten free?, safe?, healthy?, vegan?, halal?, fattening?, acidic?, good for skin?
Astronomical objects:
W: constellation, right ascension, spectral type, rotational velocity, orbital period, mean radius
F: category, constellation, age, periapsis, orbital period, mean radius
D_1: observations, spectrum, planet, spectra, conjunction, transit, temple, surface
D_2: surface, orbit, bars, history, atmosphere
Q_1: atmosphere, surface, gravity, diameter, mass, rotation, revolution, moons, radius
Q_C: bigger than earth?, close to the sun?, circumpolar?, capable of supporting life?
Religions:
W: fundamentals, texts, deities, sacred sites, schools, people
F: founding figures, beliefs, practices, texts, deities, sacred sites
D_1: teachings, practice, beliefs, religion spread, principles, emergence, doctrines
D_2: basis, influence, name, truths, symbols, principles, strength, practice, origin, god, defence
Q_1: basic beliefs, teachings, holy book, practices, rise, branches, spread, sects
Q_C: monotheistic?, a religion or a way of life?, peaceful?, older than hinduism?

Table 1: Examples of attributes already explicitly encoded in human-compiled knowledge resources (W=Wikipedia; F=Freebase) or extracted by various methods from text. Some of the entries in the table are also listed in (Van Durme et al., 2008) (D_1=from documents (Paşca et al., 2007), D_2=from documents (Van Durme et al., 2008), Q_1=from queries (Paşca, 2007), Q_C=from conjectural queries (this method))

1 Introduction

Motivation: Current efforts towards injecting structured knowledge into search results place a renewed emphasis on extracting, curating and serving open-domain knowledge. Resources such as Wikipedia (Remy, 2002) and Freebase (Bollacker et al., 2008) contain knowledge about classes (*Chemical elements*, *Programming languages* and *Stadiums*) and their instances (*iodine*, *python*, *millennium stadium*). Knowledge is often represented as properties or attributes of the instances, along with values for those properties. But even the largest human-curated knowledge resources may be missing at least some relevant knowledge, for some or all instances. For example, attributes representing the geographical coordinate or seating capacity are available in both Wikipedia and Freebase for many (e.g., for *millennium stadium*, *bc place*, *georgia dome*), albeit not all (e.g., *jenner park stadium*) instances of *Stadiums*. In contrast, information on whether particular *Stadiums* are *heated* or *roof retractable* is uniformly missing. Search queries such as *"is millennium stadium heated"*, *"is bc place heated"* and *"is the georgia dome heated"* suggest that such information is relevant to Web users. Populating the knowledge resource with the attribute *heated?*, for instances of *Stadiums*, would likely be more useful than, e.g., adding knowledge about whether they are *painted white*. More generally, identifying properties or attributes of interest to Web users but missing from the knowledge resource, helps allocate limited resource-development cycles to areas within the resource where their addition is most beneficial.

Contributions: This paper introduces a method for the acquisition of class attributes, from queries in the form *"<be> I A"* (e.g., *"is (georgia dome)$_I$ (heated)$_A$"*) or *"why <be> I A"* (e.g., *"why was the (tiger stadium)$_I$ (demolished)$_A$"*). Through such queries, Web users likely attempt to

verify whether - or why - a particular property (*demolished?*, *heated?*) applies to a particular instance (*tiger stadium*, *georgia dome*). In Table 1, attributes from conjectural queries are different in scope and style from attributes already encoded in human-compiled knowledge resources or attributes automatically acquired by previous methods. Examples include *good for skin?* for *Food ingredients*, *capable of supporting life?* for *Astronomical objects*, or *peaceful?* for Religions. Conjectural attributes cannot be easily captured by, and are therefore complementary to, attributes produced by previous methods, including methods targeting textual fragments in the form *"A of I"* (e.g., *"(seating capacity)$_A$ of (millennium stadium)$_I$"*) occurring in documents (Tokunaga et al., 2005) or queries (Paşca and Van Durme, 2007).

2 Extraction of Conjectural Attributes

Intuitions: The extraction of attributes from queries starts from the intuition that, if an attribute A is relevant for a class C, then users are likely to ask for the value of the attribute A, for various instances I of the class C (Paşca, 2007). The submission of fact-seeking queries such as *"what is the (seating capacity)$_A$ of (millennium stadium)$_I$"*, or the more compact *"(seating capacity)$_A$ of (millennium stadium)$_I$"*, is taken as evidence that *seating capacity* is a candidate attribute of the instance *Millennium Stadium*, and transitively a candidate attribute of the class *(Stadiums)$_C$* to which *Millennium Stadium* belongs. Attributes extracted from such queries are usually limited to noun phrases.

If an attribute A is relevant for a class C, then users are also likely to ask whether the attribute A does or does not apply to various instances I of the class C. The method introduced here takes advantage of other kinds of queries, namely conjectural, or truth-verification queries. Conjectural queries ask whether something is true or not. In comparison to fact-seeking queries, conjectural queries provide only weak evidence (conjectures) that the fact being asked about is true or not. Nevertheless, conjectural queries such as *"is (millennium stadium)$_I$ (heated)$_A$"* are taken as weak evidence that *heated?* is a candidate attribute of the instance *Millennium Stadium*, and transitively as stronger evidence a candidate attribute of the class *(Stadiums)$_C$*. Intuitively, the evidence supporting the candidate attribute is stronger for the class than it is for the instance. Users are likely to ask whether an attribute does or does not apply to an instance, based on prior knowledge that the attribute already applies to the class. If users submit the query mentioned earlier in this paragraph, it is likely because they are aware that *Stadiums* may or may not be *heated*, and would like to check whether a particular instance of *Stadiums* is. Collectively, queries asking whether an attribute applies to multiple instances of the same class are indirect evidence that the attribute does in fact apply to the class.

In addition to conjectural queries, the method also takes advantage of explanation-seeking queries, as a less frequent but more reliable source of evidence available in queries. Such queries ask for an explanation of why something is true. Queries such as *"why is (millennium stadium)$_I$ (heated)$_A$"* are intuitively more reliable sources of evidence that *Millennium Stadium*, in particular, and *Stadiums*, in general, are in fact *heated*, than queries like *"is (millennium stadium)$_I$ (heated)$_A$"* are.

Scope: Attributes extracted from conjectural queries cover multiple parts of speech, from single-word adjectives (*heated?*) and multiple-word descriptors (*open to the public?*) to noun phrases (*a retractable roof?*, *a 5 star stadium?*). On another dimension, conjectural attributes can capture properties that are objective or subjective. They include the more objective *on netflix?*, *a true story?*, *rated r?* for *Films*; but also the more subjective properties of whether *Films* are *weird?* or *funny?*. Anecdotal evidence of query frequency distribution in query logs suggests that Web users

Figure 1: Overview of extraction of conjectural attributes from Web search queries

are sometimes as interested in subjective attributes as they are in objective ones.

Referring back to Table 1, at least some of the conjectural adjectives have a nominal counterpart, which could in principle be extracted by existing extraction methods. The attribute *bigger than earth?* may be thought of as roughly equivalent to *size relative to earth*; *close to the sun?* to *distance to the sun*; *capable of supporting life* to *ability to support life*. However, even large query logs may not contain such noun phrases in the query format expected by previous methods. Moreover, some of the theoretical nominal counterparts of some conjectural attributes would be difficult to even identify manually, let alone extract with previous methods. It is less clear what the nominal counterparts are in the case of *fun?* for *Actors*, *peaceful?* for *Religions*, or *waterproof?* for *Cameras*. To summarize, conjectural attributes are arguably more diverse than attributes from existing resources or previously extracted from text.

Extraction from Queries: As illustrated in Figure 1, the proposed method takes as input a set of target classes, each of which is a set of instances; and a set of anonymized queries. Instances may be available as non-disambiguated items, that is, as strings (*python*) whose meaning is otherwise not available; or as disambiguated items, that is, as pointers to knowledge base entries with a particular, disambiguated meaning (*Python (programming language)*).

The method identifies the subset of input queries that are deemed to be conjectural queries. For this purpose, queries are matched against the patterns "*<be> I A*" and "*why <be> I A*", where I and A are a possible instance and an attribute. Other simple patterns were employed in previous

Actors, Aircraft, Animated characters, Association football clubs, Astronomical objects, Automobiles, Awards, Battles and operations of World War II, Chemical elements, Cities, Companies, Countries, Currencies by country, Digital cameras, Diseases and disorders, Drugs, Empires, Films, Flowers, Food ingredients, Holidays, Hurricanes in North America, Internet search engines, Mobile phones, Mountains, National Basketball Association teams, National parks, Newspapers, Organizations designated as terrorist, Painters, Programming languages, Religious faiths traditions and movements, Rivers, Skyscrapers, Sports events, Stadiums, Treaties, Universities and colleges, Video games, Wine

Table 2: Set of 40 Wikipedia categories used as target classes in the evaluation of attributes

work (Paşca and Van Durme, 2007) to extract attributes from text. Queries that match any of the patterns are deemed to be conjectural, but only if the query fragment corresponding to I can be matched to one of the instances of the input target classes. Queries do not need to end in a question mark, in order to be deemed conjectural. Depending on whether instances are non-disambiguated or disambiguated items, the matching from a query fragment to an instance from the input classes consists in either simple string matching; or disambiguation of I in the context of the query. When matching succeeds, the query fragment A is collected as a conjectural attribute for the instance I. In turn, attributes of a class are aggregated from attributes of individual instances of the class. The relative ranking among attributes of a class promotes attributes extracted from source queries that a) match many instances of the class, and few instances of any other classes; and b) have higher frequency in query logs.

3 Experimental Setting

Textual Data Sources: The experiments rely on a random sample of around 500 million fully-anonymized Web search queries in English. Each query is available independently from other queries, and is accompanied by its frequency of occurrence in the query logs.

Target Classes: The attributes extracted from queries are evaluated over a set of 40 target classes included in Table 2. In an effort to reuse experimental setup proposed in previous work, each of the 40 manually-compiled classes introduced in (Paşca, 2007) is mapped into the Wikipedia category that best matches it. For example, the evaluation classes *Actor*, *Mountain*, *Movie*, *Religion* and *TerroristGroup* from (Paşca, 2007) are mapped into the Wikipedia categories *Actors*, *Mountains*, *Films*, *Religious faiths traditions and movements* and *Organizations designated as terrorist* respectively. Note that the name of the Wikipedia category only serves as a convenience label for its target class, and is not otherwise exploited in any way during the evaluation. Instead, a target class consists in a sample set of Wikipedia articles selected from all articles listed under the respective category in Wikipedia, or listed under sub-categories of the respective category. For example, the target class *Automobiles* includes the Wikipedia articles titled *Chevrolet Tahoe*, *Jaguar XJ220* etc., as illustrated in Table 3. Due to noise within Wikipedia categories, spurious instances such as *Vibrating alert* (for *Mobile phones*) or *Veljko Rus* (for *Universities and colleges*) are occasionally present in the target classes, which makes the evaluation set more realistic than artificially clean. The resulting set of 40 target classes contains an average of 14,974 instances per class.

Extraction Parameters: A tagger links query fragments to their disambiguated, corresponding Wikipedia instances (i.e., to Wikipedia articles). The tagger is simplified to select the longest instance mentions in case of multiple, overlapping possible mentions. Depending on the sources of textual data available for training, any taggers (Cucerzan, 2007; Ratinov et al., 2011; Pantel et al., 2012) that disambiguate text fragments relative to Wikipedia entries can be employed.

The application of extraction patterns to disambiguated queries identifies matching source queries, which contain a Wikipedia instance and a candidate attribute. There are almost 1 million

Class: Examples of Instances
Actors: Bobby Todd, Chunky Pandey, Cody Estes, Emma Cleasby, Koen Crucke, Lauro Delgado, Leandra Leal, Leo Hallerstam, Lindsay Sloane, Margit Saad, Renu Saikia
Animated characters: Boris Badenov, Flying Dutchman (SpongeBob SquarePants), Kang and Kodos, Little Red-Haired Girl, Road Runner (video game), Strafe (Transformers), Subaru Sumeragi, Teletraan I, Zor (Robotech)
Astronomical objects: 65 Andromedae, Beta Lacertae, Dione (moon), Eta Corvi, Jewel Box (star cluster), Lacaille 8760, Messier 90, Mu Pegasi, Radio relics, S/2011 J 2
Automobiles: Chevrolet Tahoe, Ford CD2 platform, Jaguar XJ220, Lancia Dikappa, Mercedes-Benz Vario, Mitsubishi Mizushima, Rolls-Royce Phantom VI, Simca Esplanada, Tata Pixel
Awards: Academy Award, Dan Immerfall, Kalyna Roberge, Pulitzer Prizes, Justin Winsor Prize (library), Palme d'Or, Phajol Moolsan, Wallace Roney
Chemical elements: Aluminium, Bromine, Californium, Group 12 element, Group 7 element, Ilmenium, Lanthanum, Nebulium, Period 3 element, Polonium, Unbiunium, Yttrium, Zirconium
Companies: Consilient, Dorfan, Enercell, Fixafone, Gigaset Communications, Ingman, N3V Games, Quakers Yard and Merthyr Railway, Southeastern Airlines, TV Senado
Digital cameras: Fujifilm FinePix T-series, Fujifilm FinePix X100, General Imaging, Kodak DCS 300 series, Nikon Coolpix series, Nikon D60, Olympus E-400, Pentax K10D, Polaroid Z340
Diseases and disorders: Anthropophilia in animals, Bladder spasm, Hyperlysinemia, Inappropriate sinus tachycardia, Lymphoid Leucosis, Pantothenate kinase-associated neurodegeneration, Repetitive strain injury, Xwrits
Films: I Am Trying to Break Your Heart: A Film About Wilco, It Takes Two (1995 film), Mockingbird Don't Sing, Moster fra Mols, Prayers for Bobby, Roped, Strings (2004 film), Ten Thousand Years Older
Flowers: Acanthephippium mantinianum, Alcea, Dahlia 'Moonfire', Evergreen rose, Flower frog, Geranium caespitosum, Gilliflower, Hyacinth (plant), Iris douglasiana, Lavandula stoechas
Hurricanes in North America: 1856 Last Island hurricane, 1948 Miami hurricane, Effects of Hurricane Isabel in New York and New England, Hurricane Bertha (2008), Hurricane Charley, Hurricane Jova (2011)
Mobile phones: HTC Desire, LG Rumor, LG VX9400, Motorola A910, Nokia 1100, Nokia 7230, Nokia N71, Samsung S5560, Samsung SGH-T809, Samsung SPH-A460, Serene (phone), Vibrating alert
Mountains: Boulder Hills, Cerro San Luis Obispo, Lumiere Peak, Mount Erymanthos, Mount Shindainichi, Mynydd Mawr, Peters Mountain, Poroto Mountains, Remsspitze, Steel Peak, Stob Coire an Laoigh, Stoodley Pike
National Basketball Association teams: Chicago Bulls, Cleveland Cavaliers, Dallas Mavericks, Golden State Warriors, Houston Rockets, Indiana Pacers, Miami Heat, Philadelphia 76ers
National parks: Bu Gia Map National Park, Defileul Jiului National Park, Dooragan National Park, Great Basalt Wall National Park, Orang National Park, Tortuguero National Park
Newspapers: 7days, Chicago Sun-Times, Chojoongdong, Church Times, Corriere del Trentino, Government Gazette (Greece), L'Ordine Nuovo, Le Propagateur Catholique, Novato Advance, Seattle Medium, Venad Pathrika
Painters: Chester Harding (painter), Domingo Antonio Velasco, Gifford Beal, Giovanni Ambrogio de Predis, Lech Rzewuski, Ronnie Landfield, Tarcisio Merati, Tyler Vlahovich, Wolfgang Bauer (artist)
Programming languages: ALGOL 68, Brutus2D, C11 (C standard revision), CORC, Ease (programming language), GiNaC, IBM Basic assembly language, KANT (software), NESL, Obliq, RoboLogix, Rubinius
Religious faiths traditions and movements: Astral body, Chen Tao ("True Way"), Fudoki, Gnostic Mass, Ife, Lutheran Church in Singapore, Omnism, Pneumatic (Gnosticism)
Rivers: Arctic Red River, Deadwood River, Kuzumaru Dam, Ounce River, Pingo River, Presumpscot River, Reno (river), Sand Creek (Denver, Colorado), Viehmoorgraben, Zebracu River
Skyscrapers: 100 Montgomery Street, 15 Penn Plaza, 19 South LaSalle Street, 595 Market Street, EQT Plaza, Edelweiss (skyscraper), Transamerica Tower (Baltimore)
Wine: Acqui Terme, Cinque Terre, Costers del Segre, Elqui River, Los Palacios (Vino de la Tierra), Norte de Granada, Santorini, Sovana DOC, Vinho Verde, Wachau wine

Table 3: Examples of instances for a sample of target classes, where an instance is represented by the title of its Wikipedia article

such matching queries. In turn, Wikipedia instances are separately associated with the classes (Wikipedia categories) to which they belong, including the target classes described earlier. Transitively, attributes extracted from queries are thus associated to target classes. If an attribute co-occurs in the source queries with fewer than five, distinct instances of a class, then the attribute is deemed unreliable and therefore removed from the set of candidate attributes of the class.

Label	Value	Examples of Attributes
vital	1.0	Painters: famous when he was alive?
		Programming languages: compiled?
		Stadiums: covered?
okay	0.5	Painters: born in italy?
		Programming languages: useful?
		Stadiums: safe?
wrong	0.0	Painters: stolen?
		Programming languages: affectionate pets?
		Stadiums: good college?

Table 4: Correctness labels manually assigned to attributes extracted for various classes

Class	Precision of Extracted Attributes			
	%vital	%okay	%wrong	Score
Animated characters	64	32	4	0.80
Companies	84	0	16	0.84
Empires	68	20	12	0.78
Hurricanes in North America	52	44	4	0.74
Organizations designated as terrorist	56	8	36	0.60
Programming languages	96	4	0	0.98
Wine	44	4	52	0.46
...				
Avg-All-Classes	78	12	10	0.84

Table 5: Accuracy of attributes extracted from conjectural queries, over the set of 40 target classes

4 Evaluation Results

Attribute Accuracy: A sample of 25 attributes from each target class is manually assigned correctness labels. Following previously introduced methodology, an attribute is marked as *vital*, if it must be present amongrepresentative attributes of the class; *okay*, if it provides useful but non-essential information; and *wrong*, if it is incorrect (Paşca, 2007). When attributes are not *vital*, the choice between the labels *okay* vs. *wrong* often depends on whether the instances mentioned in source queries have been disambiguated to the correct entries in Wikipedia or not. To compute the precision score over a set of attributes, the correctness labels are converted to numeric values as shown in Table 4. Precision is the sum of the correctness values of the attributes, divided by the number of attributes.

Table 5 summarizes the resulting precision scores over the evaluation set of target classes. The scores vary from one class to another, for example 0.46 for *Wine* but 0.98 for *Programming languages*. The average score is 0.84, indicating that attributes extracted from conjectural queries have encouraging levels of accuracy. Table 6 shows examples of attributes extracted for some of the target classes, whereas Table 7 contains examples of source queries from which various attributes are extracted. Attributes extracted from *why*-prefixed source queries tend to be individually more reliable that attributes from *is*-prefixed queries. As noted earlier, this is because *why*-prefixed queries tend to request an explanation for something already known to be true to the questioner, instead of merely asking whether something is true or not.

Error Analysis: Erroneous attributes are extracted mainly due to two reasons. First, the presence of noisy instances in an input target class, illustrated earlier in Table 3, causes the extraction of attributes that may be relevant to individual instances but not to the class. For example, a significant number of instances in the target classes *Awards* and *Wine* are incorrect, because Wikipedia lists the categories *Award winners* and *Wine regions* as sub-categories of *Awards* and *Wine* respectively.

Class: Extracted Attributes
Association football clubs: playing today?, in debt?, a big club?, in the champions league?, in europe?, in fifa 12?, wearing black armbands?, for sale?, going to sign anyone?,..
Automobiles: a good car?, reliable?, front wheel drive?, a good first car?, rear wheel drive?, all wheel drive?, safe?, a girl car?, fast?, 4 wheel drive?, awd?, a sports car?, discontinued?, worth it?, a chick car?, good in snow?, good on gas?, parts expensive?, expensive to maintain?, discontinued?, being discontinued?,..
Awards: married?, famous?, a vegetarian?, overrated?, broke?, cancelled?, left handed?, based on a true story?, playing tonight?, having a baby?, appropriate for kids?, a good show?,..
Battles and operations of World War II: a success?, fought?, necessary?, a turning point?, an important battle?, important to the allies?, a failure?, in world war 2?, successful?, inevitable?, called that?, important to the us?, planned?,..
Chemical elements: dangerous?, flammable?, a metal nonmetal or metalloid?, a solid?, magnetic?, reactive?, toxic?, malleable?, a metalloid?, poisonous?, man made?, rare?, harmful?, a transition metal?, explosive?, an element?, found in nature?, conductive?, ductile?, paramagnetic?, a gas?, safe?, a solid liquid or gas at room temperature?, expensive?, hazardous?, soluble?, stable?, a conductor?, a mineral?, a cation or anion?, a compound?, combustible?, soluble in water?, brittle?, diamagnetic?, corrosive?, diatomic?,..
Companies: a good company to work for?, a good place to work?, free?, worth it?, a scam?, a good brand?, a public company?, legit?, expensive?, reliable?, publicly traded?, open on thanksgiving?, hiring?, down?, expensive?, a franchise?, a fortune 500 company?, going out of business?, open on labor day?, open on easter?, open today?, legitimate?, a good investment?, a word?, dead?, real?, open on sunday?, for sale?, slow?, an american company?, in trouble?,..
Currencies by country: strong?, pegged?, pegged to us dollar?, backed by gold?, overvalued?, going down?, rising?, strong?, strengthening?, a good investment?, appreciating?, depreciating?, a convertible currency?, a proper noun?, capitalized?, convertible?,..
Digital cameras: a good camera?, full frame?, discontinued?, worth it?, worth it?, a professional camera?, waterproof?, a professional camera?, fx or dx?, sdhc compatible?, worth the money?, out of stock?, full frame sensor?, made in japan?, weather sealed?, worth buying?, a good camera for beginners?, being replaced?, easy to use?, weather sealed?, good for video?, a full frame dslr?,..
Diseases and disorders: hereditary?, contagious?, curable?, genetic?, dangerous?, fatal?, serious?, painful?, common?, deadly?, a disability?, a disease?, treatable?, life threatening?, permanent?, inherited?, infectious?, reversible?, rare?, preventable?, a sign of pregnancy?, an autoimmune disease?, dominant or recessive?, communicable?, chronic?, an std?, a mental illness?,..
Films: on netflix?, a true story?, rated r?, a good movie?, scary?, on dvd?, appropriate for kids?, rated pg-13?, funny?, on demand?, for kids?, a remake?, still in theaters?, a disney movie?, in english?, out?, sad?, worth watching?, accurate?, in theaters?, gory?,..
Food ingredients: gluten free?, safe?, bad for you?, good for you?, healthy?, vegan?, safe during pregnancy?, dangerous?, harmful?, toxic?, halal?, fattening?, bad for dogs?, poisonous?, natural?, vegetarian?, acidic?, edible?, good for health?, kosher?, kosher for passover?, unhealthy?, soluble in water?, paleo?, organic?, polar?, flammable?, alkaline?, safe to eat?, good for skin?, a carbohydrate?, safe for pregnant women?, addictive?, good for diabetics?, good for hair?, safe for babies?, a compound?,..
Internet search engines: safe?, down?, free?, legit?, important?, legal?, better than google?, not working?, a meta search engine?, effective?, reliable?, accurate?, case sensitive?, profitable?, account free?, blocked?, a buy?,..
Mobile phones: a good phone?, 3g?, a smartphone?, triband?, available in india?, android?, worth it?, worth buying?, gsm?, better than iphone?, symbian?, touch screen?, compatible with mac?, a smart phone?, 4g?, worth buying?, wifi?, 4g?, discontinued?, quad band?, coming to verizon?, a world phone?, better than iphone 4?, free?, dual sim?, unlocked?,..
National parks: established?, safe?, a world heritage site?, famous?, unique?, expensive?, busy in may?, good for nightlife?, hot in september?, in danger?, in europe?, nice?, open in august?,..
Newspapers: conservative?, liberal or conservative?, reliable?, biased?, right wing?, a reliable source?, a tabloid?, left wing?, a broadsheet?, credible?, a daily newspaper?, a magazine?, a national newspaper?, a scholarly source?, free?, real?, important?, app free?, fake?, reputable?, a peer reviewed journal?, a reputable source?, a scholarly journal?,..
Programming languages: case sensitive?, object oriented?, easy to learn?, open source?, still used?, a scripting language?, compiled?, easy?, compiled or interpreted?, installed?, fast?, object oriented language?, oop?, platform independent?, turing complete?, compatible with windows 7?, safe?, popular?, slow?, developed?, a good language?, obsolete?, portable?, secure?, thread safe?,..
Rivers: polluted?, safe?, clean?, safe to swim in?, famous?, brown?, flooding?, tidal?, man made?, navigable?, sustainable?, a good place to live?, worth visiting?, drying up?,..
Stadiums: covered?, a dome?, air conditioned?, indoors?, heated?, roof open?, a retractable roof?, safe?, grass?, haunted?, demolished?, complete?, fixed?, open to the public?, open today?, still standing?, turf?, worth it?,..
Treaties: signed?, legally binding?, fair?, controversial?, a failure?, successful?, necessary?, needed?, written?, significant?, a law?, significant?, effective?, introduced?, written?, working?,..
Universities and colleges: a party school?, a public or private university?, aacsb accredited?, a good school for nursing?, fully accredited?, a good university?, jesuit?, a good school for business?, test optional?, a four year college?, a graduate school?, a reputable school?, a safe campus?, a state college?, mba accredited?, a top school?, considered ivy league?, in a bad neighborhood?, accredited in canada?, going to a bowl game 2011?, quarter or semester?, single choice early action?,..
Video games: multiplayer?, free?, worth buying?, fun?, 2 player?, online?, hard?, out?, for mac?, rated m?, on ps3?, on pc?, safe?, split screen?, compatible with windows 7?, on xbox 360?, scary?, free roam?, worth playing?, region free?, co op?, on wii?, compatible with vista?, expensive?, open world?, not working?, real?, on psp?, laggy?,..
Wine: a grape?, expensive?, safe?, worth visiting?, famous?, important?, an island?, hot in october?, sweet?, a dry wine?, a region?, in the eu?, in tuscany?, nice?, red or white wine?, italian?, a champagne?, a city or country?, a dessert wine?,..

Table 6: Examples of attributes extracted from queries

Consequently, *Awards* and *Wine* respectively contain many people names and geographical regions as their instances. Noisy instances in target classes could be removed or reduced, by filtering the

Source Query Containing a Mention of an Instance and an *Attribute*	Instance	Class
are e36 m3 *reliable*	BMW M3	Automobiles
is the chevy avalanche *reliable*	Chevrolet Avalanche	Automobiles
why is the honda civic *reliable*	Honda Civic	Automobiles
is chrysler neon *a good car*	Chrysler Neon	Automobiles
is honda element *a good car*	Honda Element	Automobiles
are honda accords *rear wheel drive*	Honda Accord	Automobiles
is a mini cooper s *rear wheel drive*	Mini	Automobiles
is dodge charger *rear wheel drive*	Dodge Charger (LX)	Automobiles
is the jaguar xf *rear wheel drive*	Jaguar XF	Automobiles
are ford mustangs *expensive to maintain*	Ford Mustang	Automobiles
are bmw 3 series *expensive to maintain*	BMW 3 Series	Automobiles
is range rover *expensive to maintain*	Range Rover	Automobiles
are boxsters *expensive to maintain*	Porsche Boxster	Automobiles
is a range rover *expensive to maintain*	Range Rover	Automobiles
are 350z *expensive to maintain*	Nissan 350Z	Automobiles
why was the battle of el alamein *important to the allies*	Second Battle of El Alamein	Battles and operations of World War II
why was the north africa campaign *important to the allies*	North African Campaign	Battles and operations of World War II
is europium *dangerous*	Europium	Chemical elements
why is radon *dangerous*	Radon	Chemical elements
is scandium *a solid liquid or gas at room temperature*	Scandium	Chemical elements
is xenon *a solid liquid or gas at room temperature*	Xenon	Chemical elements
is iron *corrosive*	Iron	Chemical elements
is hydrogen gas *corrosive*	Hydrogen	Chemical elements
is chlorine *corrosive*	Chlorine	Chemical elements
why is chlorine *corrosive*	Chlorine	Chemical elements
why is oxygen *soluble in water*	Oxygen	Chemical elements
why is iodine *soluble in water*	Iodine	Chemical elements
why are the green bay packers *publicly traded*	Green Bay Packers	Companies
is quicken loans *publicly traded*	Quicken Loans	Companies
why is starbucks *a good company to work for*	Starbucks	Companies
why is southwest airlines *a good company to work for*	Southwest Airlines	Companies
is henkel *a good company to work for*	Henkel	Companies
is sodexo *a good company to work for*	Sodexo	Companies
is porter cable *a good brand*	Porter-Cable	Companies
is chevrolet *a good brand*	Chevrolet	Companies
why is coca cola *a good brand*	Coca-Cola	Companies
why is nike *a good brand*	Nike, Inc.	Companies
is singapore dollar *backed by gold*	Singapore dollar	Currencies by country
is the philippine peso *backed by gold*	Philippine peso	Currencies by country
is iraq dinar *a convertible currency*	Iraqi dinar	Currencies by country
why is the malaysian ringgit *pegged to us dollars*	Malaysian ringgit	Currencies by country
why is hong kong dollar *pegged to us dollar*	Hong Kong dollar	Currencies by country
is singapore dollar *pegged to us dollar*	Singapore dollar	Currencies by country
is indian rupee *pegged to us dollar*	Indian rupee	Currencies by country
is the d90 *weather sealed*	Nikon D90	Digital cameras
is d700 *weather sealed*	Nikon D700	Digital cameras
is nikon d90 *weather sealed*	Nikon D90	Digital cameras
is nikon d300 *full frame sensor*	Nikon D300	Digital cameras
why is gothika *rated r*	Gothika	Films
why was the book of eli *rated r*	The Book of Eli	Films
why was the matrix *rated r*	The Matrix	Films

Table 7: Examples of source queries from which candidate attributes are extracted for various target classes

category-to-category edges from Wikipedia into a category taxonomy (Ponzetto and Strube, 2007; Ponzetto and Navigli, 2009) prior to assembling the target classes from categories. Second, the incorrect disambiguation of instances in queries causes attributes of a different instance with a similar name to be associated with the wrong class.

Conjectural Attribute → Possible Mappings to Equivalent Nominal Attributes	
Astronomical objects: bigger than earth? → size relative to earth^{A}, size^{P}, volume^{W,F,P}	Astronomical objects: capable of supporting life? → ability to support life^{A}
Astronomical objects: circumpolar? → ∅	Food ingredients: fattening? → energy^{W,F}, calories^{W,F}
Chemical elements: man made? → manufacturing process^{P}	Chemical elements: toxic? → toxicity^{P}
Chemical elements: soluble in water? → solubility in water^{W,F,P}	Chemical elements: a solid liquid or gas at room temperature? → phase^{W,F,P}
Chemical elements: conductive? → conductivity^{P}, electrical conductivity^{A}, electrical resistivity^{W,F}	Automobiles: reliable? → reliability^{P}
Automobiles: front wheel drive? → driveline^{F}	Automobiles: safe? → safety rating^{P}
Mobile phones: smartphone? → function as smartphone^{A}	Mobile phones: compatible with mac? → compatibility with mac^{P}
Cities: dog friendly? → dog friendliness^{A}	Cities: nice place to live? → attractiveness as a place to live^{A}
Cities: cheap? → cost of living^{P}	Companies: publicly traded? → stock symbol^{P}
Companies: fortune 500 company? → ∅	Companies: for sale? → availability for sale^{A}
Companies: a good place to work? → ranking^{P}, employee benefits^{P}	Films: on netflix? → availability on netflix^{A}, netflix title^{F}, providers^{P}
Films: sad? → genre^{F,P}	National parks: busy in may? → number of visitors in may^{A}, may visitors statistics^{A}, visitor statistics^{P}
National Basketball Association teams: losing money? → cash flow^{A}, financial statement^{P}, debt,	Newspapers: conservative? → preference for conservative views^{A}, political alignment^{W}, political bias^{P}
Newspapers: reputable? ? → reputation^{A}	Programming languages: case sensitive? → case sensitivity^{A}
Programming languages: obsolete? → ∅	Treaties: binding? → ∅

Table 8: Examples of mappings from conjectural attributes to their equivalent nominal attributes, if any. Nominal attributes present in Wikipedia, Freebase or among non-conjectural attributes extracted with previous methods (Paşca et al., 2007), are marked as W, F and/or P respectively; otherwise, they are marked A. A lack of equivalent nominal attributes is marked as ∅

Relation to Nominal Attributes: Table 8 reviews a sample of conjectural attributes, from the point of view of the availability of equivalent noun-phrase attributes into which conjectural attributes can be mapped. The nominal attributes would correspond to attributes available in human-compiled resources or that may be extracted by previous methods. Out of a sample of 200 conjectural attributes, 72% are manually found to have possible mappings into equivalent nominal attributes, of which 28% fully preserve the meaning (e.g., *safe?→safety rating*) but 44% only partially approximate the meaning (e.g., *bigger than earth?→volume*). When possible mappings to nominal attributes exist, whether fully or partially meaning-preserving, 18% (Wikipedia), 22% (Freebase) and 33% are present among the attributes available for instances of the respective classes in Wikipedia, Freebase or among the top 500 attributes returned by the method from (Paşca, 2007). About 31% of the conjectural attributes in the sample are found to not have a nominal equivalent, and an additional 28% have a possible nominal equivalent that is not present in any of the respective resources. Overall, the analysis suggests that conjectural attributes are not already captured by, and therefore complement, existing or previously extracted attributes.

Relation to Unary Attributes: To our knowledge, (Van Durme et al., 2008) is the only previous attribute extraction method allowing for the extraction of non-nominal attributes. It parses document sentences, leading to so-called unary attributes of classes, such as *trapped*, *dangerous*, and *unfortunate* for *Animals*. The attributes are collected from document sentences, roughly by identifying adjectival modifiers of mentions of the class (“*..the animals were trapped..*”, “*..the trapped*

Class	Extracted Attributes
Painter	D_U: famous, romantic, distinguished, celebrated, well-known, pre-raphaelite, flemish, dutch, abstract
	Q_C: famous when he was alive?, an important artist?, dead?, born in italy?, rich or poor?, left handed?
Animal	D_U: dead, trapped, dangerous, unfortunate, intact, hungry, wounded, tropical, sick, favourite
	Q_C: good with children?, aggressive?, smart?, dangerous?, hard to train?, endangered?, nocturnal?
Drug	D_U: dangerous, powerful, addictive, safe, illegal, experimental, effective, prescribed, harmful, hallucinatory
	Q_C: safe during pregnancy?, addictive?, dangerous?, over the counter?, a controlled substance?, legal?, effective?
Apple	D_U: red, juicy, fresh, bad, substantive, stuffed, shiny, ripe, green, baked
	Q_C: good for baking?, tart?, cooking apple?, sweet?, healthy?, good for canning?, biennial?, gmo?
Earthquake	D_U: disastrous, violent, underwater, prolonged, powerful, popular, monstrous, fatal, famous, epic
	Q_C: predicted?, destructive?, man made?, deadly?, devastating?, common?, normal?

Table 9: Comparison between unary attributes (D_U) extracted from documents as described in (Van Durme et al., 2008), and conjectural attributes (Q_C) extracted from queries as described in the current paper. To extract comparable attributes, the classes from (Van Durme et al., 2008), shown in the first column, are manually mapped into their equivalent Wikipedia categories *Painters*, *Animals*, *Drugs*, *Apple cultivars* and *Earthquakes* respectively

animals..”). In comparison to (Van Durme et al., 2008), our method takes noisy queries as input, rather than clean document sentences from reliable documents such as news articles. The comparison in Table 9, between attributes extracted for a sample of classes, highlights a few additional differences between the two methods. First, the attributes extracted in (Van Durme et al., 2008) seem to be limited to single-word adjectives, whereas conjectural attributes accommodate other phrases too. Second, the different intuitions behind the two methods determine what kind of attributes one would expect to see extracted. Unary attributes from (Van Durme et al., 2008) often capture transient states or events in which the respective classes are involved, like *Animals* being *trapped*, or *Apples* being *ripe* or *baked*. Such states or events may be relevant in the particular context of the document containing the source sentences. But they are not necessarily generally-relevant, context-independent properties that Web users would likely inquire about. Indeed, adding the property *trapped* for the class *Animals* to Wikipedia, and filling in its corresponding values for all instances (kinds) of *Animals*, would likely have little value. In contrast, conjectural attributes like *aggressive?* or *nocturnal?* are information-seeking, and often refer to permanent, context-independent rather than transient, context-dependent properties of the instances of the class *Animals*.

5 Related Work

A variety of methods address the more general task of acquisition of open-domain relations from text, e.g., (Zhu et al., 2009; Carlson et al., 2010; Fader et al., 2011; Lao et al., 2011). In order to acquire class attributes in particular, a common strategy is to first acquire attributes of instances, then aggregate or propagate (Talukdar and Pereira, 2010) attributes, from instances to the classes to which the instances belong. The identification of relevant instances within queries is related to the task of word sense disambiguation (Ponzetto and Navigli, 2010).

Data available within Web documents, from which attributes are extracted in previous work, includes unstructured (Tokunaga et al., 2005; Paşca et al., 2007), structured (Raju et al., 2008) and semi-structured text (Yoshinaga and Torisawa, 2007), layout formatting tags (Wong et al., 2008), itemized lists or tables (Cafarella et al., 2008). Another source of attributes is data in human-compiled encyclopedia (Wu et al., 2008; Cui et al., 2009), including infoboxes and category labels (Suchanek et al., 2007; Nastase and Strube, 2008; Wu and Weld, 2008) associated with

Wikipedia articles (Remy, 2002). The role of Web search queries, as an alternative textual data source to Web documents in open-domain information extraction, has been investigated in the tasks of attribute extraction (Paşca, 2007), as well as in collecting labeled (Sekine and Suzuki, 2007; Pennacchiotti and Pantel, 2009) or unlabeled sets of related instances (Jain and Pennacchiotti, 2010) and ranking of class labels already extracted from text (Billerbeck et al., 2010).

6 Conclusion

Collectively, queries that inquire whether an attribute applies to individual instances or not are evidence that the attribute in question does apply to classes to which the instances belong. The resulting conjectural attributes have encouraging accuracy, and complement attributes available in manually-compiled resources or automatically extracted with previous methods. Current work investigates the role of repositories of class labels (*heated stadiums*) extracted from text for various instances (*Millennium Stadium*), as an additional source of evidence towards extracting class attributes (*heated?*); and the acquisition of (mostly binary) values of conjectural attributes for individual instances.

References

Billerbeck, B., Demartini, G., Firan, C., Iofciu, T., and Krestel, R. (2010). Ranking entities using Web search query logs. In *Proceedings of the 14th European Conference on Research and Advanced Technology for Digital Libraries (ECDL-10)*, pages 273–281, Glasgow, Scotland.

Bollacker, K., Evans, C., Paritosh, P., Sturge, T., and Taylor, J. (2008). Freebase: A collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 International Conference on Management of Data (SIGMOD-08)*, pages 1247–1250, Vancouver, Canada.

Cafarella, M., Halevy, A., Wang, D., Wu, E., and Zhang, Y. (2008). WebTables: Exploring the power of tables on the Web. In *Proceedings of the 34th Conference on Very Large Data Bases (VLDB-08)*, pages 538–549, Auckland, New Zealand.

Carlson, A., Betteridge, J., Wang, R., Hruschka, E., and Mitchell, T. (2010). Coupled semi-supervised learning for information extraction. In *Proceedings of the 3rd ACM Conference on Web Search and Data Mining (WSDM-10)*, pages 101–110, New York.

Cucerzan, S. (2007). Large-scale named entity disambiguation based on Wikipedia data. In *Proceedings of the 2007 Conference on Empirical Methods in Natural Language Processing (EMNLP-07)*, pages 708–716, Prague, Czech Republic.

Cui, G., Lu, Q., Li, W., and Chen, Y. (2009). Automatic acquisition of attributes for ontology construction. In *Proceedings of the 22nd International Conference on Computer Processing of Oriental Languages*, pages 248–259, Hong Kong.

Fader, A., Soderland, S., and Etzioni, O. (2011). Identifying relations for open information extraction. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing (EMNLP-11)*, pages 1535–1545, Edinburgh, Scotland.

Jain, A. and Pennacchiotti, M. (2010). Open entity extraction from Web search query logs. In *Proceedings of the 23rd International Conference on Computational Linguistics (COLING-10)*, pages 510–518, Beijing, China.

Lao, N., Mitchell, T., and Cohen, W. (2011). Random walk inference and learning in a large scale knowledge base. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing (EMNLP-11)*, pages 529–539, Edinburgh, Scotland.

Nastase, V. and Strube, M. (2008). Decoding Wikipedia categories for knowledge acquisition. In *Proceedings of the 23rd National Conference on Artificial Intelligence (AAAI-08)*, pages 1219–1224, Chicago, Illinois.

Paşca, M. (2007). Organizing and searching the World Wide Web of facts - step two: Harnessing the wisdom of the crowds. In *Proceedings of the 16th World Wide Web Conference (WWW-07)*, pages 101–110, Banff, Canada.

Paşca, M. and Van Durme, B. (2007). What you seek is what you get: Extraction of class attributes from query logs. In *Proceedings of the 20th International Joint Conference on Artificial Intelligence (IJCAI-07)*, pages 2832–2837, Hyderabad, India.

Paşca, M., Van Durme, B., and Garera, N. (2007). The role of documents vs. queries in extracting class attributes from text. In *Proceedings of the 16th International Conference on Information and Knowledge Management (CIKM-07)*, pages 485–494, Lisbon, Portugal.

Pantel, P., Lin, T., and Gamon, M. (2012). Mining entity types from query logs via user intent modeling. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (ACL-12)*, pages 563–571, Jeju Island, Korea.

Pennacchiotti, M. and Pantel, P. (2009). Entity extraction via ensemble semantics. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing (EMNLP-09)*, pages 238–247, Singapore.

Ponzetto, S. and Navigli, R. (2009). Large-scale taxonomy mapping for restructuring and integrating Wikipedia. In *Proceedings of the 21st International Joint Conference on Artificial Intelligence (IJCAI-09)*, pages 2083–2088, Pasadena, California.

Ponzetto, S. and Navigli, R. (2010). Knowledge-rich word sense disambiguation rivaling supervised systems. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics (ACL-10)*, pages 1522–1531, Uppsala, Sweden.

Ponzetto, S. and Strube, M. (2007). Deriving a large scale taxonomy from Wikipedia. In *Proceedings of the 22nd National Conference on Artificial Intelligence (AAAI-07)*, pages 1440–1447, Vancouver, British Columbia.

Raju, S., Pingali, P., and Varma, V. (2008). An unsupervised approach to product attribute extraction. In *Proceedings of the 31st International Conference on Research and Development in Information Retrieval (SIGIR-08)*, pages 35–42, Singapore.

Ratinov, L., Roth, D., Downey, D., and Anderson, M. (2011). Local and global algorithms for disambiguation to Wikipedia. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics (ACL-11)*, pages 1375–1384, Portland, Oregon.

Remy, M. (2002). Wikipedia: The free encyclopedia. *Online Information Review*, 26(6):434.

Sekine, S. and Suzuki, H. (2007). Acquiring ontological knowledge from query logs. In *Proceedings of the 16th World Wide Web Conference (WWW-07), Posters*, pages 1223–1224, Banff, Canada.

Suchanek, F., Kasneci, G., and Weikum, G. (2007). Yago: a core of semantic knowledge unifying WordNet and Wikipedia. In *Proceedings of the 16th World Wide Web Conference (WWW-07)*, pages 697–706, Banff, Canada.

Talukdar, P. and Pereira, F. (2010). Experiments in graph-based semi-supervised learning methods for class-instance acquisition. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics (ACL-10)*, pages 1473–1481, Uppsala, Sweden.

Tokunaga, K., Kazama, J., and Torisawa, K. (2005). Automatic discovery of attribute words from Web documents. In *Proceedings of the 2nd International Joint Conference on Natural Language Processing (IJCNLP-05)*, pages 106–118, Jeju Island, Korea.

Van Durme, B., Qian, T., and Schubert, L. (2008). Class-driven attribute extraction. In *Proceedings of the 22nd International Conference on Computational Linguistics (COLING-08)*, pages 921–928, Manchester, United Kingdom.

Wong, T., Lam, W., and Wong, T. (2008). An unsupervised framework for extracting and normalizing product attributes from multiple Web sites. In *Proceedings of the 31st International Conference on Research and Development in Information Retrieval (SIGIR-08)*, pages 35–42, Singapore.

Wu, F., Hoffmann, R., and Weld, D. (2008). Information extraction from Wikipedia: Moving down the long tail. In *Proceedings of the 14th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD-08)*, pages 731–739.

Wu, F. and Weld, D. (2008). Automatically refining the Wikipedia infobox ontology. In *Proceedings of the 17th World Wide Web Conference (WWW-08)*, pages 635–644, Beijing, China.

Yoshinaga, N. and Torisawa, K. (2007). Open-domain attribute-value acquisition from semi-structured texts. In *Proceedings of the 6th International Semantic Web Conference (ISWC-07), Workshop on Text to Knowledge: The Lexicon/Ontology Interface (OntoLex-2007)*, pages 55–66, Busan, South Korea.

Zhu, J., Nie, Z., Liu, X., Zhang, B., and Wen, J. (2009). StatSnowball: a statistical approach to extracting entity relationships. In *Proceedings of the 18th World Wide Web Conference (WWW-09)*, pages 101–110, Madrid, Spain.

A Comprehensive Analysis of Constituent Coordination for Grammar Engineering

Agnieszka PATEJUK Adam PRZEPIÓRKOWSKI
Institute of Computer Science, Polish Academy of Sciences,
ul. Jana Kazimierza 5, 01-248 Warszawa, Poland
{aep,adamp}@ipipan.waw.pl

ABSTRACT
This paper provides an explicit, formal analysis of two not only interesting but also frequent coordination phenomena: the coordination of unlike categories and the coordination of distinct grammatical functions possibly belonging to entirely different levels of structure. The proposed account of the former makes it possible to take full advantage of new generation valence dictionaries which encode information about the possibility of such non-standard coordination, creating new possibilities for existing grammar implementations. Furthermore, interactions with complex phenomena such as case assignment are taken into consideration, providing a solution which may be adopted for modelling similar phenomena in other languages. The other variety of coordination addressed in this paper sheds some new light on coordination and how it should be modelled. It demonstrates on the basis of attested data that coordination is subject to fewer constraints than previously assumed and provides an elegant linguistically-motivated solution employing currently available LFG mechanisms.

TITLE AND ABSTRACT IN POLISH

Kompleksowa analiza koordynacji składnikowej na potrzeby inżynierii lingwistycznej

Niniejszy artykuł przedstawia sformalizowaną analizę dwóch zarówno interesujących, jak i częstych zjawisk związanych z koordynacją: koordynację różnych kategorii oraz koordynację różnych funkcji gramatycznych, które mogą również należeć do zupełnie odmiennych poziomów struktury. Zaproponowana analiza pozwala na wykorzystanie w pełni możliwości stwarzanych przez słowniki walencyjne nowej generacji, które zawierają informacje o możliwości wystąpienia niejednorodnej koordynacji, co otwiera zupełnie nowe perspektywy przed istniejącymi już implementacjami gramatyk. Ponadto brane są pod uwagę interakcje ze złożonymi zjawiskami takimi jak nadawanie przypadka, które to rozwiązanie może zostać przystosowane do opisu podobnych zjawisk w innych językach. Drugi rodzaj koordynacji opisany w artykule rzuca nowe światło na koordynację i jej formalny opis: na podstawie autentycznych danych pokazano, że koordynacja podlega znacznie mniejszej liczbie ograniczeń, niż wcześniej sądzono, oraz zaproponowano eleganckie rozwiązanie o solidnych podstawach lingwistycznych, wykorzystujące obecnie dostępne mechanizmy LFG.

KEYWORDS: coordination, unlikes, formal grammar, parsing, Polish, LFG.

KEYWORDS IN POLISH: koordynacja, gramatyka formalna, parsowanie, polski, LFG.

Proceedings of COLING 2012: Technical Papers, pages 2191–2208,
COLING 2012, Mumbai, December 2012.

1 Introduction

During the last two decades of the previous millennium grammar engineering was solidly placed within the core of Natural Language Processing (NLP), as also witnessed by the number of papers devoted to this topic in COLING proceedings of 1980s and 1990s. While – in the age of inductive NLP – this theme has almost disappeared from major conferences, grammar engineering efforts, supported by new corpus-based methods, have continued, especially within the HPSG[1] and LFG[2] communities, where large-scale multilingual grammar engineering initiatives were set up: DELPH-IN (`http://www.delph-in.net/`) and PARGRAM (`http://pargram.b.uib.no/`). There has also been much progress during the last 15 years or so in the theoretical linguistic foundations of these efforts, with flourishing HPSG and LFG conferences and with analyses set up within these frameworks appearing in prestigious linguistic journals.

One of the topics that has received much attention in formal theoretical linguistics is coordination, a phenomenon which is not only theoretically challenging, but also – due to its textual frequency – crucial to grammar engineering. Unfortunately, as coordination remains difficult to describe accurately and exhaustively, this theoretical interest is not fully reflected in existing grammar implementations.

The aim of this paper is to present a comprehensive implementation of constituent coordination, a part of an ongoing effort to develop a wide-coverage LFG parser of Polish (Patejuk and Przepiórkowski, 2012b). Polish is a good test-bed for the task at hand, as it offers a wide range of interactions between coordination on one hand and various agreement, case assignment and valence phenomena on the other.

One aspect of coordination that has remained especially elusive in grammar engineering is the possibility to coordinate elements which are unlike in some sense. Polish offers a much wider range of unlike constituent coordination than has been discussed in NLP, including the so-called lexico-semantic coordination (Kallas 1993; Chaves and Paperno 2007; Gazdik 2010), i.e., the coordination of *very* unlikes, where coordinated items do not even represent the same grammatical function. While such cases go against any comprehensive analysis of coordination we are aware of, they are textually frequent and therefore should be taken into account in any wide-coverage grammar implementation effort.

Note that, for reasons of space, we do not provide here an analysis of non-constituent coordination, as in *give a teacher an apple and a policeman a flower* (Steedman, 2000, p.46).[3] This subtype of coordination has an elegant analysis in different versions of categorial grammars, while it remains troublesome for many other grammatical theories. Within LFG, a comprehensive analysis of non-constituent coordination is offered in Maxwell and Manning 1996. In fact, while the LFG analysis seems to cover roughly the same patch as categorial grammar analyses, Maxwell and Manning (1996) claim that it is superior,[4] as it provides more natural accounts of cases such as *You may call me directly or after 3pm through my secretary* and *She put a lamp on the table, and on the ledge a large antique punchbowl*, which are difficult because coordinated elements are not only non-constituents, but they also differ in the number and order of constituents. The analysis of the current paper may be extended to cover non-constituent

[1] Head-driven Phrase Structure Grammar; cf. Pollard and Sag 1987, 1994.
[2] Lexical-Functional Grammar, cf. Bresnan 1982 and Dalrymple 2001.
[3] Steedman 2000 calls this phenomenon *argument cluster coordination*.
[4] See Levine 2011, § 2.4, for a refutal of this claim.

coordination along the lines of Maxwell and Manning 1996.

2 Basics: Coordination of 'like' categories

The usual intuition behind implementations of coordination is that, wherever some phrase type XP (e.g., a nominal phrase) may occur, a coordination of XP-like elements (i.e., of nominal phrases) may occur instead. In LFG, this intuition is expressed via constituency rules:

(1) $\begin{array}{lcccc} \text{XP} & \rightarrow & \text{XP} & \text{Conj} & \text{XP} \\ & & \downarrow\in\uparrow & & \downarrow\in\uparrow \end{array}$

LFG makes a distinction between c(onstituent)-structure and f(unctional)-structure (among other linguistic levels). Annotations of elements of c-structure rules, such as $\downarrow\in\uparrow$, are used to construct corresponding f-structures. In this case, the two conjuncts are elements of the set representing the mother. For example, when used to parse the Polish sentence (2), the above rule constructs the set representation of the subject of the sentence, as in (3).[5]

(2) Idą Jan i Marysia.
walk.PL Jan.SG and Marysia.SG
'Jan and Marysia walk.'

(3) $\left\{\left[\text{PRED} \quad \text{'Jan'}\right], \left[\text{PRED} \quad \text{'Marysia'}\right]\right\}$

This simple example also illustrates the immediate weakness of the basic intuition given above: neither the singular *Jan*, nor the singular *Marysia* alone can directly replace the coordination *Jan i Marysia*, as this would lead to number disagreement with the plural verb. Because of such agreement facts, coordinate structures are represented in LFG as hybrid feature structures, which contain sets such as (3), but may also contain their own features (Dalrymple and Kaplan, 2000); a fuller representation of the subject of (2) would be (4).

(4) $\left[\begin{array}{l} \left\{\left[\begin{array}{ll} \text{PRED} & \text{'Jan'} \\ \text{NUM} & \text{SG} \end{array}\right], \left[\begin{array}{ll} \text{PRED} & \text{'Marysia'} \\ \text{NUM} & \text{SG} \end{array}\right]\right\} \\ \\ \text{NUM} \quad \text{PL} \end{array}\right]$

3 Basics: Coordination of 'unlike' categories

An influential paper that demonstrated that the assumption of 'likeness' of coordinated elements is too strong is Sag et al. 1985, set within GPSG.[6] Under their analysis, examples (5)–(7) are grammatical because both conjuncts satisfy the underspecified requirements which hold for the syntactic position they occupy.

(5) That was a rude remark and in very bad taste.

(6) We walked slowly and with great care.

(7) Pat became a republican and quite conservative.

[5] The attribute PRED stands for PREDICATE and represents the basic predicate-argument structure of a given element. Here both predicates 'Jan' and 'Marysia' have no arguments.

[6] Generalized Phrase Structure Grammar; cf. Gazdar et al. 1985.

(8) *Tracy has become a republican and of the opinion that we must place nuclear weapons in Europe.

For example, the verb BE, as in (5), only requires that its complement be predicative, [PRD +], and both conjuncts *a rude remark* and *in very bad taste* are [PRD +] (apart from being a noun phrase and a prepositional phrase, respectively). Similarly, the adjunct in (6) is specified as [MANNER +], and again both conjuncts, although categorially unlike, are [MANNER +]. On the other hand, the verb BECOME puts stronger restrictions on its complement: not only should it be predicative, but also nominal, [N +], in the sense that nouns and adjectives are nominal, while prepositions and verbs are not (they are marked as [N −]). Hence, (7), involving two predicative nominal conjuncts (a noun phrase and an adjectival phrase), is grammatical, while (8), involving a [N −] conjunct (the prepositional *of the opinion...*) is not.

In order for this analysis to work, the theory needs to implement a notion of feature structure subsumption. In GPSG such a notion is hardwired into the Head Feature Convention (cf. Sag et al. 1985, § 2.4). Attempts have also been made to carry over these subsumption-based insights into HPSG, but they require certain extensions of the formal apparatus assumed within HPSG; two such analyses are proposed in Sag 2002 and Yatabe 2004.

Despite the fact that LFG offers interesting analyses of various aspects of coordination (Maxwell and Manning 1996 on non-constituent coordination; Dalrymple and Kaplan 2000 and Dalrymple et al. 2009 on how coordination influences the representation of features such as person, gender and case; Dalrymple and Nikolaeva 2006, Kuhn and Sadler 2007, and Dalrymple and Hristov 2010 on the interaction between coordination and agreement), there is surprisingly little explicit discussion of coordination of unlike constituents. The assumption seems to be that, at the level of c-structure, there are no categorial constraints on conjuncts, i.e., instead of (1) above, the relevant rule would rather resemble (9) below (cf. Peterson 2004, p. 652).

(9) $\text{XP} \rightarrow \underset{\downarrow\in\uparrow}{\text{YP}} \quad \text{Conj} \quad \underset{\downarrow\in\uparrow}{\text{ZP}}$

In LFG, any appearances of categorial 'likeness' are a side effect of constraints imposed at the functional level of representation. For example, in Polish, subjects are typically nominative (but see below), a fact which may be expressed via (10).[7]

(10) $(\uparrow \text{SUBJ CASE}) =_c \text{NOM}$

In fact, on standard LFG assumptions, nothing more is required to ensure that *both conjuncts* in (2) are nominative. This is because LFG distinguishes between distributive features, such as CASE, and non-distributive features, such as NUM(BER) (Dalrymple and Kaplan, 2000). The former distribute over all elements of coordination, so the specification CASE = NOM for the outer hybrid feature structure in (11) results in CASE = NOM on all conjuncts within the set.

(11)
$$\left[\begin{array}{ll} \left\{\left[\begin{array}{ll}\text{PRED} & \text{'JAN'}\\ \text{NUM} & \text{SG}\\ \text{CASE} & \text{NOM}\end{array}\right], \left[\begin{array}{ll}\text{PRED} & \text{'MARYSIA'}\\ \text{NUM} & \text{SG}\\ \text{CASE} & \text{NOM}\end{array}\right]\right\} \\ \text{NUM} \quad \text{PL}\end{array}\right]$$

[7] The constraining equation, expressed by '$=_c$', *verifies* that the equality holds, rather than assigning a value, as in case of the operator '='.

Using these mechanisms, it is simple to implement in LFG an analysis similar to that of Sag et al. 1985. For example, part of the lexical entry of the verb BE might be: (↑ OBJ PRD) = +. Assuming that PRD[8] is a distributive feature, the requirement that the object be predicative will percolate to all conjuncts, without predetermining their categorial status, and thus accounting for (5).

4 Towards an LFG analysis of the coordination of unlikes

At the end of the paper, Sag et al. (1985) discuss cases of coordination of an NP with a clause (p. 165):

(12) Pat remembered the appointment and that it was important to be on time.

(13) That Himmler appointed Heydrich and the implications thereof frightened many observers.

The analysis they propose is far from elegant and it involves, *inter alia*, the assumption that such clauses are in a sense NPs. Apparently, they are attempting to avoid a disjunctive specification of the verb's argument, one that would state that, e.g., the object of the verb REMEMBER is either an NP or a clause (or a coordination of such elements).

When other languages are taken into consideration, it turns out that such disjunctive specifications are inevitable. For example, Kosek 1999, pp. 43–44, cites the following:

(14) Owinął dziecko w koc i ręcznikiem.
wrapped baby in blanket.ACC and towel.INST
'He wrapped the baby in a blanket and in a towel.'

(15) Nadajesz się do tej pracy i na dyrektora.
are fit for this job and for manager
'You are fit for this job and for a manager.'

Prepositional phrases involved in these examples must be headed by the specific prepositions W, DO and NA, and the NP in (14) must be in the instrumental case. It seems highly unlikely that the intuitive disjunctive specifications 'PP[*w*] ∨ NP[*inst*]' (for (14)) and 'PP[*do*] ∨ PP[*na*]' (for (15)) could be replaced by non-disjunctive specifications capturing putative commonalities between, for instance, PP[*w*] and NP[*inst*] (to the exclusion of other types of phrases).

It turns out that such disjunctive specifications are also problematic for LFG. The obvious way to formalise the requirements of the verb OWINĄĆ 'wrap', as in (14), would be:[9]

(16) (↑ OBL CASE) = INST ∨ (↑ OBL PFORM) = W

Unfortunately, the statement (16) does not have the intended meaning. Because of the distributivity of CASE (and, presumably, PFORM, which stands for 'prepositional form'), the statement, when applied to a coordinate complement, has the following effect: either all conjuncts are instrumental, or all conjuncts have the prepositional form W. Making (any of) these features non-distributive does not solve the problem, as it would only render the relevant constraints inapplicable to individual conjuncts.

[8] Not to be confused with 'PRED'.

[9] We use here '∨' to mark disjunction, instead of the '|' usual in the LFG literature; similarly, we explicitly mark conjunction with '∧' instead of leaving it implicit.

There is, however, a solution which does not require extending the formal apparatus of LFG, although it is based on a relatively rarely used LFG mechanism, namely, the so-called off-path constraints (Dalrymple, 2001, p. 148).[10]

Off-path constraints make it possible to restrict the path (or, more importantly, its part) used by other statements. For example, while the minimal feature structure satisfying (17) is that of (18), the statement (19), with an off-path constraint added to the attribute A, specifies (20).[11]

(17) $(\uparrow \text{A B C}) =_c +$

(18) $\begin{bmatrix} \text{A} & \begin{bmatrix} \text{B} & \begin{bmatrix} \text{C} & + \end{bmatrix} \end{bmatrix} \end{bmatrix}$

(19) $(\uparrow \text{A B C}) =_c +$
$(\leftarrow \text{D}) =_c \text{E}$

(20) $\begin{bmatrix} \text{A} & \begin{bmatrix} \text{B} & \begin{bmatrix} \text{C} & + \end{bmatrix} \end{bmatrix} \\ \text{D} & \text{E} \end{bmatrix}$

More formally, '←' denotes the f-structure which contains the attribute to which it is attached, while '→' denotes the f-structure which is the value of the attribute to which it is attached. Hence, (21) (i.e., with '←' above replaced by '→') specifies the structure in (22).

(21) $(\uparrow \text{A B C}) =_c +$
$(\rightarrow \text{D}) =_c \text{E}$

(22) $\begin{bmatrix} \text{A} & \begin{bmatrix} \text{B} & \begin{bmatrix} \text{C} & + \end{bmatrix} \\ \text{D} & \text{E} \end{bmatrix} \end{bmatrix}$

The last piece of puzzle necessary to successfully analyse coordination of unlikes in LFG is the standard possibility to require the existence of a certain feature (whatever its value) in a feature structure: for example, (↑ PRED), without any equality sign, requires that the feature PRED be present in the feature structure being described. Given that PRED is a distributive feature, this specification forces all conjuncts to contain PRED (i.e., they must be semantically non-vacuous).

Returning to the failed attempt to formalise a disjunctive constraint in (16), let us note that the problem occurs because the disjunction is understood too early: instead of being interpreted as "for every conjunct, either... or...", it means: "either for every conjunct..., or for every conjunct...". What is needed is a means of 'smuggling' the disjunction into conjuncts before it is interpreted. Off-path constraints provide a mechanism to achieve this.

The relevant statement, replacing (16), is given below:

(23) $(\uparrow \text{OBL} \quad \text{PRED}) \\ (\leftarrow \text{CASE}) =_c \text{INST} \vee (\leftarrow \text{PFORM}) =_c \text{W}$

(23) ensures that – in Polish – there are no semantically vacuous (expletive) oblique complements, i.e., each such complement has a PRED value. This part of the statement is trivial. The main import of the statement is given in the off-path constraint part: for each such PRED, either the value of the CASE attribute (at the same level as the PRED) is instrumental, or the value of PFORM (again, at the same level) is W. As a result, disjunction is interpreted independently for each conjunct.

[10] We would like to thank Mary Dalrymple for suggesting to us that off-path constraints could be used to account for the coordination of unlikes, namely, for Polish unlike category subjects. (The usual disclaimers apply.) Przepiórkowski and Patejuk 2012 presents this analysis – and an alternative account – in more detail.

[11] Note that off-path constraints are placed *below* the attribute to which they apply.

5 Interactions with case assignment

Polish is a morphologically rich language with 7 grammatical cases. While case values of some arguments are stable and do not change with the syntactic environment (such case values are said to be assigned lexically or inherently), other case values depend on the syntactic context, e.g., the presence of negation or the categorial status of the argument. The syntactic (or structural) case assignment facts may be partially summarised as follows:[12]

(24) a. **subjects** bearing structural case are in the nominative,

b. with the exception of numeral phrase subjects, headed by so-called governing numerals (see below), which are in the accusative;

(25) a. **objects** bearing structural case are in the accusative,

b. unless they are in the syntactic scope of sentential negation, in which case they are in the genitive (so-called Genitive of Negation, GoN).

It should be possible to model these facts in a straightforward way by case assignment statements such as the following (for (24) above):[13]

(26) $[(\uparrow \textsc{subj acm}) =_c \textsc{rec} \wedge (\uparrow \textsc{subj case}) =_c \textsc{acc}] \vee (\uparrow \textsc{subj case}) =_c \textsc{nom}$

In this statement, ACM represents *accommodability*, a lexical feature introduced for Polish by Bień and Saloni (1982) to distinguish numeral forms governing the genitive noun (the value of ACM in such cases is REC) from numeral forms agreeing with the following noun.

Although, again, (26) does not have the intended meaning (its effect is that either all conjuncts are accusative numeral phrases or they are all nominative), the following version does the trick:

(27) $(\uparrow \textsc{subj} \quad \textsc{pred} \quad) \\ [(\leftarrow \textsc{acm}) =_c \textsc{rec} \wedge (\leftarrow \textsc{case}) =_c \textsc{acc}] \vee (\leftarrow \textsc{case}) =_c \textsc{nom}$

Such statements are successful in handling coordinate arguments, as in the example below.[14]

(28) [Pan Mirosław] i [czternastu ludzi] pracowało dzień i noc.
Mr Mirosław.NOM.SG and fourteen.ACC.PL man.GEN.PL worked.3.SG.N day and night
'Mr Mirosław and fourteen men worked night and day.' (NKJP)

On the other hand, just as in English, some Polish verbs allow for either an NP or a clause (or their coordination) in an argument position (here: subject); a relevant example is given below:

(29) Jana dziwiło, [że Maria wybiera Piotra], i [jej brak gustu].
Jan.ACC puzzled.3.SG.N that Maria chooses Piotr and her lack of taste.NOM.SG
'(The fact) that Maria prefers Piotr and her lack of taste puzzled Jan.'
(Świdziński, 1992, 1993)

For such verbs, the following constraint holds:

[12] See Przepiórkowski 1999 for extensive justification. We limit our considerations to arguments of verbs here.

[13] Where necessary, square brackets are used for the purpose of grouping constraints.

[14] "NKJP" marks attested examples found in the National Corpus of Polish (`http://nkjp.pl/`; Przepiórkowski et al. 2010, 2012).

(30) (↑ SUBJ PRED)
[(← ACM) =$_c$ REC ∧ (← CASE) =$_c$ ACC] ∨
(← CASE) =$_c$ NOM ∨ (← COMP-FORM) =$_c$ ŻE

A similar account can be offered for syntactically case-assigned objects, which – for some verbs – may be alternatively realised as clauses. Consider the following examples:

(31) Doradził mu [wyjazd] i [żeby nie wracał].
advised him leave.ACC and that NEG come back
'He advised him to leave and not to come back.' Kallas (1993)

(32) (Wcale) nie doradził mu [wyjazdu] ani [żeby nie wracał].
not at all NEG advised him leave.GEN nor that NEG come back
'He did not advise him to leave nor not to come back.'

These examples illustrate the principles of structural case assignment in Polish referring to objects (see (25) above): verbs assign accusative case to nominal objects in the absence of negation, as in (31), while genitive is assigned if negation is present, see (32). This basic principle is formalised in (33).

(33) (↑ OBJ PRED)
[¬((OBJ ←) NEG) ∧ (← CASE)=$_c$ ACC] ∨
[((OBJ ←) NEG)=$_c$ + ∧ (← CASE)=$_c$ GEN]

However, in case of verbs such as DORADZIĆ 'advise', as in (32), another disjunct must be added, allowing for a clausal object:

(34) (↑ OBJ PRED)
[¬((OBJ ←) NEG) ∧ (← CASE)=$_c$ ACC] ∨
[((OBJ ←) NEG)=$_c$ + ∧ (← CASE)=$_c$ GEN] ∨
(← COMP-FORM) =$_c$ ŻEBY

Again, the distributive feature PRED is used as an anchor so that disjunctive off-path constraints are checked against each element of OBJ independently (so, each conjunct under coordination). Note that a combination of outside-in (used so far) and inside-out equations is employed in (33)–(34). The former designate a path leading downwards into the f-structure, the latter make it possible to construct a path leading in the opposite direction, namely to higher f-structures; '(OBJ ←)' creates a path starting from the f-structure designated by '←', leading to the one which contains OBJ. Paths constructed in this way may be used as designators in outside-in equations as in the first disjunct of (33)–(34) which requires that the particular OBJ f-structure be marked for accusative case and that there be no negation in the f-structure containing OBJ (see '¬((OBJ ←) NEG)'). The second disjunct uses the same mechanisms to ensure that a given OBJ f-structure be marked for genitive case and that there be negation in the f-structure which contains OBJ (see '((OBJ ←) NEG)=$_c$ +').[15]

6 Implications for valence lexicons

Any adequate account of coordination of unlike categories implies a certain organisation of the valence dictionary, i.e., a lexicon recording information about arguments of predicates. Such a

[15]This solution may be extended to handle optional GoN under transferred negation in verb chains. This is achieved using functional uncertainty in relevant off-path constraints.

lexicon should make it clear whether arguments in the same position can be coordinated.

Note that it is not enough to know that various categorial kinds of arguments bear the same grammatical function in relation to the predicate. For example, the Polish verb MÓWIĆ 'say, speak' takes as its (passivisable) object the following categories, among others: *oratio recta* (direct speech; marked as `or` below), prepositional phrase headed by O and requiring a locative phrase (`prepnp(o,loc)`), subordinate finite clause introduced by the complementiser ŻE (`cp(że)`), embedded question (`cp(intrel)`), etc. While various of these possible objects may be coordinated, apparently an *oratio recta* object cannot be coordinated with any other category. This calls for postulating two different subcategorisation frames: one involving an *oratio recta* object, and another involving other kinds of objects.

Valence dictionaries we are aware of, Polish or otherwise, do not include such 'coordinatability' information. However, a new valence dictionary, Walenty,[16] is being developed for Polish which encodes such information explicitly. In this valence dictionary, any frame allowing for the coordination of unlikes is accompanied by an attested example, usually from the National Corpus of Polish. Figure 1 shows a screenshot of the application assisting in the creation of this dictionary.

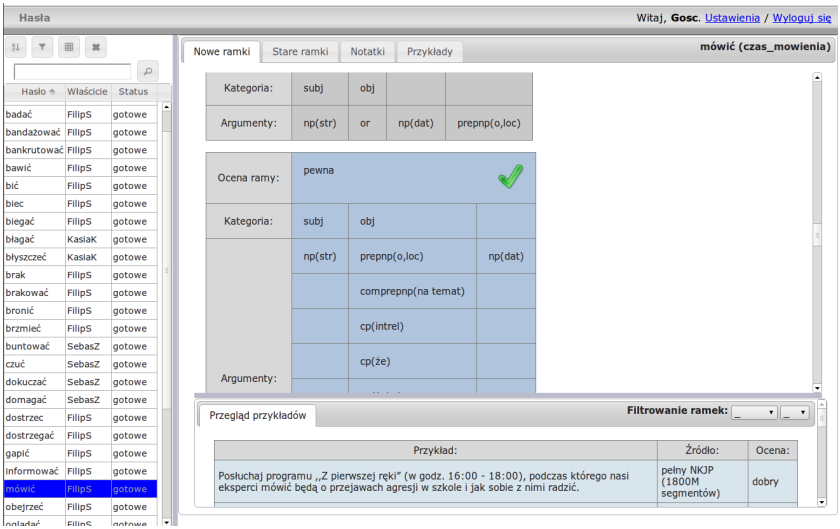


Figure 1: An application for creating Walenty, a new valence dictionary of Polish – a screenshot showing subcategorisation frames for MÓWIĆ 'say, speak': one involving an *oratio recta* object (near the top), and another involving an object which may be realised as a prepositional phrase, an embedded clause, etc. (in the bottom). An example from the National Corpus of Polish, illustrating the possibility to coordinate `prepnp(o,loc)` and `cp(intrel)` is provided at the very bottom.

[16] The preliminary version is available at: `http://clip.ipipan.waw.pl/Walenty`.

7 Coordination of very unlikes

Standard coordination of unlike categories involves elements which bear the same grammatical function. For example, in (12) above, both *the appointment* and *that it was important to be on time* correspond to the direct object of *remembered*. However, in some languages and under certain circumstances it is possible to coordinate elements bearing starkly different grammatical functions, as in Polish (35), involving coordination of the subject and the direct object of the verb *uczył* 'teach'.

(35) Kto i kogo będzie uczył?
who.NOM and who.ACC will teach
'Who will teach whom?' (NKJP)

To the best of our knowledge, such cases, called in English – after Mel'čuk 1988, p. 40 – *lexico-semantic coordination* or – as in Chaves and Paperno 2007, who provide a preliminary HPSG analysis – *hybrid coordination*, were first described for Russian (and in Russian) in Sannikov 1979, 1980, and for Polish (and in Polish) in Kallas 1993. Similar data are also discussed for French and Hungarian, and given the first LFG account, in Gazdik 2010 (only for *wh*-words, though).

Examination of Polish data reveals that such coordination is more robust than previously described in the literature. First of all, elements which enter lexico-semantic coordination are not only *wh*-phrases, as in (35), but also phrases containing a pronoun expressing an existential quantifier, a universal quantifier or phrases with n-words, as in (36), (37) and (38), respectively.

(36) czy komukolwiek, kiedykolwiek i do czegokolwiek przydał się poradnik
PART anybody.DAT anytime and for anything.GEN come in handy guide
'Has a(ny) guide ever come in handy to anybody for anything?' (NKJP)

(37) Obiecać można wszystko i wszystkim.
promise may everything.ACC and everyone.DAT
'One may promise everything to everyone.' (NKJP)

(38) nikogo i nic nie może tłumaczyć.
nobody.GEN and nothing.NOM NEG can excuse
'Nothing may excuse anybody.' (NKJP)

Second, as already evidenced in (38), and contra theoretical assumptions of the HPSG and LFG analyses mentioned above, the coordinated elements may be dependents of different predicates. This is more clearly illustrated in (39), where *skąd* 'where from' is an adjunct of the verb *otrzymujemy* 'receive' and *jakie* 'what kind' is a modifier of the noun *informacje* 'information'.

(39) Skąd i jakie otrzymujemy informacje?
whence and what receive information
'What information and where from do we receive?' (NKJP)

While coordinated elements may be dependents of different heads, the semantics of (35)–(39) is that of a single clause; for example, (35) may be translated into 'Who will teach whom?', and not into 'Who will teach (somebody) and whom will (somebody) teach?'. Moreover, when the conjunction is *i* 'and', it may often be omitted, without any obvious change of meaning:

(40) Kto kogo będzie uczył?
who.NOM who.ACC will teach
'Who will teach whom?'

This is reflected in the implemented LFG analysis, where the main c-structure rule handling lexico-semantic coordination is (41), where *type* is a parameter which may assume the following values: WH (question word; cf. (35) and (39)), ANY (existential quantifier; cf. (36)), ALL (universal quantifier; cf. (37)), NEG (n-word; cf. (38)).

(41) $\text{XPlexsem-mono}_{type} \rightarrow \underset{\uparrow=\downarrow}{\text{XPextr}_{type}} \; [, \; \underset{\uparrow=\downarrow}{\text{XPextr}_{type}}]^* \; \text{CONJ} \; \underset{\uparrow=\downarrow}{\text{XPextr}_{type}}$

According to this rule, the f-structure of each extracted phrase of a given type is identified with the f-structure of the mother. This is achieved using the '↑=↓' annotation on all conjuncts which treats them as co-heads: f-structure fragments built by particular conjuncts, see (42), are unified in one top-level f-structure, resulting in (43). The latter, once it is unified with the f-structure of the rest of the utterance, renders (44).

(42) a. $\left[\text{ADJ} \left\{\left[\text{PRED} \quad \text{'WHENCE'}\right]\right\}\right]$

b. $\left[\text{OBJ} \left[\text{ADJ} \left\{\left[\text{PRED} \quad \text{'WHAT'}\right]\right\}\right]\right]$

(43) $\begin{bmatrix} \text{ADJ} & \left\{\left[\text{PRED} \quad \text{'WHENCE'}\right]\right\} \\ \text{OBJ} & \left[\text{ADJ} \left\{\left[\text{PRED} \quad \text{'WHAT'}\right]\right\}\right] \end{bmatrix}$

(44) $\begin{bmatrix} \text{PRED} & \text{'RECEIVE}\langle\boxed{1},\boxed{2}\rangle\text{'} \\ \text{SUBJ } \boxed{1} & \left[\text{PRED} \quad \text{'PRO'}\right] \\ \text{OBJ } \boxed{2} & \begin{bmatrix} \text{PRED} & \text{'INFORMATION'} \\ \text{ADJ} & \left\{\left[\text{PRED} \quad \text{'WHAT'}\right]\right\} \end{bmatrix} \\ \text{ADJ} & \left\{\left[\text{PRED} \quad \text{'WHENCE'}\right]\right\} \end{bmatrix}$

The complete analysis, which is not given here in detail for lack of space (e.g., the definition of XP_{type} is omitted), also includes the rule (45) which, together with (46)–(47), encodes appropriate locality (or island) constraints on the provenance of coordinated phrases.[17] While (46) defines the set of grammatical functions which may be assigned to a given element, (47) defines the main extraction path of such elements: they may be extracted across any number of infinitival complements (XCOMP) and, below that, across any number of grammatical functions GF, but – crucially – not across finite sentential complements, etc.[18] Since each conjunct resolves its functional annotation independently, it is possible for every conjunct to bear entirely different grammatical function and depend on different heads, as in (42a) vs (42b).

(45) $\text{XPextr}_{type} \rightarrow \underset{(\uparrow \text{ PATH } \text{GF}^+)=\downarrow}{\text{XP}_{type}}$

(46) $\text{GF} \equiv \{\text{SUBJ} \,|\, \text{OBJ} \,|\, \text{OBJ}_\theta \,|\, \text{OBL} \,|\, \text{ADJ} \in\}$

(47) $\text{PATH} \equiv \text{XCOMP}^*$

Let us finish this section with the observation that there is a subtype of lexico-semantic coordination which is not amenable to the above analysis:

[17] The complete analysis is presented in Patejuk and Przepiórkowski 2012a.

[18] For *wh*-phrases (47) is extended to allow for extraction from sentential complements: $\text{PATH} \equiv \text{COMP}^* \; \text{XCOMP}^*$, to account for examples such as *Kto i kogo chciałeś, żeby zaprosił?* 'Who did you want to invite whom?', lit.: 'who and whom you-wanted that invite.FIN'.

(48) Nie wiadomo było, czy *(i) kiedy wróci.
NEG know was whether and when returns
'It was not clear whether and when (s)he would return.' (NKJP)

As indicated by the asterisk and the parentheses, this example is ungrammatical once the conjunction *i* is omitted. Moreover, the meaning of the embedded question is biclausal: 'It was not clear whether (s)he would return, and – if (s)he did – when (s)he would return.' Such cases, where the first conjunct in the lexico-semantic coordination is the question particle *czy*, are handled by the following c-structure rule:[19]

(49) $\text{XPlexsem-bi}_{wh} \rightarrow \underset{\downarrow\in\uparrow}{\text{PART}_{wh}} \; [, \; \underset{\downarrow\in\uparrow}{\text{XPextr}_{wh}}]^* \; \text{CONJ} \; \underset{\downarrow\in\uparrow}{\text{XPextr}_{wh}}$

The crucial difference between (41) and (49) is that f-structures of the conjuncts in the latter are not identified with the mother, but become elements of the set representation of the mother; cf. the representation of the conjuncts in (50) and the resulting coordination in (51).

(50) a. $\left[\text{CLAUSE-TYPE} \quad \text{INT}\right]$

b. $\left[\text{ADJ} \quad \left\{\left[\text{PRED} \quad \text{'WHEN'}\right]\right\}\right]$

(51) $\left\{\left[\text{CLAUSE-TYPE} \quad \text{INT}\right], \left[\text{ADJ} \quad \left\{\left[\text{PRED} \quad \text{'WHEN'}\right]\right\}\right]\right\}$

When (51) is unified with the rest of the utterance, a biclausal coordinate structure results, as illustrated in (52):

(52) $\left\{\left[\begin{array}{l}\text{PRED} \quad \text{'RETURN}\langle\boxed{1}\rangle\text{'} \\ \text{SUBJ} \; \boxed{1} \quad \left[\text{PRED} \quad \text{'PRO'}\right] \\ \text{CLAUSE-TYPE} \quad \text{INT}\end{array}\right], \left[\begin{array}{l}\text{PRED} \quad \text{'RETURN}\langle\boxed{1}\rangle\text{'} \\ \text{SUBJ} \quad \boxed{1} \\ \text{ADJ} \quad \left\{\left[\text{PRED} \quad \text{'WHEN'}\right]\right\}\end{array}\right]\right\}$

8 Towards evaluation

As mentioned in the Introduction, coordination is textually frequent, so any parser should be able to handle it. But exactly how frequent is it?

There are two balanced subcorpora within the National Corpus of Polish: one – let us call it NKJP1M – is manually annotated and contains around 1 million words, another one – NKJP250M – contains 250 million words tagged automatically. As shown in Table 1, depending on which subcorpus is used to gather statistics, between 37.5% and 38.7% of all sentences involve coordination, which amply justifies the claim advanced in the Introduction.

It is considerably more difficult to estimate how many of these coordinate structures involve unlike categories. Instead, let us concentrate on cases of lexico-semantic coordination, which are easier to find automatically. Of these, undoubtedly the most frequent – and the ones most readily noticed in the linguistic literature – are cases of coordination of *wh*-phrases, as in (35) and (39) above. This subtype of lexico-semantic coordination is especially important in NLP, for example, in question answering, where Polish users will naturally use such constructions.

[19] There are examples where the question particle is the last conjunct. Handling these requires simple modifications.

corpus	# of sentences	# of conjunctions	# of sentences containing conjunctions	% of sentences containing conjunctions
NKJP1M	85 663	44 841	32 147	37.5
NKJP250M	18 625 185	10 455 657	7 210 648	38.7

Table 1: The number of conjunctions and the percentage of sentences containing conjunctions in balanced subcorpora of the National Corpus of Polish.

In NKJP250M, there are 272 results of the query '`Kto [orth="i|lub|oraz|albo"] [orth!=co/i]`', where *kto* means 'who' (nominative), *i*, *lub*, *oraz* and *albo* are the most frequent conjunctions, and *co* means 'what' (nominative or accusative); the last part of this query ensures that structures like *Kto i co*, i.e., possibly involving two nominative phrases, are not among the reported results.[20] Compared to over 18 million sentences of NKJP250M, this number seems to be infinitesimal, but note that this is only one subtype of *wh*-lexico-semantic coordination; *Gdzie i...* 'where and...' occurs 157 times, *Komu i...* 'who.DAT and' occurs 47 times, etc.[21] These numbers are sufficiently large to show that this is not a marginal construction in Polish.

The LFG grammar of Polish (Patejuk and Przepiórkowski, 2012b), which includes the analysis of coordination sketched above, is implemented in the XLE system developed at PARC.[22] It is based on two previous implemented grammars of Polish: its c-structure is based on a DCG (Warren and Pereira, 1980) grammar used by the parser Świgra (Świdziński, 1992; Woliński, 2004, 2005), while its f-structure is inspired by an HPSG grammar (Przepiórkowski et al., 2002; Marciniak et al., 2003). The quality of the grammar is ensured in a two-fold way: using treebank testing, but also verifying the linguistic coverage against manually constructed testsuites.

The former takes the form of reparsing Składnica (Woliński et al., 2011), a treebank of Polish containing parses for sentences extracted from the manually annotated subcorpus of the National Corpus of Polish and parsed using Świgra. The treebank coverage of the LFG grammar amounts to 90%. Unfortunately, the current version of Składnica contains sentences which were relatively unproblematic for Świgra and for human annotators, i.e., it is skewed towards simple and short sentences. In particular, coordination is underrepresented in general (1869 out of 8227 sentences, or 22.7%, contain conjunctions), and no cases of lexico-semantic coordination were found by the authors.[23] The complete NKJP1M, from which Składnica draws its sentences, does contain instances of lexico-semantic coordination, e.g., *Kto i dlaczego boi się prywatyzacji?* 'Who and why is afraid of privatisation?'.

While Składnica currently contains good parses for 8227 sentences, it contains many more sentences for which human annotators could not identify a good parse among the trees

[20] The query syntax is described in more detail at `http://nkjp.pl/poliqarp/help/en.html`.

[21] Also note that such queries only find cases of coordination of two *wh*-phrases, while – as expected – it is possible to coordinate more of them, as in the attested (NKJP) *Gdzie, jak i za ile będą się bawić*, lit. 'Where, how and for how much will they have fun'. Since the first word is capitalised, such queries also do not target embedded questions.

[22] Maxwell and Kaplan 1996; `http://www2.parc.com/isl/groups/nltt/xle/`

[23] The treebank search engine available at `http://nlp.ipipan.waw.pl:8000/ui.xhtml` has been employed, and queries such as '`[base = /kto/ & orth = /K.*/]`' were given, which should find any capitalised form of the pronoun KTO 'who'. Out of 39 sentences beginning with a form of KTO, 57 beginning with a form of CO 'what', 22 beginning with KIEDY 'when', 21 beginning with DLACZEGO 'why' and 5 beginning with GDZIE 'where', none contained lexico-semantic coordination.

generated by the DCG grammar. It is hoped that extensions related to coordination, including the two described above, make it possible to parse sentences which have been rejected so far due to the limitations of the previous grammar and its valence dictionary, which did not take coordination of unlikes into consideration.

The other method of testing relies on constructed sentences: currently there are over 1200 items which provide a means of comprehensive grammar testing, making it possible to test a wide range of phenomena, ensuring proper handling of fundamental issues, but also giving an opportunity to test very sophisticated phenomena where complex interactions between various areas of the grammar are involved. Also, while treebank testing is limited to positive examples exclusively, there are numerous negative examples among constructed testsuite items. This testsuite is currently being extended with examples of unlike and lexico-semantic coordination, so no quantitative results can be cited here (they should be available by the time of COLING 2012), but the grammar correctly parses any sentences involving such difficult instances of coordination that we have come across.

Conclusion

While currently a niche activity in NLP, grammar engineering is actively pursued and finds high-profile applications, such as the use of the English LFG grammar in Microsoft's Bing search engine.[24] Since coordination is textually very frequent, any self-respecting grammar should allow for a comprehensive treatment of this phenomenon. The analysis presented above, taking into account not only run of the mill cases of coordination, but also coordination of unlike and very unlike constituents, is implemented within a large LFG grammar of Polish.

It must be stressed, however, that – although the details of the interaction of coordination with case assignment and other phenomena vary from language to language – the general mechanisms described in sections 4–7 seem to be applicable to any language. In particular, coordination of unlikes has been by now reported for many languages, syntactic case assignment mechanisms which interact with coordination can be observed in Slavic, Baltic and Finno-Ugric languages (to limit ourselves to European languages), and lexico-semantic coordination has been described for French, Hungarian, Romanian and a number of Slavic languages.

Due to space constraints, many aspects of the implementation of coordination in the LFG grammar of Polish have been omitted here. Probably the most important is agreement between the verb (or, more generally, a predicate) and its coordinated subject, but also the interesting phenomenon of single conjunct agreement. However, unlike coordination of unlikes, these phenomena already have established accounts in the LFG literature, e.g., Dalrymple and Nikolaeva 2006, Kuhn and Sadler 2007 and Dalrymple et al. 2009, and their standard analyses are implemented in the current grammar. The first stable version of the grammar will be made publicly available by the end of January 2013 at `http://zil.ipipan.waw.pl/LFG`.

Acknowledgements

This research is supported by the POIG.01.01.02-14-013/09 project which is co-financed by the European Union under the European Regional Development Fund. The publication of the resulting implemented grammar is supported by the CESAR project (European project CIP ICT-PSP 271022, part of META-NET).

[24] See `http://www.essex.ac.uk/linguistics/external/LFG/Bulletin/Jul09.html` and Ahmed and Hautli 2011, p. 9.

References

Ahmed, T. and Hautli, A. (2011). A first approach towards an Urdu WordNet. *Linguistics and Literature Review*, 1(1):1–14.

Bień, J. S. and Saloni, Z. (1982). Pojęcie wyrazu morfologicznego i jego zastosowanie do opisu fleksji polskiej (wersja wstępna). *Prace Filologiczne*, XXXI:31–45.

Bresnan, J., editor (1982). *The Mental Representation of Grammatical Relations*. MIT Press Series on Cognitive Theory and Mental Representation. The MIT Press, Cambridge, MA.

Butt, M. and King, T. H., editors (1996). *The Proceedings of the LFG'96 Conference*, Stanford, CA. CSLI Publications.

Butt, M. and King, T. H., editors (2010). *The Proceedings of the LFG'10 Conference*, Stanford, CA. CSLI Publications.

Butt, M. and King, T. H., editors (2012). *The Proceedings of the LFG'12 Conference*, Stanford, CA. CSLI Publications.

Chaves, R. P. and Paperno, D. (2007). On the Russian hybrid coordination construction. In Müller, S., editor, *The Proceedings of the 14th International Conference on Head-Driven Phrase Structure Grammar*, pages 46–64, Stanford. CSLI Publications.

Dalrymple, M. (2001). *Lexical-Functional Grammar*. Academic Press.

Dalrymple, M. and Hristov, B. (2010). Agreement Patterns and Coordination in Lexical Functional Grammar. In (Butt and King, 2010), pages 186–206.

Dalrymple, M. and Kaplan, R. M. (2000). Feature indeterminacy and feature resolution. *Language*, 76(4):759–798.

Dalrymple, M., King, T. H., and Sadler, L. (2009). Indeterminacy by underspecification. *Journal of Linguistics*, 45:31–68.

Dalrymple, M. and Nikolaeva, I. (2006). Syntax of natural and accidental coordination: Evidence from agreement. *Language*, 82:824–849.

Gazdar, G., Klein, E., Pullum, G. K., and Sag, I. A. (1985). *Generalized Phrase Structure Grammar*. Blackwell / Harvard University Press, Cambridge / Cambridge, MA.

Gazdik, A. (2010). Multiple Questions in French and Hungarian: An LFG Account. In (Butt and King, 2010), pages 249–269.

Kallas, K. (1993). *Składnia współczesnych polskich konstrukcji współrzędnych*. Wydawnictwo Uniwersytetu Mikołaja Kopernika, Toruń.

Kosek, I. (1999). *Przyczasownikowe frazy przyimkowo-nominalne w zdaniach współczesnego języka polskiego*. Wydawnictwo Uniwersytetu Warmińsko-Mazurskiego, Olsztyn.

Kuhn, J. and Sadler, L. (2007). Single conjunct agreement and the formal treatment of coordination in LFG. In Butt, M. and King, T. H., editors, *The Proceedings of the LFG'07 Conference*, pages 302–322, University of Stanford, California, USA.

Levine, R. D. (2011). Linearization and its discontents. In Müller, S., editor, *Proceedings of the HPSG 2011 Conference*, pages 126–146, Stanford, CA. CSLI Publications.

Marciniak, M., Mykowiecka, A., Przepiórkowski, A., and Kupść, A. (2003). An HPSG-annotated test suite for Polish. In Abeillé, A., editor, *Treebanks: Building and Using Parsed Corpora*, volume 20 of *Text, Speech and Language Technology*, pages 129–146. Kluwer, Dordrecht.

Maxwell, III, J. T. and Kaplan, R. M. (1996). An efficient parser for LFG. In (Butt and King, 1996).

Maxwell, III, J. T. and Manning, C. D. (1996). A theory of non-constituent coordination based on finite-state rules. In (Butt and King, 1996).

Mel'čuk, I. (1988). *Dependency Syntax: Theory and Practice*. The SUNY Press, Albany, NY.

Patejuk, A. and Przepiórkowski, A. (2012a). Lexico-semantic coordination in Polish. In (Butt and King, 2012).

Patejuk, A. and Przepiórkowski, A. (2012b). Towards an LFG parser for Polish: An exercise in parasitic grammar development. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation, LREC 2012*, pages 3849–3852, Istanbul, Turkey. ELRA.

Peterson, P. G. (2004). Coordination: Consequences of a lexical-functional account. *Natural Language and Linguistic Theory*, 22:643–679.

Pollard, C. and Sag, I. A. (1987). *Information-Based Syntax and Semantics, Volume 1: Fundamentals*. Number 13 in CSLI Lecture Notes. CSLI Publications, Stanford, CA.

Pollard, C. and Sag, I. A. (1994). *Head-driven Phrase Structure Grammar*. Chicago University Press / CSLI Publications, Chicago, IL.

Przepiórkowski, A. (1999). *Case Assignment and the Complement-Adjunct Dichotomy: A Non-Configurational Constraint-Based Approach*. Ph.D. dissertation, Universität Tübingen, Germany.

Przepiórkowski, A., Bańko, M., Górski, R. L., and Lewandowska-Tomaszczyk, B., editors (2012). *Narodowy Korpus Języka Polskiego*. Wydawnictwo Naukowe PWN, Warsaw.

Przepiórkowski, A., Górski, R. L., Łaziński, M., and Pęzik, P. (2010). Recent developments in the National Corpus of Polish. In *Proceedings of the Seventh International Conference on Language Resources and Evaluation, LREC 2010*, Valletta, Malta. ELRA.

Przepiórkowski, A., Kupść, A., Marciniak, M., and Mykowiecka, A. (2002). *Formalny opis języka polskiego: Teoria i implementacja*. Akademicka Oficyna Wydawnicza EXIT, Warsaw.

Przepiórkowski, A. and Patejuk, A. (2012). On case assignment and the coordination of unlikes: The limits of distributive features. In (Butt and King, 2012).

Sag, I. A. (2002). Coordination and underspecification. In Kim, J.-B. and Wechsler, S., editors, *Proceedings of the HPSG 2002 Conference*, pages 267–291. CSLI Publications, Stanford, CA.

Sag, I. A., Gazdar, G., Wasow, T., and Weisler, S. (1985). Coordination and how to distinguish categories. *Natural Language and Linguistic Theory*, 3:117–171.

Sannikov, V. Z. (1979). Sočinitel'nye i sravnitel'nye konstrukcii: ix blizost', ix sintaksičeskoe predstavlenie I. *WSA*, 4:413–432.

Sannikov, V. Z. (1980). Sočinitel'nye i sravnitel'nye konstrukcii: ix blizost', ix sintaksičeskoe predstavlenie II. *WSA*, 5:211–242.

Steedman, M. (2000). *The Syntactic Process*. The MIT Press, Cambridge, MA.

Świdziński, M. (1992). *Gramatyka formalna języka polskiego*, volume 349 of *Rozprawy Uniwersytetu Warszawskiego*. Wydawnictwa Uniwersytetu Warszawskiego, Warsaw.

Świdziński, M. (1993). Dalsze kłopoty z bezokolicznikiem. In Sambor, J., Linde-Usiekniewicz, J., and Huszcza, R., editors, *Językoznawstwo synchroniczne i diachroniczne*, pages 303–314. Wydawnictwa Uniwersytetu Warszawskiego, Warsaw.

Warren, D. H. D. and Pereira, F. C. N. (1980). Definite clause grammars for language analysis — a survey of the formalism and a comparison with augmented transition networks. *Artificial Intelligence*, 13:231–278.

Woliński, M. (2004). *Komputerowa weryfikacja gramatyki Świdzińskiego*. Ph.D. dissertation, Instytut Podstaw Informatyki, Polska Akademia Nauk, Warsaw.

Woliński, M. (2005). An efficient implementation of a large grammar of Polish. *Archives of Control Sciences*, 15(3):251–258.

Woliński, M., Głowińska, K., and Świdziński, M. (2011). A preliminary version of Składnica—a treebank of Polish. In Vetulani, Z., editor, *Proceedings of the 5th Language & Technology Conference*, pages 299–303, Poznań.

Yatabe, S. (2004). A comprehensive theory of coordination of unlikes. In Müller, S., editor, *Proceedings of the HPSG 2004 Conference*, pages 335–355, Stanford, CA. CSLI Publications.

Simple and Effective Parameter Tuning for Domain Adaptation of Statistical Machine Translation

Pavel Pecina[1], Antonio Toral[2], Josef van Genabith[2]
(1) Faculty of Mathematics and Physics, Charles University in Prague, Czech Republic
(2) Centre for Next Generation Localisation, School of Computing, Dublin City University, Ireland
pecina@ufal.mff.cuni.cz, atoral@computing.dcu.ie, josef@computing.dcu.ie

ABSTRACT
Current state-of-the-art Statistical Machine Translation systems are based on log-linear models that combine a set of feature functions to score translation hypotheses during decoding. The models are parametrized by a vector of weights usually optimized on a set of sentences and their reference translations, called development data. In this paper, we explore a (common and industry relevant) scenario where a system trained and tuned on general domain data needs to be adapted to a specific domain for which no or only very limited in-domain bilingual data is available. It turns out that such systems can be adapted successfully by re-tuning model parameters using surprisingly small amounts of parallel in-domain data, by cross-tuning or no tuning at all. We show in detail how and why this is effective, compare the approaches and effort involved. We also study the effect of system hyperparameters (such as maximum phrase length and development data size) and their optimal values in this scenario.

TITLE AND ABSTRACT IN CZECH

Jednoduchá a efektivní optimalizace parametrů pro doménovou adaptaci statistického strojového překladu

Současné systémy statistického strojového překladu jsou založeny na logaritmicko-lineárních modelech, které pro hodnocení překladových hypotéz ve fázi dekódování kombinují sadu příznakových funkcí. Tyto modely jsou parametrizovány vektorem vah, které se optimalizují na tzv. vývojových datech, tj. množině vět a jejich referenčních překladů. V tomto článku se zabýváme (často a pro průmyslové nasazení relevantní) situací, kdy je třeba překladový systém natrénovaný na datech z obecné domény adaptovat na nějakou specifickou doménu, pro kterou jsou k dispozici paralelní data jen ve velice omezeném (či žádném) množství. Ukazujeme, že takové systémy mohou být vhodně adaptovány pomocí optimalizace parametrů za použití jen překvapivě malého množství paralelních doménově-specifických dat nebo tzv. křížovou optimalizací. Možností je také nepoužití optimalizace vůbec. Jednotlivé přístupy analyzujeme a porovnáváme jejich cekovou náročnost. Dále se zabýváme analýzou systémových hyperparametrů (např. maximální délkou frází a velikostí vývojových dat) a jejich optimalizací.

KEYWORDS: machine translation, domain adaptation, parameter optimization.

KEYWORDS IN CZECH: strojový překlad, doménová adaptace, optimalizace parametrů.

Proceedings of COLING 2012: Technical Papers, pages 2209–2224,
COLING 2012, Mumbai, December 2012.

1 Introduction

Statistical Machine Translation (SMT) is an instance of a machine learning application and, in general, will work best if the data for training and testing are drawn from the same distribution (i.e. domain, genre, and style). In practice, however, it is often difficult to obtain sufficient amounts of in-domain data (in particular parallel data required for translation and distortion models) to train a well performing system for a specific domain.

Recently, Pecina et al. (2011) showed that just using in-domain development data for parameter tuning improves output quality of a Phrase-Based SMT (PB-SMT) system trained on general--domain data but applied to a specific domain. Although further additional improvements can be realized by using in-domain parallel and/or monolingual training data, parameter tuning on in-domain data requires only a relatively small set of parallel sentences, which is often easier to obtain. They report on a series of experiments carried out on the domains of Natural Environment (*env*) and Labour Legislation (*lab*) and two language pairs: English–French and English–Greek (both directions) and observe a substantial average relative improvement of 25% in terms of BLEU (Papineni et al., 2002) when switching from general-domain to in-domain tuning.

In this paper, we corroborate the results reported by Pecina et al. (2011), carrying out similar experiments on the domain of medical texts (*med*). In contrast to earlier work, we explain the improvements brought about by specific domain tuning by analysing the results in detail. In a nutshell: domain tuning for matching-domain training, tuning and test data results in feature vectors that trust (often long) translation table entries, while tuning with and for specific domains (while using generic training data) allows the MT system to stitch together translations from smaller bits and pieces with significantly more reordering, effectively undoing or "de--tuning" any previous optimizations. In a sense, this is natural: substantial divergence between test and training data means that in particular long and potentially high quality phrase pairs obtained in training may no longer be applicable to the test data and that this divergence can only be bridged by smaller translation units and more flexible recombination. Furthermore, our findings show that in the general-domain training and specific-domain test scenario, approaches that do not perform any parameter tuning (at all) or that tune on other specific development sets may in fact fare better than tuning on general-domain data. In addition, there is a question of how much specific-domain tuning data is in fact required to "de-tune" a general domain system to a specific domain. Finally, given the fact that a general-domain system can only use limited length translation units when translating specific-domain data, we explore limited length training and decoding.

After a brief overview of the log-linear model including its parameter optimization and an overview of the state-of-the-art in domain adaptation for SMT, we describe our experiments, present the results, the analysis, explore the resulting research questions with additional experiments, and conclude.

2 Phrase-Based Statistical Machine Translation

In PB-SMT, implemented e.g. in Moses (Koehn et al., 2007), an input sentence is segmented into sequences of consecutive words, called phrases. Each phrase is then translated into a target language phrase, which may be reordered with other translated phrases to produce the output.

Formally, the model is based on the noisy channel model. The translation **e** of an input sentence **f** is searched for by maximizing the translation probability $p(\mathbf{e}|\mathbf{f})$ formulated as a log-linear combination of a set of feature functions h_i and their weights λ_i:

$$p(\mathbf{e}|\mathbf{f}) = \prod_i^n h_i(\mathbf{e},\mathbf{f})^{\lambda_i}$$

Typically, the components include features of the following models: *phrase translation model*, which ensures that the source and target phrases are good translations of each other (e.g. direct and inverse phrase translation probability, direct and indirect lexical weighting, and phrase penalty), *language model*, which ensures that the translations are fluent, *reordering* (*distortion*) *model*, which allows to reorder phrases in the input sentences (e.g. distance-based and lexicalized reordering) and *word penalty*, which prevents the translations from being too long or too short. These models are trained on either parallel or monolingual training data.

The weights of the log-linear combination influence overall translation quality; however, the optimal setting depends on the translation direction and data. A common solution to optimise weights is to use Minimum Error Rate Training (MERT), proposed by Och (2003), which automatically searches for the values that minimize a given error measure (or maximize a given translation quality measure) on a development set of parallel sentences. Theoretically, any automatic measure can be used for this purpose; however, the most commonly used is BLEU (Papineni et al., 2002). The search algorithm is a type of coordinate ascent: considering n-best translation hypotheses for each input sentence, it updates the feature weight most likely promising to improve the objective and iterates until convergence. The error surface is highly non-convex and as the algorithm cannot explore the whole parameter space, it may converge to a local maximum; in practise, it often produces good results (Bertoldi et al., 2009).

3 Domain adaptation in Statistical Machine Translation

Domain-adaptation is a very active research topic within the area of SMT. Three main topics can be identified: (i) combination of in-domain and out-of-domain resources for training, (ii) training data selection, and (iii) acquisition of specific-domain data. Below we briefly review a selection of relevant work that falls into these topics.

The first attempt to perform domain adaptation was carried out by Langlais (2002), who integrated in-domain lexicons in the translation model. Koehn and Schroeder (2007) integrate in-domain and out-of-domain language models as log-linear features in Moses. Nakov (2008) combines in-domain translation and reordering models with out-of-domain models. Finch and Sumita (2008) use a probabilistic mixture model combining two models for questions and declarative sentences with a general model.

Training data selection is another approach to domain-adaptation. The assumption is that a general-domain corpus, if sufficiently broad, includes sentences that resemble the target domain. Eck et al. (2004) present a technique for adapting the language model by selecting similar sentences from available training data. Hildebrand et al. (2005) extended this approach to the translation model. Foster et al. (2010) weigh phrase pairs from out-of-domain corpora according to their relevance to the target domain.

Munteanu and Marcu (2005) extract in-domain sentence pairs from comparable corpora. Daumé III and Jagarlamudi (2011) attempt to reduce out-of-vocabulary terms when targeting a specific domain by mining their translations from comparable corpora. Bertoldi et al. (2009) rely on large in-domain monolingual data to create synthetic parallel corpora for training.

languages (L1-L2)	*dom*	*set*	*sentences*	*L1 tokens*	*/ voc*	*L2 tokens*	*/ voc*
English–French	*gen*	train	1,725,096	47,956,886	73,645	53,262,628	103,436
		dev	2,000	58,655	5,734	67,295	6,913
		test	2,000	57,951	5,649	66,200	6,876
	env	dev	1,392	41,382	4,660	49,657	5,542
		test	2,000	58,865	5,483	70,740	6,617
	lab	dev	1,411	52,156	4,478	61,191	5,535
		test	2,000	71,688	5,277	84,397	6,630
	med	dev	1,064	16,807	3,484	18,932	4,865
		test	2,000	31,725	5,268	34,884	7,331
English–Greek	*gen*	train	964,242	27,446,726	61,497	27,537,853	173,435
		dev	2,000	58,655	5,734	63,349	9,191
		test	2,000	57,951	5,649	62,332	9,037
	env	dev	1,000	27,865	3,586	30,510	5,467
		test	2,000	58,073	4,893	63,551	8,229
	lab	dev	506	15,129	2,227	16,089	3,333
		test	2,000	62,953	4,022	66,770	7,056
	med	dev	1,064	16,807	3,484	20,625	3,893
		test	2,000	31,725	5,268	38,614	5,754

Table 1: Statistics of the training, development and test data sets from the domains used in the experiments including the number of sentence pairs, tokens, and vocabulary size (*voc*).

Pecina et al. (2011) exploit automatically web-crawled in-domain resources for parameter optimization and improving language models. Pecina et al. (2012) extend the work by using the web-crawled resources to also improve translation models.

4 Experimental setup

Our experimental setup follows and extends the one used in Pecina et al. (2011). In addition to the two evaluation domains (*env*, *lab*) used in that work, and in order to corroborate their earlier findings, we also carry out experiments on medical domain data (*med*).

4.1 Data

Our general-domain system is trained on the Europarl parallel corpus (Koehn, 2005, v5) extracted from the proceedings of the European Parliament and for the purposes of this work considered to contain general-domain texts (it covers a very broad range of topics and it is to a considerable extent spoken language). The general-domain development and test data used for parameter optimization and testing, respectively, are adopted from the WPT 2005[1] machine translation shared task. These sets were extracted from the same source as Europarl and contain 2,000 sentence pairs each.

The specific-domain development and test data for the *env* and *lab* domains were acquired by domain-focused web-crawling within the PANACEA project[2] and are available from the ELRA catalogue[3] under reference numbers ELRA-W0057 and ELRA-W0058. The entire acquisition procedure is described in detail in Pecina et al. (2011). The test sets consist of 2,000 sentence pairs each and the amount of sentence pairs in the development sets varies from 506 to 2,000.

[1] http://www.statmt.org/wpt05/
[2] http://www.panacea-lr.eu/
[3] http://catalog.elra.info/

dev	*test*	*English–French*		*French–English*		*English–Greek*		*Greek–English*	
gen	*gen*	49.12	*0.00*	57.00	*0.00*	42.24	*0.00*	44.15	*0.00*
	env	28.03	*−42.94*	31.79	*−44.23*	20.20	*−52.18*	29.23	*−33.79*
	lab	22.26	*−54.68*	27.00	*−52.63*	22.92	*−45.74*	31.71	*−28.18*
	med	12.32	*−74.92*	15.33	*−73.11*	8.96	*−78.79*	14.79	*−66.50*
average			*−57.51*		*−56.65*		*−58.90*		*−42.82*

Table 2: Results (in BLEU) of the systems tuned on general-domain and tested on the specific domains (*env*, *lab*, *med*) compared with the results on the general domain (*gen*); the figures in italics indicate the relative change (in percentage).

The *med* development and test data were extracted from the EMEA parallel corpus of texts from the European Medicines Agency, distributed as a part of the OPUS corpus (Tiedemann, 2009). A set of 3,500 parallel sentences in English, French, and Greek was randomly sampled from the sentence-aligned corpus data and manually checked for translation quality. Correct sentences were left untouched, sentences with minor errors were corrected, and those which required major corrections or were misaligned were discarded completely. We aimed at acquiring at least 3,000 correct sentence pairs: 2,000 for the test sets and the rest for the development sets. Finally, the test and development sets contained 2,000 and 1,064 sentence pairs respectively. All data sets used in our experiments contain one reference translation. Statistics are given in Table 1.

4.2 System description

Our MT system is based on the Moses PB-SMT system (Koehn et al., 2007). For training, all data sets are tokenized and lowercased using the Europarl tools. The original (non-lowercased) target side of the parallel data is kept for training the Moses recaser. The lowercased versions of the target side are used for training an interpolated 5-gram language model with Kneser-Ney discounting using the SRILM toolkit (Stolcke, 2002). Translation models are trained on the Europarl corpus, lowercased, and filtered on sentence level; we kept all sentence pairs having less than 100 words on each side and with length ratio within the interval $\langle 0.11, 9.0 \rangle$. The maximum length for aligned phrases is set to seven and the reordering models are generated using the following parameters: *distance, orientation-bidirectional-fe*. The resulting system combines 14 feature functions, listed below.

1. distance reordering score
2-7. lexicalised reordering scores
8. language model score
9. inverse phrase translation probability
10. inverse lexical weighting
11. direct phrase translation probability
12. direct lexical weighting
13. phrase penalty
14. word penalty

The corresponding parameters are optimized on the development sets by MERT. For decoding, test sentences are tokenized and lowercased. After translation, letter casing is reconstructed by the recaser and extra blank spaces are removed in order to produce human-readable text.

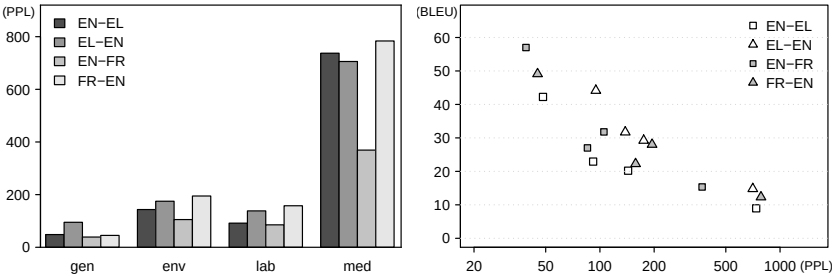


Figure 1: Perplexity (PPL) of the source side of the test sets given the language models trained on the source side of the training sets (left). Perplexity of the source side of the test data versus BLEU scores of the corresponding systems tuned on general-domain development data (right).

5 Experiments

Translation quality in our experiments is automatically evaluated using BLEU (Papineni et al., 2002) and all BLEU scores are reported as percentages.

5.1 Baseline system performance

Performance of the baseline system trained and tuned on the general-domain data and tested on the same domain varies from 42.24 to 57.00 (row 1 in Table 2). Applying the baseline general--domain system on the specific-domain data leads to significant degradation of translation quality (Banerjee et al., 2010; Wu et al., 2008). Pecina et al. (2011) reported an average decrease of 44.3% when the general-domain system was applied to the *env* and *lab* domains (see rows 2–3 in Table 2). Our experiments on the *med* domain show even more pronounced decrease: e.g. in case of the English–French translation, BLEU drops from 49.12 to 12.32; for English–Greek the change is from 42.24 to 8.96; other translation directions produce similar results. The average decrease for all directions on the *med* domain is 73.33% relative – the domain divergence between the training and test data from this domain is even more pronounced than in the case of the other two domains. The average decrease taken over all translation directions and all the domains is 53.97% relative.

5.2 Measuring domain divergence

From the results presented above, it is evident that the translation quality of a particular test set depends on the extent to which its domain differs from the domain of the training data. Quality is maximal when the domains match and decreases when the test data diverges from the training data. To quantify this observation, we measure cross perplexity of the test data given the training data. For each domain and translation direction, a language model of the same order as the maximum phrase length (7) used in the SMT systems is trained on the source side of the training data and applied to the source side of the test data. The results are presented in Figure 1 (left).

As expected, the perplexity of the general domain test sets is the lowest. It ranges from 40 to 90 depending on the language. In case of the *env* and *lab* domains, perplexity is slightly higher: on the *env* data it ranges from 100 to 190 and on the *lab* data from 80 to 160. Not surprisingly,

test	*dev*	*English–French*		*French–English*		*English–Greek*		*Greek–English*	
env	*gen*	28.03	*0.00*	31.79	*0.00*	20.20	*0.00*	29.23	*0.00*
	env	35.81	*+27.76*	39.04	*+22.81*	26.18	*+29.60*	34.16	*+16.87*
lab	*gen*	22.26	*0.00*	27.00	*0.00*	22.92	*0.00*	31.71	*0.00*
	lab	30.84	*+38.54*	33.52	*+24.15*	28.79	*+25.61*	37.55	*+18.42*
med	*gen*	12.32	*0.00*	15.33	*0.00*	8.96	*0.00*	14.79	*0.00*
	med	18.47	*+49.92*	24.42	*+59.30*	14.57	*+62.61*	18.10	*+22.38*
average			*+38.74*		*+35.42*		*+39.28*		*+19.22*

Table 3: The effect (measured by BLEU) of general-domain (*gen*) and in-domain (*env*, *lab*, *med*) tuning. The figures in italics indicate relative improvement (in percentage) obtained from using in-domain development data for optimization (with respect to tuning on general-domain data).

the perplexity scores obtained on the *med* domain are substantially higher; for most language directions they exceed 700. The only exception is the French–English test set, for which the score is as low as 370. This higher drop is consistent across the other domains (compare the yellow bars with other language pairs in Figure 1, left) and in line with the higher decrease of translation quality for this domain in terms of BLEU (see Section 5.1).

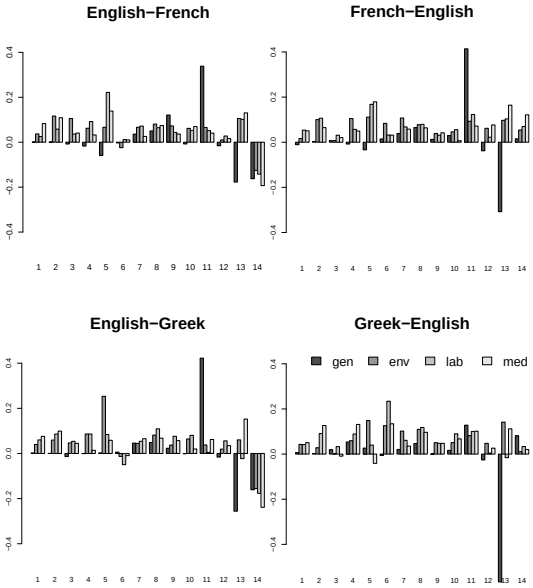
To complete the picture, we directly compare the perplexity scores with the translation quality measured by BLEU and provide a plot in Figure 1 (right). It is quite obvious that the perplexity scores on the logarithmic X axis (PPL) are highly correlated (inversely) with the BLEU scores on the Y axis. Higher perplexity indicates lower translation quality. This finding is in line with previous research on translation confidence estimation (Specia et al., 2011; He et al., 2010).

5.3 Parameter tuning on specific-domain development data

The baseline systems trained and tuned on general-domain data perform much worse on specific domains. Pecina et al. (2011) reported that a surprisingly significant amount of loss can be recovered by tuning on in-domain development data. The average relative improvement measured on the *env* and *lab* domains reported in this work was 25.5%. Our results, including those on the *med* domain, confirm the previous findings (see Table 3). The average relative improvement of BLEU e.g. in English–French translation is 38.74%. Similar improvements are obtained on French–English and English–Greek. Slightly lower improvements were achieved on Greek–English, 19.22% on average. The overall average increase of BLEU is 33.16% relative. Given that the development sets contain only several hundred sentence pairs each, such improvement is remarkable.

5.4 Analysis of model parameters

The only component that changes when the system is tuned on in-domain data are the weights of the feature functions in the log-linear model optimized by MERT. The reordering, language, and translation models all remain untouched (trained on general-domain data). Recall that the parameter space searched through by MERT is large and the error surface highly non-convex, therefore the resulting weight vectors might not be globally optimal and there might be other (i.e. different) weight vectors which perform equally well or even better. For this reason, the actual parameter values are not usually investigated. However, our experiments (Figure 2, left) show that the parameter values and their changes observed when switching from general--domain to specific-domain tuning are in fact highly consistent, indicating interesting trends.



EN–FR	*gen*	*env*	*lab*	*med*
gen	–	0.25	0.12	0.15
env	0.25	–	0.82	0.89
lab	0.12	0.82	–	0.89
med	0.15	0.89	0.89	–

FR–EN	*gen*	*env*	*lab*	*med*
gen	–	0.10	0.15	−0.12
env	0.10	–	0.92	0.86
lab	0.15	0.92	–	0.92
med	−0.12	0.86	0.92	–

EN–EL	*gen*	*env*	*lab*	*med*
gen	–	0.17	0.28	0.18
env	0.17	–	0.81	0.73
lab	0.28	0.81	–	0.76
med	0.18	0.73	0.76	–

EL–EN	*gen*	*env*	*lab*	*med*
gen	–	−0.29	0.19	−0.20
env	−0.29	–	0.72	0.71
lab	0.19	0.72	–	0.81
med	−0.20	0.71	0.81	–

Figure 2: Visualization of model weights of the four systems in the twelve evaluation scenarios; the black bars refer to model weights of the systems tuned on general-domain (*gen*) development sets while the grey bars refer to the model weights of the systems tuned on specific-domain development sets (*env*, *lab*, *med*) (left). Cosine similarity of the system feature vectors (right).

First, we analyse parameters of the systems tuned on the general-domain data (black bars):

1. The high weights assigned to h_{11} (*direct phrase translation probability*) indicate that the phrase pairs in the systems' translation tables apply well to the development data which are from the same domain as the training data; a high reward is given to translation hypotheses consisting of phrases with high translation probability (i.e. good general--domain translations).
2. The low negative weights assigned to h_{13} (*phrase penalty*) imply that the systems prefer hypotheses consisting of fewer but longer phrases.
3. Reordering in the hypotheses is not rewarded (weights of the reordering models h_1–h_7 are assigned values around zero). In some cases (e.g. for English–French and French–English), reordering is even slightly penalized (some weights of h_1–h_7 are negative).
4. The weight of h_{14} (*word penalty*) is negative for translations from English and slightly positive for translations to English. This reflects the fact that translation from English prefers shorter hypotheses and translation to English prefers longer hypotheses.

test	*dev*	*EN–FR*	*FR–EN*	*EN–EL*	*EL–EN*	*Average*
gen	*gen*	4.37	3.46	3.76	2.35	3.49
env	*gen*	3.00	2.49	2.69	2.18	2.59
	def	2.33	2.12	2.12	2.03	2.15
	env	2.16	1.77	2.17	1.54	1.91
lab	*gen*	2.82	2.45	2.97	2.43	2.67
	def	2.24	2.09	2.30	2.21	2.21
	lab	2.05	1.83	2.46	2.30	2.16
med	*gen*	2.00	1.71	1.74	1.43	1.72
	def	1.62	1.52	1.47	1.41	1.51
	med	1.54	1.20	1.38	1.21	1.33

Table 4: Average phrase lengths in translations of all test sets (in all directions) by systems tuned on general (*gen*) and specific domains (*env*, *lab*, *med*) and with the default weights (*def*).

Now, we compare these findings with the systems tuned on the specific domains (grey bars).

1. The weights of h_{11} (*direct phrase translation probability*) decrease rapidly, in some scenarios this weight is very close to zero. The translation tables do not provide enough good quality translations for the specific domains and the best translations of the development sentences consist of phrases with varying translation probabilities.
2. Hypotheses consisting of few (and long) phrases are not rewarded anymore (weights of h_{13} are higher); in most cases they are penalized and hypotheses consisting of more (and short) phrases are allowed or even preferred.
3. In almost all cases the reordering feature weights (features h_1–h_7) increased substantially and for specific-domain data the model significantly prefers hypotheses with altered word order (which is consistent with the two preceding observations).
4. Language model weights (h_8) do not change substantially, its importance remains similar on general-domain and specific-domain data.

These findings are highly consistent across domains and language pairs. The weight vectors of the systems tuned on specific-domain data are quite similar but differ substantially from the parameters obtained by tuning on general-domain. This observation can be quantified by measuring cosine similarity (see Figure 2, right) as proposed by Hopkins and May (2011). Lower scores, as in the first rows/columns of each table, indicate low similarity of the vectors – specific-domain tuned weights differ a lot from the general-domain tuned ones; and vice versa – specific-domain tuned parameters are quite similar when compared to each other.

5.5 Analysis of phrase-length distribution

From the analysis presented above, we conclude that a PB-SMT system tuned on data from the same domain as the training data strongly prefers to construct translations consisting of long phrases. Such phrases are usually of good translation quality (local mistakes of word alignment disappear), fluent (formed by consecutive sequences of words), and recurrent (frequent in data from the same domain); therefore they form good translations of the input sentences and are preferred during decoding. This is, of course, a positive behaviour when the system translates sentences from the same domain. However, if this is not the case and the input sentences contain no or very few longer phrases from the translation tables, the system is not able to construct good translations from shorter phrases.

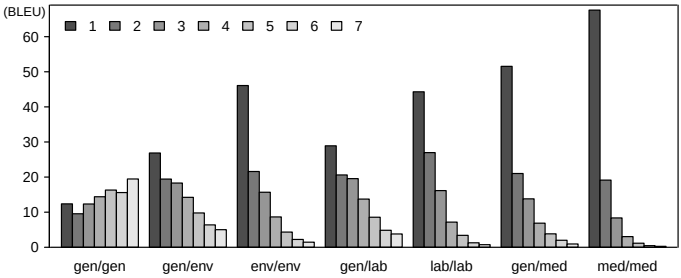


Figure 3: Phrase-length distribution in English–French translations by systems tuned and tested (*dev*/*test*) on various combinations of general (*gen*) and specific (*env*, *lab*, *med*) domains.

To support this hypothesis we analyse the phrase length distribution actually seen in the translation of the test sets. The average phrase lengths estimated for various combinations of tuning and test domains and all language pairs are shown in Table 4. The highest values are observed for translations of general-domain test sets by systems tuned on the same domain: 3.49 on average across all language pairs. The scores for systems trained on general and tuned and tested on specific-domain data are significantly lower and range from 1.21 to 3.00, depending on the domain and language pair. Figure 3 presents complete phrase-length distribution in English–French translations by systems tuned and tested on various combinations of general and specific domains. Generally, a higher divergence of the test domain from the training domain leads to shorter phrases being used in translation. However, when the systems tuned on general-domain are applied to specific domains, the average phrase lengths are consistently longer than for specific-domain tuning. The systems are tuned to prefer long phrases (Table 4) but the translation quality is lower (Table 3). This situation can be interpreted as overtraining, the model overfits the training (and tuning) data and on different data fails to form the best possible translations (given the translation, reordering, and language models).

5.6 Overfitting reduction

The optimal solution in case of such overfitting is to employ a sufficient amount of specific--domain development data, effectively tuning the system to using shorter phrases (see Figure 3). However, if such tuning data is not available (which is quite a realistic scenario in many applications) we explore the following alternatives: simply side-step parameter tuning (no tuning at all), or tune on a different domain, or use smaller amounts of development data, or reduce the maximum phrase length in decoding. All these methods work surprisingly well and are discussed in the following subsections.

5.6.1 No parameter tuning

Essentially, there are two options how to set the weight vectors without tuning. Either we can use the default weights set by Moses ($h_{1,\ldots,7} = 0.3$, $h_8 = 0.5$, $h_{9,\ldots,13} = 0.2$, $h_{14} = -1$) or a flat vector ($h_{1,\ldots,14} = 1$). We explored both options and the results are given in Table 5 (see the rows denoted *def* and *flat*, respectively, in the development data column). In all scenarios, both options outperform the systems trained and tuned on general-domain data. In some cases (e.g. English–Greek translations in all the specific domains), the results are very close

test	*dev*	*English–French*		*French–English*		*English–Greek*		*Greek–English*	
env	*gen*	28.03	*0.00*	31.79	*0.00*	20.20	*0.00*	29.23	*0.00*
	env	35.81	*+27.76*	**39.04**	***+22.81***	**26.18**	***+29.60***	**34.16**	***+16.87***
	lab	**36.16**	***+29.00***	38.78	*+21.99*	**26.13**	***+29.36***	33.85	*+15.81*
	med	32.40	*+15.59*	36.89	*+16.04*	24.89	*+23.22*	**34.01**	***+16.35***
	def	34.94	*+24.65*	34.05	*+7.11*	**26.09**	***+29.16***	31.33	*+7.18*
	flat	32.22	*+14.95*	37.66	*+18.46*	21.91	*+8.47*	32.84	*+12.35*
lab	*gen*	22.26	*0.00*	27.00	*0.00*	22.92	*0.00*	31.71	*0.00*
	env	30.13	*+35.35*	**33.21**	***+23.00***	28.36	*+23.73*	**37.57**	***+18.48***
	lab	**30.84**	***+38.54***	**33.52**	***+24.15***	**28.79**	***+25.61***	**37.55**	***+18.42***
	med	27.04	*+21.47*	30.77	*+13.96*	26.85	*+17.15*	**37.52**	***+18.32***
	def	29.26	*+31.45*	29.73	*+10.11*	28.48	*+24.26*	34.95	*+10.22*
	flat	27.16	*+22.01*	32.24	*+19.41*	25.13	*+9.64*	35.79	*+12.87*
med	*gen*	12.32	*0.00*	15.33	*0.00*	8.96	*0.00*	14.79	*0.00*
	env	**18.74**	***+52.11***	23.75	*+54.92*	13.89	*+55.02*	**17.88**	***+20.89***
	lab	**18.91**	***+53.49***	23.73	*+54.79*	13.69	*+52.79*	17.62	*+19.13*
	med	18.47	*+49.92*	**24.42**	***+59.30***	**14.57**	***+62.61***	**18.10**	***+22.38***
	def	18.20	+47.73	21.15	*+37.96*	13.82	*+54.24*	16.70	*+12.91*
	flat	17.06	*+38.47*	23.02	*+50.16*	11.99	*+33.82*	17.71	*+19.74*

Table 5: Translation quality (in BLEU) of the general-domain systems tuned and tested on various domains. The figures in italics indicate relative improvement (in percentage) over the system tuned on general domain. The figures in bold denote the best performing combination for each test domain and translation direction and those which are not significantly different (Koehn, 2004, $p = 0.05$).

to those of systems tuned on specific-domain data. The overall average relative improvement of the systems with default parameters over the systems tuned on general domain is 24.75% (compare with 33.16% obtained from specific-domain tuning). The average phrase length in translations produced by such systems falls between the scores of general-domain-tuned and specific-domain-tuned systems (see rows with *def* in the development data column in Table 4). The systems with the flat weight vectors achieve an average relative improvement of 21.70%. However, they outperform the systems with the default parameters always when the translation direction is to English; the systems with the default parameters are better when translating from English.

5.6.2 Cross-domain tuning

It seems that the problem of the overfitted general-domain models and their poor performance on specific domains can be reduced by "diverting" the systems away from the general domain they are tuned to translate – but not necessarily towards a particular specific domain. To analyse this hypothesis we perform "cross-domain" tuning, i.e. tuning on specific domains different from the test domains. The results are shown in Table 5 (see the rows where the test and development domain do not match). In all scenarios the cross-domain tuned system performs better than the un-tuned ones. In a few cases the systems tuned on a cross domain perform even better than the in-domain ones: e.g. the EN–FR system tuned on the *lab* domain and tested on the *env* and *med* domains or the EL–EN system tuned on the *env* domain and tested on the *lab* domain, however, in most such cases the improvement is not statistically significant. The overall average relative gain over the systems tuned on general domain is 27.62% (compare with 24.75% obtained from no tuning and 33.16% from in-domain tuning).

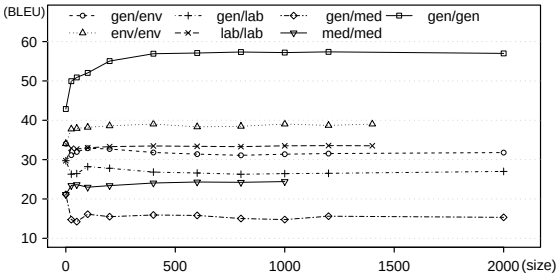


Figure 4: Translation quality (BLEU) of French–English systems tuned on data of varying size.

Similar results were observed also on mixtures of two domains (e.g. tuned on *lab+env* and tested on *med*). In general, we can conclude that cross-domain tuning is a reasonable solution when no in-domain development data is available (and the domains differ in a similar way).

5.6.3 Tuning on small development data

In the previous two scenarios we did not use any specific-domain development data for tuning, but were able to get very close to the performance of the systems tuned on a specific domain. Specific-domain parallel data is scarce, for many domains not available at all and must be prepared by manual translation of monolingual in-domain sentences. We investigate how much development data is needed. The only technical requirement is that MERT, the parameter optimization method, must converge in a reasonable number of iterations. For this reason, typical development sets contain about 1,000 – 2,000 sentence pairs (compare e.g. the size of development sets provided for the WMT[4] translation shared tasks). We vary the amount of sentences in our development sets, tune the systems, test their performance on the test sets and plot learning curves to capture the dependency of translation quality (in terms of BLEU) against gradually increasing the size of development data.

The general shapes of the curves are consistent across all translations (and domains) and thus we provide the curves for the English–French translation direction only (see Figure 4). Increasing the size of development sets is beneficial only in case the domains of development and test data are the same. The curve of the system tuned and tested on the general domain reaches a plateau for about 500 sentence pairs. In case of in-domain tuning for specific domains, the plateau is reached much earlier. Usually, as few as 100–200 sentence pairs are enough to get optimal results. This is encouraging, as tuning on specific-domains yields best results and fortunately requires only very limited amounts of bilingual data (and expense). Development sets of more than 400–600 sentences pairs do not improve translation quality at all and make the tuning process take longer. The systems tuned on the general domain and tested on specific domain do not benefit from the development data at all. The relatively high BLEU scores achieved with no tuning (zero development data size) decrease with increasing size of the development sets.

[4] http://www.statmt.org/wmt12/

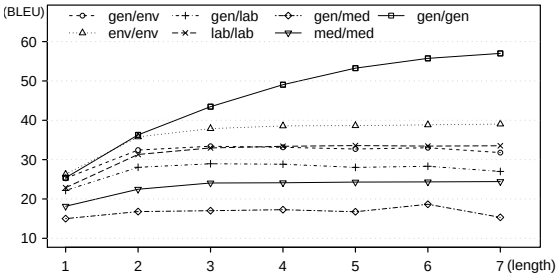


Figure 5: Translation quality (BLEU) of French–English systems with varying max phrase length.

5.6.4 Limiting phrase length

In the last experiment presented in this paper we limit the maximum phrase length allowed during training and decoding and study how system performance changes. The systems tuned on general-domain prefer longer phrases which, however, do not occur frequently in the specific--domain test sets. Our baseline systems, trained and tuned on general domain with maximum phrase length set to seven, translate general-domain test sets with an average phrase length of 3.49 (see Table 4). However, for the systems tuned and tested on in-domain data, this score is as low as 1.80. Figure 5 illustrates how the translation quality changes when the maximum phrase length varies from one to seven. The only case when longer phrases improve translation quality is for the systems trained, tuned and tested on the same (general) domain. In all other cases, the results for phrases up to three words long are as good as for longer phrases. If the domain of the test data does not match the domain of training and tuning data, the maximum phrase length set to three is enough in all scenarios. Longer phrases lead to degradation of translation quality and increase time for training and decoding, as well as memory requirements for building and storing the translation models. A similar result was reported already by Koehn et al. (2003). They observed that limiting the maximum length of a phrase to only three words achieved top performance. However, current state-of-the-art SMT systems usually benefit from longer phrases than three (see e.g. the top curve in Figure 5 which refers to a general-domain system applied to a general-domain test set), and our result applies only to scenarios where the training and test domains do not match; in that case setting the maximum phrase length to three is sufficient.

6 Conclusions

In this work, we have analysed domain adaptation of PB-SMT by tuning parameters of the underlying log-linear model. We confirmed the observation from previous research that systems trained and tuned on general domain perform poorly on specific domains. This finding is not very surprising, but the amount of loss and the fact that it is observed consistently in many evaluation scenarios was unexpected. We found that perplexity of the source side of the test data given the source side of the training nicely correlates with the translation quality.

Further, we confirmed that tuning the systems trained on general domain on specific target domain data recovers a (often) spectacular amount of the loss. We carried out a detailed analysis of the model parameters and phrase length distribution in translations of the test data and found that a system trained and tuned on general domain strongly prefers long and few

phrases in the output translations and therefore underperforms on specific domains where such phrases do not occur frequently. By contrast, the same systems tuned on specific-domain data form output translations from shorter phrases, allow more reordering and perform significantly and consistently better on specific domain data.

We investigated possible solutions for (common) scenarios when no or very little in-domain data is available for parameter tuning. Skipping tuning, i.e. using the default model parameters, performs surprisingly well and always outperforms systems tuned on general domain. Based on this observation, this should be preferred over general domain tuning if the test domain differs substantially. Cross-domain tuning on a different set also offers a good solution when no in-domain development data is available, especially when the domains differ in a similar way (e.g. measured by perplexity). This step has the effect of disassembling the original general-domain system towards shorter phrases and it does not matter much which different development set to use.

The analysis of learning curves of the tuning process showed that in-domain tuning of the general-domain systems requires about 100–200 sentence pairs to achieve decent translation quality (in terms of BLEU, the gain obtained from tuning on more data was negligible). We also experimented with limiting the maximum phrase length of decoding. The results showed that setting this parameter to three is sufficient for translating data from specific domains; longer phrases in this case do not improve translation quality and increase computational requirements of the translation systems. The last two results (limiting phrase length and using sufficient amounts of development data) have efficiency implications of paramount importance in industrial application scenarios.

Acknowledgments

This research was supported by the the Czech Science Foundation (grant no. P103/12/G084), by EU FP7 projects PANACEA (contract no. 248064) and Khresmoi (contract no. 257528), and by Science Foundation Ireland (grant no. 07/CE/I1142) as part of the Centre for Next Generation Localisation (http://www.cngl.ie/) at Dublin City University.

References

Banerjee, P., Du, J., Li, B., Naskar, S., Way, A., and van Genabith, J. (2010). Combining Multi-Domain Statistical Machine Translation Models using Automatic Classifiers. In *9th Conference of the Association for MT in Americas*, pages 141–150, Denver, Colorado, USA.

Bertoldi, N., Haddow, B., and Fouet, J.-B. (2009). Improved minimum error rate training in moses. *Prague Bulletin of Mathematical Linguistics*, No. 91:7–16.

Callison-Burch, C., Osborne, M., and Koehn, P. (2006). Re-evaluating the role of bleu in machine translation research. In *11th Conference of the European Chapter of the Association for Computational Linguistics*, pages 249–256, Trento, Italy.

Daumé III, H. and Jagarlamudi, J. (2011). Domain adaptation for machine translation by mining unseen words. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics and Human Language Technologies, Short Papers*, pages 407–412, Portland, Oregon, USA.

Eck, M., Vogel, S., and Waibel, A. (2004). Language Model Adaptation for Statistical Machine Translation based on Information Retrieval. In *Proceedings of the International Conference on Language Resources and Evaluation*, Lisbon, Portugal.

Finch, A. and Sumita, E. (2008). Dynamic model interpolation for statistical machine translation. In *Proceedings of the Third Workshop on Statistical Machine Translation*, pages 208–215, Columbus, Ohio, USA.

Foster, G., Goutte, C., and Kuhn, R. (2010). Discriminative instance weighting for domain adaptation in statistical machine translation. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 451–459, Cambridge, MA.

He, Y., Ma, Y., Roturier, J., Way, A., and van Genabith, J. (2010). Improving the Post-Editing Experience Using Translation Recommendation: A User Study. In *Proceedings of the Ninth Conference of the Association for Machine Translation in the Americas*, pages 247–256, Denver, Colorado, USA.

Hildebrand, A. S., Eck, M., Vogel, S., and Waibel, A. (2005). Adaptation of the Translation Model for Statistical Machine Translation based on Information Retrieval. In *Proceedings of the 10th Annual Conference of the European Association for Machine Translation*, pages 133–142, Budapest, Hungary.

Hopkins, M. and May, J. (2011). Tuning as ranking. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 1352–1362, Edinburgh, United Kingdom.

Koehn, P. (2004). Statistical significance tests for machine translation evaluation. In Lin, D. and Wu, D., editors, *Proceedings of EMNLP 2004*, pages 388–395, Barcelona, Spain.

Koehn, P. (2005). Europarl: A Parallel Corpus for Statistical Machine Translation. In *Conference Proceedings of the Tenth Machine Translation Summit*, pages 79–86, Phuket, Thailand.

Koehn, P., Hoang, H., Birch, A., Callison-Burch, C., Federico, M., Bertoldi, N., Cowan, B., Shen, W., Moran, C., Zens, R., Dyer, C., Bojar, O., Constantin, A., and Herbst, E. (2007). Moses: open source toolkit for statistical machine translation. In *Proceedings of the 45th Annual Meeting of the ACL on Interactive Poster and Demonstration Sessions*, pages 177–180, Prague, Czech Republic.

Koehn, P., Och, F. J., and Marcu, D. (2003). Statistical phrase-based translation. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology - Volume 1*, pages 48–54, Edmonton, Canada.

Koehn, P. and Schroeder, J. (2007). Experiments in domain adaptation for statistical machine translation. In *Proceedings of the Second Workshop on Statistical Machine Translation*, pages 224–227, Prague, Czech Republic.

Langlais, P. (2002). Improving a general-purpose Statistical Translation Engine by terminological lexicons. In *COLING-02 on COMPUTERM 2002: second international workshop on computational terminology - Volume 14*, pages 1–7, Taipei, Taiwan.

Munteanu, D. S. and Marcu, D. (2005). Improving Machine Translation Performance by Exploiting Non-Parallel Corpora. *Computational Linguistics*, 31:477–504.

Nakov, P. (2008). Improving English-Spanish statistical machine translation: experiments in domain adaptation, sentence paraphrasing, tokenization, and recasing. In *Proceedings of the Third Workshop on Statistical Machine Translation*, pages 147–150, Columbus, Ohio, USA.

Och, F. J. (2003). Minimum error rate training in statistical machine translation. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics*, pages 160–167, Sapporo, Japan.

Papineni, K., Roukos, S., Ward, T., and Zhu, W.-J. (2002). BLEU: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, USA.

Pecina, P., Toral, A., Papavassiliou, V., Prokopidis, P., and van Genabith, J. (2012). Domain adaptation of statistical machine translation using web-crawled resources: a case study. In Cettolo, M., Federico, M., Specia, L., and Way, A., editors, *EAMT 2012: Proceedings of the 16th Annual Conference of the European Association for Machine Translation*, pages 145–152, Trento, Italy.

Pecina, P., Toral, A., Way, A., Papavassiliou, V., Prokopidis, P., and Giagkou, M. (2011). Towards Using Web-Crawled Data for Domain Adaptation in Statistical Machine Translation. In *Proceedings of the 15th Annual Conference of the European Associtation for Machine Translation*, pages 297–304, Leuven, Belgium.

Specia, L., Hajlaoui, N., Hallett, C., and Aziz, W. (2011). Predicting machine translation adequacy. In *Proceedings of the Machine Translation Summit XIII*, Xiamen, China.

Stolcke, A. (2002). SRILM-an extensible language modeling toolkit. In *Proceedings of International Conference on Spoken Language Processing*, pages 257–286, Denver, Colorado, USA.

Tiedemann, J. (2009). News from OPUS – A Collection of Multilingual Parallel Corpora with Tools and Interfaces. In Nicolov, N., Angelova, G., and Mitkov, R., editors, *Recent Advances in Natural Language Processing V*, volume 309 of *Current Issues in Linguistic Theory*, pages 227–248. John Benjamins, Amsterdam & Philadelphia.

Wu, H., Wang, H., and Zong, C. (2008). Domain adaptation for statistical machine translation with domain dictionary and monolingual corpora. In *Proceedings of the 22nd International Conference on Computational Linguistics - Volume 1*, pages 993–1000.

A supervised aggregation framework for multi-document summarization

Yulong Pei Wenpeng Yin Qifeng Fan Lian'en Huang
The Shenzhen Key Lab for Cloud Computing Technology & Applications (SPCCTA)
Shenzhen Graduate School
Peking University, Shenzhen 518055, P.R. China
{paul.yulong.pei,mr.yinwenpeng,fanqf1026}@gmail.com, hle@net.pku.edu.cn

ABSTRACT
In most summarization approaches, sentence ranking plays a vital role. Most previous work explored different features and combined them into unified ranking methods. However, it would be imprecise to rank sentences from a single point of view because contributions from the features are onefold in these methods. In this paper, a novel supervised aggregation approach for summarization is proposed which combines different summarization methods including LexPageRank, LexHITS, manifold-ranking method and DivRank. Human labeled data are used to train an optimization model which combines these multiple summarizers and then the weights assigned to each individual summarizer are learned. Experiments are conducted on DUC2004 data set and the results demonstrate the effectiveness of the supervised aggregation method compared with typical ensemble approaches. In addition, we also investigate the influence of training data construction and component diversity on the summarization results.

KEYWORDS: Multi-document summarization, supervised aggregation framework.

Proceedings of COLING 2012: Technical Papers, pages 2225–2242,
COLING 2012, Mumbai, December 2012.

1 Introduction

Multi-document summarization aims to generate a compressed summary by extracting the major information from a collection of documents sharing the same or similar topics. With the massive explosion of information on the web, e.g., news, blogs and microblogs, multi-document summarization, as an effective solution for information explosion, provides improved mechanisms for understanding documents and reducing information overload. Therefore it has attracted considerable attention recently.

Generally speaking, summarization can be categorized into two types: extractive summarization and abstractive summarization. Extractive summarization generates summary directly by choosing sentences from original documents while abstractive summarization requires formulating new sentences according to the text content. Although abstractive summarization could be more concise and understandable, it usually involves heavy machinery from natural language processing (Hahn and Mani, 2000). In this paper, we mainly focus on extractive multi-document summarization.

In extractive summarization tasks, sentence ranking is the issue of most concern (Wei et al., 2009). A number of methods have been proposed in the literature from different aspects to rank sentences. Feature-based approaches rank sentences by exploring combination of different features of sentences such as term frequency, sentence position, length and etc. Graph-based ranking approaches aim to design different strategies to rank sentences using random walk model to capture relations between sentences, i.e., LexPageRank (Erkan and Radev, 2004), TextRank (Mihalcea and Tarau, 2004) and DivRank (Mei et al., 2010). Nowadays, some machine learning algorithms have also been applied in summarization for learning optimal feature weights automatically, for instance, some leaning to rank models (Svore et al., 2007; Jin et al., 2010; Shen and Li, 2011) have been introduced in summarization tasks.

Most previous work concentrated on exploring different features, and the features were combined in unified ranking strategies for summarization. However, to identify the importance of sentences from a single point of view would be difficult (Wong et al., 2008) because contributions from the features are onefold in these methods. To address this problem, it is natural to propose an ensemble approach which combines different summarization methods to rank the sentences. Ensemble methods have been used in a variety of applications including web search, spam detection and collaborative filtering. Specific to ranking problems, ensemble ranking or ranking aggregation methods have been widely studied in many different tasks especially in information retrieval. However, there are limited work attempts on applying ensemble methods to summarization. In (Wang and Li, 2010) and (Wang and Li, 2011), a weighted consensus method was proposed to aggregate multiple summarization methods. Although the ensemble method used in these work outperforms individual summarizers and some other combination methods, there exists a serious drawback in unsupervised methods: because the assignment of weights to different summarizers is based on the consensus, contribution from some summarizer containing inferior ranking results may lead to an inaccurate final result.

In order to deal with the drawbacks in unified ranking strategies and unsupervised aggregation methods, we propose a supervised aggregation framework for summarization in this study. Taking a summarization task as a ranking problem, we combine several different summarizers, learn the weights assigned to each summarizer with human labeled data and then rank sentences according to their combined scores. This aggregation approach generates promising results by aggregating several different summarization methods. Experiments on DUC2004

data set have been conducted and the results demonstrate the effectiveness of the proposed supervised summarization aggregation method which outperforms typical ensemble schemes under various evaluation metrics. In addition, the influence of schemes to construct training data and component diversity on the summarization results has also been investigated.

The rest of this article is organized as follows. We briefly review the related work in Section 2 and the supervised aggregation framework for summarization is introduced in Section 3. We present the supervised aggregation summarization method implementation in Section 4 and experiments are discussed in Section 5. Finally, we draw a conclusion of this study.

2 Related work

The related work will be introduced from two aspects, first we describe some representative summarization methods and then the typical work about rank aggregation are presented briefly.

2.1 Multi-document summarization

Multi-document summarization is a process to generate a summary by reducing documents in size while retaining the main characteristics of the original documents. In order to archive this goal, different features and ranking strategies have been studied.

Traditional feature-based ranking methods explored different features of sentences to score and rank the sentences. One of the most popular feature-based methods is centroid-based method (Radev et al., 2004). Radev et al. implemented MEAD as a cetroid-based summarizer by combining several predefined features including TF*IDF, cluster centroid and position to score the sentences. Lin and Hovy (Lin and Hovy, 2002) used term frequency, sentence position, stigma words and simplified Maximal Marginal Relevance (MMR) to build the NeATS multi-document summarization system. High frequent words were proved crucial in reflecting the focus of documents (Nenkova et al., 2006) and You Ouyang et al. studied the influence of different word positions in summarization (Ouyang et al., 2010).

Graph-based ranking algorithms nowadays are successfully applied in summarization and LexPageRank (Erkan and Radev, 2004) is the representative work which is based on the PageRank algorithm (Page et al., 1999). Graph-based ranking algorithms take global information into consideration rather than rely only on vertex-specific information, therefore have been proved successful in multi-document summarization. Some methods have extended the traditional graph-based models recently including multi-layer graph incorporated with different relationship (Wan and Yang, 2008), multi-modality graph based on the manifold-ranking method (Wan and Xiao, 2009) and DivRank (Mei et al., 2010) introducing the time-variant matrix into a reinforced random walk to balance prestige and diversity.

Topic model has also been exploited in summarization recently. The query Latent Dirichlet Allocation (qLDA) model was proposed in (Tang et al., 2009), and this model takes into account the query information to extract query-oriented summaries. HIRESUM model (Haghighi and Vanderwende, 2009) was presented based on hierarchical Latent Dirichlet Allocation (hLDA) to represent content specificity as a hierarchy of topic vocabulary distributions. Celikyilmaz and Hakkani-Tur also utilized a hLDA-style model to devise a sentence-level probabilistic topic model and a hybrid learning algorithm for extracting salient features of sentences to generate summaries (Celikyilmaz and Hakkani-Tur, 2010).

To date, various machine learning methods, including unsupervised and supervised methods, have been developed for extractive summarization by learning to summarize documents automatically. For instance, Shen et al. proposed a conditional random field (CRF) based method which treats summarization task as a sequence labeling problem (Shen et al., 2007). The structural SVM approach was explored in (Li et al., 2009) to enhance diversity, coverage and balance of summary simultaneously. Learning to rank (Li, 2011) methods that are widely studied in information retrieval community have been applied in summarization (Svore et al., 2007; Jin et al., 2010; Shen and Li, 2011). These studies have used different learning strategies to rank sentences for summarizing documents.

In order to identify the importance of sentences from multiple aspects, aggregation methods can be used in summarization to combine results from different summarizers. However, aggregation methods for summarization are seldom been discussed in previous work. An exception is proposed in (Wang and Li, 2011), in this study a weighted consensus summarization based on optimization was applied in summarization by aggregating four different summarization methods.

2.2 Rank aggregation

Rank aggregation is aimed at combining results of objects from multiple ranking functions to generate a better one and it has been applied into a variety of applications including information retrieval and collaborative filtering. In general, rank aggregation can be categorized into two types: order-based and score-based (Liu et al., 2007). Order-based aggregation method takes order information as input from individual rankers and score-based method utilizes ranking scores from component rankers.

In most existing unsupervised rank aggregation methods, the final ranking decisions depend on majority voting. Median rank aggregation (Van and Erp, 2000) sorts entities based on the medians of their ranks in all the ranking lists. To treat different ranking lists with different weights, Klementiev, Roth and Small proposed an unsupervised aggregation algorithm named ULARA (Klementiev et al., 2007) to learn the weights of ranking lists online by optimizing the weighted Borda count. However, as mentioned in the Introduction, a serious drawback exists in unsupervised aggregation methods, that some inferior rankers may influence the overall performance.

In order to improve the quality of ranking aggregation, some supervised learning methods have also been proposed. The work in (Liu et al., 2007) incorporated labeled data into a supervised rank aggregation method to minimize disagreements between ranking results and labeled data. (Chen et al., 2011) proposed a semi-supervised rank aggregation approach and the work minimizes the weight disagreements of different rankers to learn the aggregation function. In the semi-supervised case, the preference constraints on several item pairs were incorporated and the intrinsic manifold structures of items are also taken into account. In (Hoi and Jin, 2008), a different semi-supervised method was proposed, which learns query-dependent weights by exploring the underlying distribution of items to be ranked and assigns two similar retrieved items with similar ranking scores.

Since ranking sentences plays an important role in summarization tasks, we can regard each result from an individual system as a ranking of sentences (Wang and Li, 2011). Then, aggregation methods can also be applied to summarization to combine multiple ranking results into an aggregation ranking to generate the combined results. However, aggregation meth-

ods for summarization are seldom been discussed in the literature. (Wang and Li, 2010) and (Wang and Li, 2011) are attempts and, in their work, a weighted consensus method was proposed for summarization and an unsupervised iteration method was applied to solve the optimization problem. Different from the unsupervised aggregation method used in these work, this paper proposes a supervised aggregation framework for summarization which combines different summarization methods and learns weights automatically with human labeled data.

3 Supervised summarization aggregation framework

In this section, we first state the problem by describing the general framework of aggregation method for summarization and then the proposed supervised summarization method is introduced in details.

3.1 Problem statement

First we consider the general framework of summarization aggregation that combines results from multiple summarizers, and each summarizer can produce a score list for sentences of a document cluster. An illustration of the framework is given by Figure 1. Suppose we have M document clusters $\{c_1, c_2, ..., c_M\}$ in the training data and the ith document cluster contains N_i sentences $\{s_1^i, s_2^i, ..., s_{N_i}^i\}$. Each sentence s_j^i is associated with a label l_j^i to denote whether it will be chosen as a summary sentence and the labels are categorized into three types as follows.

$$l_j^i = \begin{cases} +1 & \text{summary;} \\ 0 & \text{possible summary;} \\ -1 & \text{non-summary.} \end{cases} \quad (1)$$

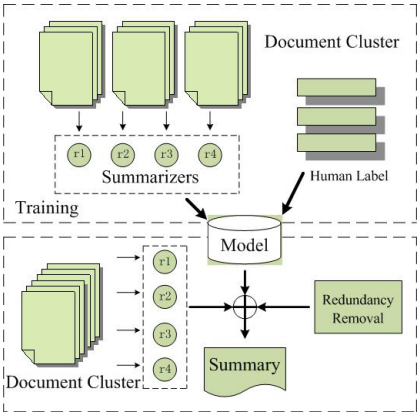


Figure 1: The framework of supervised aggregation for summarization.

Let $\mathscr{R} = \{r_1(\cdot), r_2(\cdot), ..., r_K(\cdot)\}$ denotes the set of K summarization methods and each method can produce a score list. The task of summarization aggregation is to combine the score lists

given by methods in $\mathscr{R}$ to produce better ranking results than any individual summarizer. In this study, we express the aggregation in a linear combining method and the combined ranking function can be denoted as

$$f(s) = \sum_{k=1}^{K} w_k r_k(s), \tag{2}$$

where w_k is the weight assigned to the kth individual summarization method. Thus, to learn the combination weights w_k $(k = 1, 2, ..., K)$ is pivotal in the aggregation method.

For simplicity, the aggregation score of sentence s_j^i can be rewritten in a matrix form:

$$f(s_j^i) = \mathbf{x}_j^i \mathbf{w}, \tag{3}$$

where $\mathbf{w} = [w_1, w_2, ...w_K]^T$ is the combination weights vector. $\mathbf{x}_j^i$ is a K-dimensional vector for representing the ranking scores computed by K different summarizers and denoted as

$$\mathbf{x}_j^i = [x_{j,1}^i, x_{j,2}^i, ..., x_{j,K}^i] = [r_1(s_j^i), r_2(s_j^i), ..., r_K(s_j^i)]. \tag{4}$$

3.2 Method description

In the supervised ranking aggregation method, we apply Ranking SVM (Joachims, 2002) directly. Ranking SVM trains the ranking model by decomposing a ranking list into the ordered pairs of items. In the document cluster c_i, given two sentences s_j^i and s_k^i with their score vectors $\mathbf{x}_j^i$ and $\mathbf{x}_k^i$, the training example can be built as the form $(\mathbf{x}_j^i - \mathbf{x}_k^i, z_{jk}^i)$ and the training label z_{jk}^i is defined as:

$$z_{jk}^i = \begin{cases} +1 & \text{if } s_j^i \succ s_k^i; \\ -1 & \text{if } s_k^i \succ s_j^i. \end{cases} \tag{5}$$

where $s_j^i \succ s_k^i$ denotes that sentence s_j^i is ranked higher than s_k^i. By defining the new training examples, the mathematical formulation of Ranking SVM is shown below, where the linear scoring function introduced in Formula (2) is used:

$$\begin{aligned} \min_{w} \quad & \frac{1}{2}\|\mathbf{w}\|^2 + C \sum_{i=1}^{M} \sum_{j=1}^{N_i} \sum_{k=j+1}^{N_i} \xi_{jk}^i \\ s.t. \quad & z_{jk}^i (\mathbf{w} \cdot (\mathbf{x}_j^i - \mathbf{x}_k^i)) \geq 1 - \xi_{jk}^i, i = 1, 2, ..., M, \\ & \xi_{jk}^i \geq 0, j < k \ \ and \ \ j, k \in [1, 2, ..., N_i]. \end{aligned} \tag{6}$$

where function complexity regularizer $\|\mathbf{w}\|^2$ is introduced to guarantee the generalization capacity and ξ_{jk}^i is the slack variables. C is a parameter that allows trading-off margin size against training error. Authors in (Hoi and Jin, 2008) pointed out that the above optimization problem has a drawback in training efficiency because the number of training pairs is quadratic of the number of items. Following their improved approach which can decrease the number of constraints significantly, we first build a relevance matrix A^i for ith document cluster to replace

above training labels, and the elements in the matrix are defined as follows.

$$A^i_{j,k} = \begin{cases} +1 & \text{if } l^i_j = 1 \text{ and } l^i_k = -1; \\ -1 & \text{if } l^i_j = -1 \text{ and } l^i_k = 1; \\ 1/2 & \text{if } l^i_j = 1 \text{ and } l^i_k = 0; \\ -1/2 & \text{if } l^i_j = 0 \text{ and } l^i_k = 1; \\ 0 & \text{otherwise.} \end{cases} \tag{7}$$

where l^i_j represents the label assigned to sentence s_j in the document cluster c_i and is defined in Formula (1).

Furthermore, a ranking matrix R is defined to denote the ranking results generated by the set of summarizers. For document cluster c_i, matrix $R^{i,j}$ stands for the ranking results output by summarizer $r_j(\cdot), j = 1, 2, ..., K$. Specifically, $R^{i,j}_{k,l} = 1$ if sentence s^i_k obtains higher score than s^i_l by $r_j(\cdot)$, and 0 otherwise. Next the matrix is normalized by a column-based normalization to be a transition matrix $\tilde{R}^{i,j}$, i.e. $\sum_{k=1}^{N_i} \tilde{R}^{i,j}_{k,l} = 1$.

After introducing the relevance matrix A and ranking matrix R, the original Ranking SVM shown in Formula (6) can be reformulated as:

$$\begin{aligned} \min_{w} \quad & \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{M}\xi_i \\ s.t. \quad & sim([A^i]^T, [\sum_{j=1}^{K} w_j \tilde{R}^{i,j}]) \geq 1 - \xi_i, , i = 1, 2, ..., M, \\ & \xi^i_{jk} \geq 0, j < k \ \ and \ \ j, k \in [1, 2, ..., N_i]. \end{aligned} \tag{8}$$

where $sim(\cdot,\cdot)$ is a function that measures the similarity between two matrices: the transposition of relevance matrix A^i and the combined ranking matrix $\sum_{j=1}^{K} w_j \tilde{R}^{i,j}$. In the experiments, we follow (Hoi and Jin, 2008) by using trace function to measure the similarity of the two matrices. Compared with Formula (6), the number of constraints is significantly decreased from $\mathcal{O}(MN_i^2)$ to $\mathcal{O}(M)$.

4 Supervised summarization aggregation method implementation

4.1 Construction of training data

In order to apply supervised aggregation approach in summarization, we need to construct the training set in the form $\{(s^i_1, l^i_1), (s^i_2, l^i_2), ..., (s^i_{N_i}, l^i_{N_i})\}$ where s^i_j is the jth sentence in the document cluster c_i and l^i_j is the label assigned to the sentence. To capture the features contained in suboptimal sentences, we label sentences using three categories mentioned in Section 3.1: summary (+1), possible summary (0) and non-summary (-1).

Given a document cluster c which includes N sentences $\{s_1, s_2, ..., s_N\}$ and the corresponding human generated summary set $H = \{H_1, H_2, ..., H_m\}$ (H_i is the human summary generated by the ith linguist), we compute the score $score(s|H)$ for each sentence s in document cluster c to measure whether it can be chosen as the summary sentence. Motivated by ROUGE evaluation methods (Lin and Hovy, 2003), our scoring methods compute the combination of

multiple n-grams (1-gram and 2-gram are used in this study) probabilities of each sentence to be recognized as a summary sentence based on the human summary set.

First we compute the probability of an n-gram t under a human summary H_i as:

$$p(t|H_i) = tf(t)/|H_i|, \tag{9}$$

where $tf(t)$ is the frequency of t in H_i and $|H_i|$ is the number of n-grams (n is corresponding to the length of t) in H_i. In the data set, several human summaries are provided for each document cluster, and both average and maximum schemes introduced in (Ouyang et al., 2007) can be utilized[1]. The average scheme to obtain the probability of t under all human summaries is described as:

$$p_{avg}(t|H) = \sum_{H_i \in H} p(t|H_i)/|H|, \tag{10}$$

where $|H|$ is the number of summaries in the human summary set. And for maximum scheme the computation method is defined as:

$$p_{max}(t|H) = \max_{H_i \in H} p(t|H_i) \tag{11}$$

Motivated by the ROUGE evaluation metrics, we take into account both 1-gram and 2-gram to calculate the final score for each sentence in the following formula.

$$Score(s|H) = \alpha \sum_{t_{1-gram} \in s} p(t_{1-gram}|H) + (1-\alpha) \sum_{t_{2-gram} \in s} p(t_{2-gram}|H), \tag{12}$$

where α is used to control the ratios of these two types of n-grams in computing scores. Since 1-gram based ROUGE score has been shown to agree with human judgment most (Lin and Hovy, 2003), in our experiments α is set empirically to be 0.7.

4.2 Individual summarization methods

To evaluate the proposed supervised aggregation method for summarization, we introduce four typical summarization approaches (i.e., LexPageRank, LexHITS. Manifold-ranking, and DivRank) in the system implementation. In this section, we briefly describe these approaches which all have been proved effective in summarization task.

1. LexPageRank
   LexPageRank (Erkan and Radev, 2004) is a graph-based summarization method by introducing PageRank into summarization, which computes sentence scores by making use of the voting or recommendations between sentences. Sentences are used as nodes in the graph and the computational process can be described as:

$$PR(s_i) = \lambda \cdot \sum_{j:j \neq i} PR(s_j) \cdot w_{ji} + \frac{(1-\lambda)}{|S|}, \tag{13}$$

   where $PR(s_i)$ is the score of sentence s_i and $|S|$ denotes the number of sentences in a document cluster. w_{ji} represents the weight (e.g., cosine similarity) between sentence s_j and s_i and λ is the damping factor to control the probability to walk to a random sentence.

[1]The experimental comparison of two schemes will be described in Section 5.3.2

2. LexHITS
   Similar to LexPageRank, HITS (Kleinberg, 1999) algorithm can be applied in summarization task as well and correspondingly is named LexHITS. In LexHITS, sentences denote both authority nodes and hub nodes and the iteration process is written as follows.

$$\begin{aligned} Auth^{(T+1)}(s_i) &= \sum_{j:j\neq i} w_{ij} \cdot Hub^{(T)}(s_j) \\ Hub^{(T+1)}(s_j) &= \sum_{i:i\neq j} w_{ji} \cdot Auth^{(T)}(s_i) \end{aligned}, \tag{14}$$

   where $Auth^{(T)}(s_i)$ and $Hub^{(T)}(s_i)$ represent the authority score and hub score of sentence s_i at the Tth iteration, respectively. w_{ij} denotes the weight between sentence s_i and s_j same as Formula (13). Then sentences are ranked according to their authority scores.

3. Manifold
   Manifold-ranking based method (Zhou et al., 2004) is a universal ranking algorithm and can capture the underlying manifold structure of data. Authors in (Wan et al., 2007) used manifold-ranking approach for summarization. First the similarity matrix W is built and the element w_{ij} denotes the weight between sentence s_i and s_j like the settings in LexPageRank and LexHITS. Then normalize W by $\tilde{W} = D^{-1/2}WD^{-1/2}$ in which D is the diagonal matrix with (i,i)-element equal to the sum of the ith row of W. By incorporating a n-dimensional vector y, the score of each sentence can be iterated as:

$$MF^{(T+1)}(s_i) = \alpha \sum_{j:j\neq i} \tilde{w}_{ij} MF^{(T)}(s_j) + (1-\alpha)y_i, \tag{15}$$

   where $MF^{(T)}(s_i)$ denotes the manifold-ranking score of sentence s_i at the Tth iteration and $\tilde{w}_{ij}$ is the element in the normalized matrix $\tilde{W}$. α is a parameter between 0 and 1.

4. DivRank
   DivRank (Mei et al., 2010) belongs to the time-variant random walk process family which incorporates a variable to record the number of times of nodes having been visited. DivRank can balance prestige and diversity simultaneously by decreasing the visiting times of certain nodes. The iteration process of DivRank can be described as follows:

$$DR^{T+1}(s_i) = (1-\lambda)p^*(s_i) + \lambda \sum_{j:j\neq i} \frac{w_{ji} \cdot N^T(s_i)}{D^T(s_j)} DR^T(s_j), \tag{16}$$

   where $DR^T(s_i)$ denotes the DivRank score of sentence s_i at the Tth iteration and $p^*(s_i)$ represents the prior value of sentence s_i. $N^T(s_i)$ is the number of times the walk has visited s_i up to time T and $D^T(s_j) = \sum_{k:k\neq j} w_{kj} N^T(s_k)$. Similarly, w_{ji} is the weight between sentence s_j and s_i and denotes the transition probability from s_j to s_i.

4.3 Aggregation methods

Aiming to compare the proposed supervised aggregation method with other aggregation methods, we implement several aggregation methods using different ensemble strategies in the experimental studies including average scores, Round Robin (RR), unsupervised learning algorithm for rank aggregation (ULARA), and Weighted consensus summarization (WCS). A brief description of these aggregation methods is presented in this section.

1. Average Score (Avg_Score)
   This method normalizes the raw scores generated by different summarization systems between 0 and 1, and then uses the average score $Avg\_Score(s) = \frac{\sum_{k=1}^{K} score_k(s)}{K}$ to rank the sentences. In the formula, K is the number of summarization systems and $score_k(s)$ is the score of sentence s from kth summarization system.

2. Round Robin (RR)
   Refer to (Wang and Li, 2011), RR chooses the first sentence produced by the first summarizer and then the first sentence by the second summarizer. After all the first sentence are selected in the first round, the second round chooses the second sentences in the same way until reaching the summary length limit.

3. ULARA
   Unsupervised learning algorithm for rank aggregation (ULARA) is proposed in (Klementiev et al., 2007). ULARA applied a linear combination of the individual ranking approaches to form the aggregation result by rewarding ordering agreement between different rankers. By minimizing the weighted variance-like measures, the optimal weights assigned to component rankers are obtained.

4. Weighted Consensus Summarization (WCS)
   WCS algorithm (Wang and Li, 2011) utilizes a weighted consensus scheme to combine the results from individual summarizers. In this algorithm, the contribution from each summarization system is determined by its agreement with other systems. By minimizing the weighted distance between the consensus ranking and the individual ranking lists generated by different summarization systems, the weights that will be assigned to individual summarizers are obtained.

5. Supervised Summarization Aggregation Method (SSA)
   SSA is the supervised aggregation method for summarization described in Section 3.

In the experiments, we study the summarization performance of the implemented individual and aggregation systems, and compare the proposed supervised summarization aggregation method with other combination methods.

4.4 Redundancy removal

In order to choose more informative but less redundant sentences as the final summary, a redundancy removal step is conducted to impose the diversity penalty. In the experiments, we use the diversity penalty algorithm proposed in (Wan et al., 2007) to remove redundant sentences by introducing a penalty degree factor ω. The algorithm is described briefly in Algorithm 1.

5 Experiments

5.1 Data set

To evaluate the summarization results empirically, we use DUC2004[2] data set since generic multi-document summarization is one of the fundamental tasks in DUC2004. The data set

[2]http://duc.nist.gov/

Algorithm 1 Redundancy Removal Algorithm

1: Initialize set $S_A = \emptyset$, $S_B = \{s_1, s_2, ..., s_n\}$, and every sentence s_i in set S_B has a score $score(s_i)$ which initially is computed by the score function $r(\cdot)$;
2: Sort sentences in set S_B according to their scores in descending order;
3: Choose sentence s^* with the highest score in set S_B and move it from S_B to S_A;
4: **for** $s_j \in S_B$ **do**
5: $score(s_j) = score(s_j) - \omega \cdot w_{ji} \cdot r(s_i)$
6: **end for**
7: Go to step 2 and iterate until S_A reaches the length limit of a summary or $S_B = \emptyset$.

provides 50 document clusters and every generated summary is limited to 665 bytes. In the experiments, we randomly choose 40 document clusters as training data and the remaining 10 clusters are used as the testing data. For consistency, all the evaluation results are based on the testing set.

5.2 Evaluation methods

ROUGE (Lin and Hovy, 2003) (Recall Oriented Understudy for Gisting Evaluation) is widely applied for summarization evaluation by DUC. Therefore, we use the ROUGE toolkit[3] to evaluate the summarization results. It evaluates the quality of a summary by counting the overlapping units between the candidate summary and model summaries. ROUGE implements multiple evaluation metrics to measure the system-generated summarization such as ROUGE-N, ROUGE-L, ROUGE-W and ROUGE-SU. ROUGE-N is an n-gram recall measure computed as follows:

$$ROUGE-N = \frac{\sum_{S \in ref} \sum_{n-gram \in S} Count_{match}(n-gram)}{\sum_{S \in ref} \sum_{n-gram \in S} Count(n-gram)} \quad (17)$$

where n represents the length of n-gram, and $Count_{match}(n-gram)$ is the maximum number of n-grams co-occurring in the candidate summary and reference summaries. $Count(n-gram)$ is the number of n-grams in the reference summaries.

The ROUGE toolkit can report separate scores for 1, 2, 3 and 4-gram and among these different metrics, unigram-based ROUGE score (ROUGE-1) has been shown to correlate well with human judgments. Besides, longest common subsequence (LCS), weighted LCS and skip-bigram co-occurrences statistics are also used in ROUGE. ROUGE can generate three scores, i.e. recall, precision and F-measure, for each of the methods. In the experimental results we show three of the ROUGE metrics: ROUGE-1 (unigram-based), ROUGE-2 (bigram-based), and ROUGESU4 (extension of ROUGE-S, which is the skip-bigram co-occurrences statistics) metrics.

5.3 Evaluation results

5.3.1 Performance comparison

The proposed supervised summarization aggregation method is compared with different aggregation schemes including average score, round robin, ULARA, and WCS which are introduced in Section 4.3. In order to analyze the improvement of the aggregation method, we also list the

[3]ROUGE version 1.5.5 is used in this study, and it can be found on the website http://www.isi.edu/licensed-sw/see/rouge/

performance of all the individual methods. Besides, we also use Lead method as the baseline. The lead baseline takes the first sentences one by one in the last document in a document set, where documents are assumed to be ordered chronologically. Table 1 shows the comparison results (F-measure) on DUC2004 data set in ROUGE-1, ROUGE-2 and ROUGE-SU4 along with corresponding 95% confidence intervals and Figure 2 gives an illustration of the comparison on ROUGE-1 metric (LexPR is short for LexPageRank shown in the figure).

Systems	ROUGE-1	ROUGE-2	ROUGE-SU4
Lead	0.31861 (0.30886 - 0.32820)	0.06814 (0.06102 - 0.07631)	0.10554 (0.09953 - 0.11208)
LexPageRank	0.36211 (0.35081 - 0.37384)	0.07808 (0.07027 – 0.08675)	0.11982 (0.11311 - 0.12717)
LexHITS	0.35285 (0.33981 - 0.36551)	0.06911 (0.06137 – 0.07675)	0.11485 (0.10771 - 0.12233)
Manifold	0.37809 (0.36785 - 0.38866)	0.08046 (0.07220 – 0.08890)	0.12577 (0.11952 - 0.13224)
DivRank	0.37442 (0.36076 - 0.38693)	0.08255 (0.07400 – 0.09058)	0.12503 (0.11721 - 0.13248)
Avg_Score	0.37814 (0.36716 - 0.38914)	0.08690 (0.07900 – 0.09520)	0.12823 (0.12155 - 0.13490)
RR	0.36809 (0.35489 - 0.38028)	0.08095 (0.07255 – 0.08984)	0.12412 (0.11672 - 0.13141)
ULARA	0.37971 (0.36720 - 0.39183)	0.09010 (0.08186 - 0.09837)	0.13163 (0.12399 - 0.13880)
WCS	0.38227 (0.37019 - 0.39334)	0.09133 (0.06669 - 0.11785)	0.13285 (0.11178 - 0.15798)
SSA	**0.39766** (0.36761 - 0.42835)	**0.09528** (0.07186 - 0.12093)	**0.13939** (0.11906 - 0.16217)

Table 1: Overall performance comparison on DUC2004

From the comparison results, it can be seen that the proposed supervised summarization aggregation (SSA) method can outperform all the other ensemble methods and individual summarization approaches on all the three metrices. This comparison indicates that by incorporating human labeled data, supervised aggregation method has an advantage over unsupervised ensemble methods, i.e., when identifying the reliability of score list from a single summarizer, human labeled data could serve as an precise guidance. Moreover, almost all the aggregation methods perform better than individual summarization systems except the round robin method and this result is consistent with the comparison in (Wang and Li, 2011). For one thing, the results demonstrate that generally ensemble methods can effectively enhance the performance since these methods make the best of individual summarization methods which rank sentences from different aspects. For another, the poor performance of round robin method may result from that simply choosing the sentence with highest score in every round ignores the relationship among different score lists and the inaccuracy or overlap of the top sentences can lead to the poor effect as well.

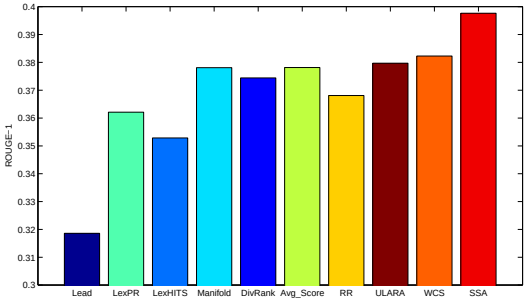


Figure 2: The comparison results of all the methods.

5.3.2 Influence of training data construction schemes

As mentioned in Section 4.1, there are two schemes to construct the training data (i.e., *average* and *maximum*). Average scheme chooses the average probability of an n-gram under all human summaries as the measure and maximum scheme applies the maximum probability value to represent the probability of an n-gram under all human summaries.

From the comparison shown in Table 2, we observe that the maximum scheme performs better than the average one. By analyzing the definition of two schemes, *maximum* tends to assign sentences with a higher value when compared with the *average* scheme and therefore it can choose more potential summary sentences into positive training set and produce better results.

Schemes	ROUGE-1	ROUGE-2	ROUGE-SU4
maximum	0.39766	0.09528	0.13939
average	0.39394	0.09306	0.13859

Table 2: Results of different schemes on DUC2004

5.3.3 Influence of component diversity

In the experimental study, we exploit four different summarization methods which are proven effective in summarization and they rank sentences from different aspects by utilizing diverse strategies. Therefore differences in algorithms and implementation make the ensemble process can comprehensively take into consideration multiple ranking strategies.

The LexPageRank summarization method scores sentences by making use of the voting or recommendations between sentences, and thus the global information of all the sentences can be in full use. LexHITS method assigns each sentence two properties, i.e., hub and authority, which can take into account the mutual relationship between sentences and provide a better view of the relationships embedded in the sentences. Manifold-ranking based method is based on a universal ranking algorithm and can capture the underlying manifold structure of data,

thus some implicit relationships between sentences are exploited. DivRank can be regarded as an expand version of LexPageRank which incorporates a variable to record the number of times of nodes having been visited. Through this improvement, the diversity and prestige of sentences to be chosen can be guaranteed simultaneously.

In this set of experiments, we further investigate the influence of different combinations of component methods[4]. We use the proposed framework to combine any three of all summarization methods and compare the results with the combination scheme using all the four methods. Table 3 shows the comparison results.

Systems	ROUGE-1	ROUGE-2	ROUGE-SU4
LH+MF+DR	0.39394	0.09306	0.13859
LPR+MF+DR	0.39393	0.09397	0.13834
LPR+LH+DR	0.38829	0.08172	0.12807
LPR+LH+MF	0.39383	0.09355	0.13916
All	0.39766	0.09528	0.13939

Table 3: Comparison results of different component combinations on DUC2004

From the table, it can be seen that different component summarization methods have less impact on the results in general and this may due to the small number of testing data. However, from the little fluctuation we can observe that aggregating all the individual methods can perform the best results. It is worth mentioning that the performance of LPR+LH+DR is relatively poor, and this result may owe to losing the Manifold-ranking based method which performs best among all the four methods.

Conclusion and perspectives

In this paper, we propose a supervised aggregation summarization framework by combining the results from four typical multi-document systems including LexPageRank, LexHITS, Manifold-ranking method and DivRank. To evaluate the proposed appraoch, we compare it with several combination methods, e.g., average score, round robin, unsupervised learning algorithm rank aggregation (ULARA) and weighted consensus summarization (WCS). And the experimental results on DUC 2004 data set demonstrate the effectiveness of our proposed framework. In addition, irrespective of the specific individual summarization methods used in this study, the supervised aggregation framework for summarization can also incorporate some more delicate and effective summarizers and generate more promising summary.

In this study, we investigate the supervised summarization aggregation approach to learn weights automatically by incorporating human labeled data. Since labeled sentences would be time-consuming and costly, in the future we will explore semi-supervised method which can decrease the amount of labeled data required. Moreover, more effective individual summarization approaches would be exploited and added into the ensemble method.

Acknowledgments

This work is supported by NSFC under the grant No. 60933004 and 61073082. We also would like to thank the reviewers for their useful comments.

[4]LH, LPR, MF and DR are short for LexHITS, LexPageRank, Manifold and DivRank, respectively.

References

Celikyilmaz, A. and Hakkani-Tur, D. (2010). A hybrid hierarchical model for multi-document summarization. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 815–824. Association for Computational Linguistics.

Chen, S., Wang, F., Song, Y., and Zhang, C. (2011). Semi-supervised ranking aggregation. *Information Processing & Management*, 47(3):415–425.

Erkan, G. and Radev, D. (2004). Lexpagerank: Prestige in multi-document text summarization. In *Proceedings of EMNLP*, volume 4.

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

Hahn, U. and Mani, I. (2000). The challenges of automatic summarization. *Computer*, 33(11):29–36.

Hoi, S. and Jin, R. (2008). Semi-supervised ensemble ranking. In *Proceedings of the 23rd national conference on Artificial intelligence*, volume 2.

Jin, F., Huang, M., and Zhu, X. (2010). A comparative study on ranking and selection strategies for multi-document summarization. In *Proceedings of the 23rd International Conference on Computational Linguistics: Posters*, pages 525–533. Association for Computational Linguistics.

Joachims, T. (2002). Optimizing search engines using clickthrough data. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 133–142. ACM.

Kleinberg, J. (1999). Authoritative sources in a hyperlinked environment. *Journal of the ACM (JACM)*, 46(5):604–632.

Klementiev, A., Roth, D., and Small, K. (2007). An unsupervised learning algorithm for rank aggregation. *Machine Learning: ECML 2007*, pages 616–623.

Li, H. (2011). Learning to rank for information retrieval and natural language processing. *Synthesis Lectures on Human Language Technologies*, 4(1):1–113.

Li, L., Zhou, K., Xue, G., Zha, H., and Yu, Y. (2009). Enhancing diversity, coverage and balance for summarization through structure learning. In *Proceedings of the 18th international conference on World wide web*, pages 71–80. ACM.

Lin, C. and Hovy, E. (2002). From single to multi-document summarization: A prototype system and its evaluation. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, pages 457–464. Association for Computational Linguistics.

Lin, C. and Hovy, E. (2003). Automatic evaluation of summaries using n-gram co-occurrence statistics. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1*, pages 71–78. Association for Computational Linguistics.

Liu, Y., Liu, T., Qin, T., Ma, Z., and Li, H. (2007). Supervised rank aggregation. In *Proceedings of the 16th international conference on World Wide Web*, pages 481–490. ACM.

Mei, Q., Guo, J., and Radev, D. (2010). Divrank: the interplay of prestige and diversity in information networks. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1009–1018. ACM.

Mihalcea, R. and Tarau, P. (2004). Textrank: Bringing order into texts. In *Proceedings of EMNLP*, volume 4. Barcelona: ACL.

Nenkova, A., Vanderwende, L., and McKeown, K. (2006). A compositional context sensitive multi-document summarizer: exploring the factors that influence summarization. In *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 573–580. ACM.

Ouyang, Y., Li, S., and Li, W. (2007). Developing learning strategies for topic-based summarization. In *Proceedings of the sixteenth ACM conference on Conference on information and knowledge management*, pages 79–86. ACM.

Ouyang, Y., Li, W., Lu, Q., and Zhang, R. (2010). A study on position information in document summarization. In *Proceedings of the 23rd International Conference on Computational Linguistics: Posters*, pages 919–927. Association for Computational Linguistics.

Page, L., Brin, S., Motwani, R., and Winograd, T. (1999). The pagerank citation ranking: Bringing order to the web.

Radev, D., Jing, H., Stys, M., and Tam, D. (2004). Centroid-based summarization of multiple documents. *Information Processing & Management*, 40(6):919–938.

Shen, C. and Li, T. (2011). Learning to rank for query-focused multi-document summarization. In *Data Mining (ICDM), 2011 IEEE 11th International Conference on*, pages 626–634. IEEE.

Shen, D., Sun, J., Li, H., Yang, Q., and Chen, Z. (2007). Document summarization using conditional random fields. In *Proceedings of IJCAI*, volume 7, pages 2862–2867.

Svore, K., Vanderwende, L., and Burges, C. (2007). Enhancing single-document summarization by combining ranknet and third-party sources. In *Proceedings of the 2007 joint conference on empirical methods in natural language processing and computational natural language learning (EMNLP-CoNLL)*, pages 448–457.

Tang, J., Yao, L., and Chen, D. (2009). Multi-topic based query-oriented summarization. In *Proceedings of SDM*, volume 9.

Van, M. and Erp, S. (2000). L.: Variants of the borda count method for combining ranked classifier hypotheses. In *in the Seventh International Workshop on Frontiers in Handwriting Recognition*, pages 443–452.

Wan, X. and Xiao, J. (2009). Graph-based multi-modality learning for topic-focused multi-document summarization. In *Proceedings of the 21st international jont conference on Artifical intelligence (IJCAI-09)*, pages 1586–1591.

Wan, X. and Yang, J. (2008). Multi-document summarization using cluster-based link analysis. In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 299–306. ACM.

Wan, X., Yang, J., and Xiao, J. (2007). Manifold-ranking based topic-focused multi-document summarization. In *Proceedings of IJCAI*, volume 7, pages 2903–2908.

Wang, D. and Li, T. (2010). Many are better than one: improving multi-document summarization via weighted consensus. In *Proceeding of the 33rd international ACM SIGIR conference on Research and development in information retrieval*, pages 809–810. ACM.

Wang, D. and Li, T. (2011). Weighted consensus multi-document summarization. *Information Processing & Management*.

Wei, F., Li, W., and He, Y. (2009). Co-feedback ranking for query-focused summarization. In *Proceedings of the ACL-IJCNLP 2009 Conference Short Papers*, pages 117–120. Association for Computational Linguistics.

Wong, K., Wu, M., and Li, W. (2008). Extractive summarization using supervised and semi-supervised learning. In *Proceedings of the 22nd International Conference on Computational Linguistics-Volume 1*, pages 985–992. Association for Computational Linguistics.

Zhou, D., Weston, J., Gretton, A., Bousquet, O., and Scholkopf, B. (2004). Ranking on data manifolds. *Advances in neural information processing systems*, 16:169–176.

Collective Search for Concept Disambiguation

Anja PILZ *Gerhard PAASS*
Fraunhofer IAIS, Schloss Birlinghoven, 53757 Sankt Augustin, Germany
anja.pilz@iais.fraunhofer.de, gerhard.paass@iais.fraunhofer.de

ABSTRACT
Name ambiguity is a major problem in information retrieval: The name "Metropolis" may refer to a movie, a physicist, or Superman's hometown. Recent work resolves ambiguity in natural language text by linking name mentions against the corresponding Wikipedia concept (Wikification). Standard methods comparing a single mention with the corresponding Wikipedia concept can potentially be improved by simultaneously considering all mentions in the input document. We propose a novel multiple assignment process based on a collective search over an inverted index that exploits the coherence of Wikipedia concepts. Based on this coherence, we compute the best fitting candidate concept for each mention and combine it with context information in a second search step. Using additional attributes an SVM then re-ranks the result of this search and estimates if a concept is not covered in Wikipedia. We give a unified view over the different performance measures used in other state-of-the art approaches and evaluate our approach on five benchmark corpora. On these corpora, our method has the most stable performance yielding similar or better results compared to other approaches.

KEYWORDS: Concept and Entity Disambiguation, Wikification, Natural Language Processing, Search and Ranking.

Proceedings of COLING 2012: Technical Papers, pages 2243–2258,
COLING 2012, Mumbai, December 2012.

1 Introduction

A major aim of search engines is the retrieval of information about concepts which may be any existing object, e.g. person, thing, notion, etc., with a designation or name. In natural language text however, many concepts share the same name and one concept may be referenced by different names. Consequently, a search based on pure string matching often yields many irrelevant results, such as a web page on Superman's hometown when indeed the user seeked information on the physicist Metropolis. Concept disambiguation which assigns the correct sense to the mention of a concept in a given context, can reduce the number of irrelevant results or group results by sense. The disambiguation of concept mentions is required in applications such as semantic search, but also many other areas like knowledge base construction or data base curation.

Recent work, for example (Ratinov et al., 2011), resolves name ambiguity by linking the name mention against the corresponding Wikipedia article, thus often terming the problem *Wikification*. For that, a name mention together with the features of its neighboring context is compared to the corresponding features of the Wikipedia article. If the difference between these features is small, a Wikipedia concept is linked to the mention and thus the name's ambiguity considered as resolved. A large number of features have been evaluated for concept disambiguation. Starting with simple bag-of-word descriptions more advanced features were developed characterizing the sense of surrounding words, e.g. topic model indices (Pilz and Paass, 2011). But often, approaches remained *local* and did not exploit the *global* coherence of candidate concepts.

In this paper we follow the *global* approach by simultaneously considering all mentions of an input document and jointly exploiting relations between potential concepts. We present a novel measure for concept coherence. We encode this information in a search index allowing fast and comprehensive access to the relational information present in large knowledge bases such as Wikipedia. One deficit of most current concept disambiguation methods is that they do not thoroughly handle the case when a concept mention is not covered by Wikipedia (nil-concepts). We use an SVM classifier to fine-tune the assignment and to detect nil-concepts. We discuss the various evaluation measures presented in other papers and apply our algorithm to five benchmark corpora. While fast and memory efficient, our algorithm yields similar or better results than its competitors and has the most stable performance of the compared methods.

2 Related Work

Concept disambiguation is closely related to the task of word sense discrimination (Schuetze, 1998), but in addition links the concepts to entries in a reference knowledge base which is often Wikipedia. Standard or *local* approaches like Cucerzan (2007) build word and feature vectors over the words occurring in a context window around the concept mention m and cluster them using similarity measures such as cosine similarity. Bunescu and Pasca (2006) correlate context words with Wikipedia categories to formulate a word-taxonomy kernel. This is used in a Ranking SVM which generates a ranked list of plausible Wikipedia concepts for a given context of a name mention m. Pilz and Paass (2011) showed that topic model indices instead of bag-of-word approaches provide a more informative context representation with better generalization properties.

Recent work on concept disambiguation follows a more *global* approach, where all concept mentions in a document are disambiguated collectively using a coherence measure that is usually

derived from the graph built over an existing knowledge base. Kulkarni et al. (2009) formulate concept assignment as an optimization problem that assigns concepts to mentions such that the mention-concept compatibility and global concept-concept coherence is maximal. They solve the problem using local hill-climbing and linear program relaxations, yielding favorable results on the **MSNBC** corpus (Cucerzan, 2007) as well as their own dataset **IITB**. Han et al. (2011) propose a graph-based collective concept linking method which can model and exploit the global interdependence between different assignment decisions. Ratinov et al. (2011) present the disambiguation model GLOW, a global approach that employs the normalized Google distance (Milne and Witten, 2008) as well as pointwise mutual information to measure the relatedness between concepts. To refine the assignment decision they additionally exploit the conditional probability that a concept belongs to a mention based on Wikipedia link information. Hoffart et al. (2011b) introduced AIDA which employs YAGO2 (Hoffart et al., 2011a) as an entity catalog and a rich source of entity types and semantic relationships among entities. They build a graph containing mentions from the input text and candidate concepts from the reference set as nodes. The edges are weighted capturing context similarities as well as coherence between Wikipedia concepts. Using a greedy algorithm they identify a dense sub-graph that contains exactly one mention-concept edge for each mention, yielding the most likely disambiguation.

We propose an approach that is based on a search index. The usefulness of search indices for concept resolution was also observed by Song and Heflin (2011) who present an efficient and scalable system for concept resolution on structured data. Opposed to our objective which is concept resolution in unstructured data, exploitable attributes are very different and often carry an inherent distinctive function. In the sequel, we give the details of our approach and compare it to a representative selection of four recent works showing that it is the most stable method yielding similar or better results on different benchmark corpora. Although all prior work shows improved results on benchmark corpora, none of them handles nil-concepts thoroughly. For specific tasks this might be appropriate, but in a more general setting this means a drastic simplification as most entities (e.g. persons) are *not* covered by Wikipedia.

3 Disambiguation as a Search Problem

We study the task of Wikification, i.e. concept disambiguation using Wikipedia as a reference knowledge base. We use the English version of Wikipedia[1] and represent it in the Lucene[2] search index **Wiki** that allows efficient search over the concepts contained in Wikipedia.
We resolve the ambiguity of a *mention* m in a text document through its assignment to a *unique concept* $c(m)$ described in Wikipedia, i.e. $c(m) \in \mathbf{Wiki} = \{c_1, ..., c_{|\mathbf{Wiki}|}\}$. If the true concept for m is not covered by an article in Wikipedia, then $c(m) \in \mathbf{C}_0$, the set of nil-concepts that we do not distinguish. Basically, **Wiki** contains all Wikipedia concepts apart from meta pages. We also excluded disambiguation pages since we assume that an assignment to such a page does not solve the task of name disambiguation. Furthermore, the varying usage of Wikipedia mark up language led to un-processable documents that are also not contained in **Wiki**. Thus, in the following, we distinguish between linkable concepts contained in the index $c \in \mathbf{Wiki}$, nil-concepts c_0 originally not covered by Wikipedia and ignored or missing concepts $\tilde{c}_0 \notin \mathbf{Wiki}$.

We assume the input to be a natural language text document with a collection of mentions $\mathbf{M} = \{m_1, \ldots, m_k\}$ to disambiguate. In the case of the benchmark corpora, these mentions are given. In other real-world applications, they can be provided by an automatic annotator, such

[1] Downloaded on September 1th, 2011.
[2] An open source search engine for large scale text collections, `http://lucene.apache.org/`

as a noun phrase or named entity recognizer (NER). Note that we do not restrict the mentions to named entities (persons, locations, etc) but also treat general concepts such as *bank* or *tree*.

To improve the individual disambiguation performance for each m_i, we simultaneously consider all mentions $\mathbf{M}$ to determine the best fitting *candidate concepts* $bestFit(m_i)$. We propose a disambiguation process that uses the search index **Wiki** to generate candidate concepts, as well as a supervised SVM classifier to adjust the ranking of these candidates and to detect nil-concepts c_0. This process consists of the following steps that are described in more detail in the following sections:

Step 1 Run a **collective search** using an **ensemble query** with terms from all mentions $m_1, \ldots, m_k$ to create sets of **potential candidates** $\mathbf{C}_i \subset$ **Wiki** for each m_i (Alg. 1.1-1.10).
Step 2 Compute the **cross coherence** over all candidates in the sets $\mathbf{C}_1, \ldots, \mathbf{C}_k$ to find related concept sets (c.f. Eq. 4), Alg.1.11-1.13).
Step 3 Determine the $bestFit_i \in$ **Wiki** for each mention m_i, based on the maximum cross coherence of each candidate in $\mathbf{C}_i$ (c.f. Eq. 6, Alg. 1.15).
Step 4 For each m_i combine the attributes of m_i and $bestFit_i$ into one query and search **Wiki**, which yields a set of improved concepts $\mathbf{C}_i^* \subset$ **Wiki** (Alg. 2.2-2.12).
Step 5 Apply an **SVM** classifier to all $\mathbf{C}_1^*, \ldots, \mathbf{C}_k^*$ for **re-ranking** and **nil-concept detection**, resulting in the final predicted concept $\hat{c}_i \in \mathbf{C}_i^* \cup \mathbf{C}_0$ for each m_i (c.f. Sec. 4.2, Alg. 2.17).

3.1 Concept Attributes in the Wiki index

Using the information stored in the article itself as well as Wikipedia's hyperlink graph, we enhance the representing concept $c \in$ **Wiki** with the searchable fields outlined in this section.

Name fields Special attention is given to name fields, since for unambiguous mentions the name is often sufficient for linkage. Each concept has a unique `titleLong` field which contains the title of the associated Wikipedia article. From this, we generate additional fields. The `title` field stores the part of `titleLong` that is not used as a disambiguation term (usually a qualifying term in parentheses). Abbreviations are generated via a simple heuristic and stored in separate `abbreviation` fields. As an example, the index concept representing the Wikipedia article *Michael Jordan (footballer)* has the fields: (`titleLong`, "Michael Jordan (footballer)"), (`title`, "Michael Jordan"), (`abbreviation`, "MJ"), (`abbreviation`, "M. Jordan") etc.

Furthermore, we add the redirect information from the Wikipedia redirect dump to the corresponding index concepts. In general, redirects provide a large resource of synonyms. In some cases, however, they can also be misleading, since they do not necessarily compose equivalence relations. For instance, *Ulrich Merkel* is a redirect for German chancellor *Angela Merkel*, but actually is the latter's spouse. Still, we consider all redirects without pre-processing, since a more well defined redirect scheme would already require a disambiguation step. The index concept for *Angela Merkel* is hence enriched with the field (`redirect`,"Ulrich Merkel").

Inspired by Ratinov et al. (2011), we create `meantBy` fields that, similar to redirects, provide concept names that may not be found in the article text itself. In a pre-processing step, we iterate over all articles in Wikipedia and analyze the pairs (c, m) of link target concept c and associated anchor text m. For each pair (c, m) we record the frequency of occurrence $\#(c, m)$ and estimate the concept-mention probability $p(c|m)$ through

$$p(c|m) \approx \frac{\#(c, m)}{\sum_{c_i \in \mathbf{Wiki}} \#(c_i, m)}. \quad (1)$$

For instance, we obtain $p(Japan|\text{"Japan"}) \approx 0.97$. Note that these are not true probabilities, since due to parsing errors or too aggressive stemming, we may observe that $\sum_i p(c_i|m) \neq 1$.

Lucene ranks the search results for a query according to a product of the following factors: the term frequency of the term x in the document, its inverted document frequency *idf(x)*, a weight factor *boost(x)* and the document's length norm (Hatcher et al., 2010). For the final index creation, we use the above probabilities as *boosts* on the `meantBy` fields: the index concept for *Japan* has the field (`meantBy`,"Japan", 0.97), where the field's searchable content is the surface form "Japan" and the field's boost is the estimated probability value $p(c|m) = 0.97$. To keep memory consumption as low as possible, we create an auxiliary index to retrieve these values efficiently.

In the following we refer to the above fields as *name* fields. Name fields allow queries of the form (`title`, m), (`redirect`, m) or (`meantBy`, m). In our experiments, we will show results when additional context information is ignored and only name fields are used for disambiguation.

Context fields Assuming that each concept is thoroughly depicted in the article's main text, we use this context (except stop words) in a designated `context` field. This allows us to place queries of the form (`context`,"w"), where "w" may be the mention itself or any other key word extracted from the input document.

Type fields For all **Wiki** concepts that can be automatically aligned with YAGO (Suchanek et al., 2008), we add the type information extracted from YAGO, such as person, location, etc.. If the mention text has been tagged as a named entity by a NER, we can use this additional meta information to place a more distinctive query, for example a query (`type`,"person").

Both context and type fields can be queried separately, and we will show the influence of context and type usage in our experiments.

Link fields Relational information is an important factor for concept resolution and Wikipedia's link structure provides a straightforward resource to model relations among concepts. We store all outlinks $\{c \rightarrow c'\}$ of a Wikipedia concept in the fields (`linkText`,"m") of the respective index concept c, where "m" is the anchor text used for the outlink target concept c'. These fields are used to compute the relatedness among concepts (c.f. Eq. 3) but also queried in the collective search step of our disambiguation algorithm (Alg. 1.1-1.2).

3.2 Mention-specific Attributes

To create specific disambiguating attributes for each mention m_i, we first extract the mention's *name*, *type* and *context* attributes from the input document.

Name and type attributes Having collected all mentions from the input document, we keep the name (i.e. the surface form) and if present, the type information as attributes for each m_i. We then run a *mention expansion* that searches for mentions that are token-wise contained in previous mentions. If the type of two mentions is the same, the shorter mention is expanded to the longer one. For example, if $\mathbf{M}$ ={("Al Gore", per), ("Gore", per), ("Gore Bay", loc)}, the result of mention expansion is $\mathbf{M}$ ={("Al Gore", per), ("Al Gore", per), ("Gore Bay", loc)}. If the NER did not identify the type of "Gore", we still assume that it refers to the person "Al Gore", since the abbreviation of person names is much more common compared to the abbreviation

of location names. In our experiments, we found that the expansion of mention names has a positive impact on disambiguation performance.

Context attributes We use both local as well as document level context information. The local context is a $[2,2]$ noun-window around the mention without stop words. Additionally, we extract tf-idf ranked key words from the document text and keep the 20 words with highest tf-idf value as document key words. This set is then localized for each m_i: from the joint set of local context words and document key words, we keep only those words that appear at least once in the text of an index concept whose title matches m_i. In the same way, we compute key words from the headline of the input document, assuming that headline information is especially important.

Topic information Additionally to the pure word-based context information, we use an LDA topic model (Blei and Lafferty, 2009) to infer the most likely topic distribution of the input document. The LDA model was trained with $Z = 500$ topics on the **CoNLL** training corpus (c.f. Sec. 5) where words are the surface forms of the named entities appearing in the documents. We then apply this topic model to the input document giving local context words of mention m_i a five-fold weight. This yields a specific topic distribution $topic(m_i)$ for each mention m_i.

Name, type and context attributes of the input mentions can be matched to the according index fields using specific queries. Topic information is used for relatedness computation as well as a distinct feature for the Ranking SVM.

4 Disambiguation via Search and Ranking

Having defined the components of our search index and the input to our system, we explain the search process for Wikification in this section. The first part of our disambiguation procedure is to *jointly* treat all mentions $m_1, \ldots, m_k$ in the input document to generate a *bestFit* candidate for each m_i. The algorithm for this is depicted in Alg. 1

4.1 bestFit concepts from collective search using ensemble queries

Our assumption is that Wikipedia articles containing many of the input mentions are likely to be of a similar content as the input document. From the *outlink target concepts* these articles provide, we can automatically generate good disambiguation candidates concepts (**step 1**).

To retrieve these candidate concepts, we create an *ensemble query* that jointly treats the names of all mentions m_i and thus exploits the co-occurrence of mentions as link texts (see Alg. 1.1). This query then contains one query term (`linkText`, m_i) per mention m_i. Using this query, a search in **Wiki** then yields a ranked list of concepts $\mathbf{C}_{coll}$ that *collectively* contain the input mentions m_i as values in their `linkText` fields (Alg. 1.2). Lucene ranks each concept $c_{coll} \in \mathbf{C}_{coll}$ with a score s_L, based on the number of matches c_{coll} has on the fields (`linkText`, m_i). The higher the ranking of c_{coll}, the more mentions the concept c_{coll} contains as link text.
We keep the top 30 concepts in $\mathbf{C}_{coll}$ from which we extract the collection of outlink targets $\mathbf{C}_{\rightarrow}$. Next, we endow each outlink target concept $c \in \mathbf{C}_{\rightarrow}$ with a weight $w(c)$ that is the sum over the concepts' scores in which it appears as an outlink target, i.e. $c_{coll} \rightarrow c$ (Alg. 1.4):

$$\forall c \in \mathbf{C}_{\rightarrow}: \quad w(c) = \sum_{c_{coll} \in \mathbf{C}_{coll}} \delta_c s_L(c_{coll}), \quad \delta_c = \begin{cases} 1 & \text{iff } c_{coll} \rightarrow c, \\ 0 & \text{else.} \end{cases} \tag{2}$$

Since the collection $\mathbf{C}_{\rightarrow}$ may contain a huge number of concepts appearing only once as an outlink target, we keep only the top 100 candidate concepts in $\mathbf{C}_{\rightarrow}$, that have the highest weights $w(c)$.
Next, we need to relate the elements in the candidate concept set $\mathbf{C}_{\rightarrow}$ to the input mentions. More specifically, we analyze for each $c \in \mathbf{C}_{\rightarrow}$ if either the title or the redirect of c contains the text of mention m_i. If so, we add c to the candidate set $\mathbf{C}_i$ for mention m_i (Alg. 1.7 ff). Note that one c can then be contained in multiple candidate sets. The result of the collective search is the collection $\{\mathbf{C}_i\}_{i=1}^{k}$, where each $\mathbf{C}_i$ is a set of candidate concepts for mention m_i.

Our intuition is that concepts mentioned jointly in an input document should be related. To model the relatedness between Wikipedia concepts, we follow the approach of Milne and Witten (2008) who define the normalized Google distance (NGD) of two concepts c_i and c_j as

$$\mathrm{NGD}(c_i, c_j) = \frac{\log\Big(|\{c' \rightarrow c_i\} \cap \{c' \rightarrow c_j\}|\Big) - \log\Big(\max(|\{c' \rightarrow c\}|, |\{c' \rightarrow c_j\}|)\Big)}{\log(|\{c' \rightarrow \cdot\}|) - \log\Big(\min(|\{c' \rightarrow c_i\}|, |\{c' \rightarrow c_j\}|)\Big)}, \tag{3}$$

where $\{c' \rightarrow c_i\}$ is the collection of all concepts c' that link to c_i (i.e. the inlinks of c_i) and $|\{c' \rightarrow \cdot\}|$ is the total number of links in Wikipedia. In the case that the concepts c_i and c_j share no inlinks, i.e. $\{c' \rightarrow c_i\} \cap \{c' \rightarrow c_j\} = \emptyset$, we define $\mathrm{NGD}(c_i, c_j) = 0$.

Using the above NGD, we can measure the relatedness of two candidate concepts. To account for the collective fitness of a set of candidates, we introduce *cross coherence* which basically states how well a concept $c_{ij} \in \mathbf{C}_i$ *fits* to the other candidate concepts $\{\mathbf{C}_l\}_{l=1}^{k}$. More formally, we define the cross coherence of a candidate concept c_{ij} and a collection of concepts $\{\mathbf{C}_l\}_{l=1}^{k}$ as

$$\text{cross coherence}(c_{ij}, \{\mathbf{C}_l\}_{l=1}^{k}) = \frac{1}{k} \sum_{\substack{i'=1 \\ i' \neq i; \mathbf{C}_i \neq \mathbf{C}_{i'}}}^{k} \frac{1}{|\mathbf{C}_{i'}|} \sum_{\substack{c' \in \mathbf{C}_{i'} \\ c_{ij} \neq c'}} \Delta\mathrm{NGD}(c_{ij}, c'), \tag{4}$$

with k the number of mentions in the document, i the index of mention m_i and j the index over the candidate concepts for m_i. The second sum is the average NGD (Eq. 3) of c_{ij} to the concepts in another candidate set $\mathbf{C}_{i'}$ which is again averaged over all candidate sets by the first sum. Cross coherence can be interpreted as the average distance of a concept to a collection of concepts and has range $[0, 1]$, where 0 denotes a *completely unrelated* concept. We compute cross coherence in **step 2** (Alg. 1.11-1.13) to determine the relatedness of candidates extracted in the previous **step 1**.

The factor Δ in Eq. 4 serves as an additional relatedness weighting between two concepts. While both Milne and Witten (2008) and Ratinov et al. (2011) used the standard NGD with $\Delta = 1$, we analyze three additional weighting schemes. The scheme $\Delta_{\cos}\mathrm{NGD}$ weighs the NGD via the cosine distance $\cos(c_i, c_j)$ between the term vectors of two article texts. Additionally, we introduce $\Delta_{topics}\mathrm{NGD}$ that uses the thematical distance between two article link text collections. More specifically, we use a LDA topic model to infer the topic probability distribution over the words contained in a concept's outlink collection $\{c \rightarrow c'\}$ (for more details on the topic model, see 3.2). We define Δ_{topics} as the Hellinger distance between two concepts' outlink text topic probability distributions:

$$\Delta_{topics}(c_i, c_j) = 1 - \sum_{z=1}^{Z} \sqrt{topic_z(c_i) \cdot topic_z(c_j)}, \tag{5}$$

where $topic(c_i)$ and $topic(c_j)$ are the topic probability distribution vectors for the link texts of the concepts c_i and c_j and Z is the number of topics in the LDA model. The subtraction from 1 assures that $\Delta_{topics} = 0$ iff $topic(c_i) = topic(c_j)$ and is required to maintain the interpretation of cross coherence as a distance. The last relatedness measure we analyze is cosine distance without NGD.

In **step 3**, we compute the final result of the collective search procedure, i.e. the *bestFit* concepts. We define the *bestFit* candidate concept for each mention m_i by the product of the weight $w(c)$ (computed in **step 1**) and c's cross coherence value (computed in **step 2**):

$$bestFit_i = \arg\max_{c \in \mathbf{C}_i} (w(c) \cdot \text{cross coherence}(c)). \tag{6}$$

When the concept-mention association in **step 1(c)** yields no result, no *bestFit* candidate can be assigned. Note that if we used the triple of $w(c)$, cross coherence(c) and $p(c|m)$, high-prior candidates are likely to dominate, even if their coherence is low.

Algorithm 1: Collective search for *bestFit* candidate generation

Input: List of mentions $\mathbf{M} = \{m_1, \ldots, m_k\}$
Output: A $bestFit_i$ candidate for each $m_i \in \mathbf{M}$, i.e. $\{(m_1, bestFit_1), \ldots, (m_k, bestFit_k)\}$
1.1 $query = (\texttt{linkText}, name(m_1)) \wedge \ldots \wedge (\texttt{linkText}, name(m_k))$ // **step 1(a)**: create ensemble query using all m_i
1.2 $\mathbf{C}_{coll}$ = search **Wiki** using *query*
1.3 $\mathbf{C}_{\rightarrow} = \bigcup_{c_{coll} \in \mathbf{C}_{coll}} \{c_{coll} \rightarrow c'\}$ // collect outlink target concepts from collective search result $\mathbf{C}_{coll}$
1.4 **for** $c \in \mathbf{C}_{\rightarrow}$ **do** // **step 1(b)**: compute concept weights
1.5 compute concept weight according to Eq. 2
1.6 keep only top 100 link target concepts in $\mathbf{C}_{\rightarrow}$
1.7 **for** $m_i \in \mathbf{M}$ **do** // **step 1(c)**: relate concepts to mentions
1.8 initialize candidate set $\mathbf{C}_i = \emptyset$
1.9 **for** $c \in \mathbf{C}_{\rightarrow}$ **do**
1.10 add c to candidate set $\mathbf{C}_i$ if `title` or `redirect` of c contains mention text m_i
1.11 **for** $i = 1, \ldots, k$ **do** // **step 2**
1.12 **for** $c_{ij} \in \mathbf{C}_i$ **do**
1.13 compute cross coherence according to Eq. 4
1.14 **for** $m_i \in \mathbf{M}$ **do** // **step 3**
1.15 find *bestFit* concept according to Eq. 6
1.16 **return** $\{(m_1, bestFit_1), \ldots, (m_k, bestFit_k)\}$

4.2 Combining Search Results and Supervised Learning

The final disambiguation algorithm Alg. 2 has two steps (**step 4** and **5**). First, we run a search on **Wiki** to create ranked sets of candidate concepts $\mathbf{C}_1^*, \ldots, \mathbf{C}_k^*$ with one set $\mathbf{C}_i^* \subset$ **Wiki** per mention m_i. Second, a pre-trained SVM is applied to re-rank this output and detect nil-concepts. The result is the disambiguated list of input mentions, where each mention m_i is associated with a unique concept $\hat{c}_i \in$ **Wiki** $\cup \mathbf{C}_0$, i.e. $\{(m_1, \hat{c}_1), \ldots, (m_k, \hat{c}_k)\}$.

In the search part (**step 4**), we restrict the size of each $\mathbf{C}_i^*$ (i.e. the number of search results) to 5, which we experimentally found to be sufficient. Initially, we also require each concept $c_i^* \in \mathbf{C}_i^*$ to have at least 5 inlinks. This *inlink prior* aims at filtering out rarely referenced concepts. Then we run separate searches using only the `titleLong`, `title` and `redirect` fields of the index documents to find direct matches between mention m_i and concepts $c \in$ **Wiki** (Alg. 2.3). If such a match $\tilde{c}$ has been found, we give an additional query boost for the attributes of $\tilde{c}$, that is the title of $\tilde{c}$ is used as an additional query term with a five times higher weight than the other

query terms. For the *bestFit* concept we proceed analogously.
If either $\tilde{c}$ or $bestFit_i$ has a lower number of inlinks than initially assumed, the inlink prior is adapted automatically (Alg. 2.6). Alternatively, if the maximum returned score of the first search (Alg. 2.10) is less than a threshold $\tau = 1$, we re-run the search without the prior constraint (Alg. 2.12). After prioritisation on the results from direct and collective search, we add each mention's individual attributes to account for type and context information. In our experiments, we evaluate searches of different coverage, more specifically searches using

- name attributes, i.e. we add queries only on name fields (Alg. 2.7)
- name and type attributes, i.e. we extend the query using the mention's type (Alg. 2.8)
- name, type and context attributes, i.e. we additionally query context fields (Alg. 2.9).

Using this comprehensive query, the search result in **Wiki** is a either a set of ranked concepts $\mathbf{C}_i^*$ or an empty set, in which case the search did yield no result. We collect all concept sets $\mathbf{C}_i^*$ into an overall set $\{\mathbf{C}_i^*\}_{i=1}^k$ on which we apply a linear ranking SVM (**step 5**). Each concept c_i^* is represented by a vector of features that are computed both from the index ranking $s_L(c_i^*)$ as well as in relation to the input mention. We use the ranking in different feature representations: $s_{L,\log}(c_i^*) = \log s_L(c_i^*)$, $s_{L,norm}(c_i^*) = \frac{s_L(c_i^*)}{\sum_{c_i^{*'} \in \mathbf{C}_i^*} s_L(c_i^{*'})}$, $s_{L,rank}(c_i^*) = \frac{s_L(c_i^*)}{\arg\max\ s_L(c_i^* \in \mathbf{C}_i^*)}$. Additional features are the concept-mention probability $p(c_i^*|m_i)$, the cross coherence of c_i^* computed as in 4 but now in relation to the improved concept set $\{\mathbf{C}_i^*\}_{i=1}^k$, the Hellinger distance over the topic distributions $topic(m)$ and $topic(c_i^*)$. As proposed by Bunescu and Pasca (2006), we use a feature f_0 for nil-concepts c_0 that is required for the automatic detection of these nil-concepts.

We train the Ranking SVM on the **CoNLL train** corpus which is annotated with Wikipedia concepts as well as nil-concepts. Positive and negative examples are extracted in the same way as we generate disambiguation candidates. For instance, a positive example is the correct candidate c_i^* for a mention m_i and the negative examples are all other $c_i^* \in \mathbf{C}_i^*$ for that mention. Additionally, if not already present when the search did yield no result, we add a candidate c_0 for each mention whose only feature is f_0.

In the final **step 5** we use the trained SVM to re-rank the index output (Alg. 2.17). While the index search often provides a reliable candidate, implicit features such as coherence, concept-mention probability and topic similarity are only partially graspable by **Wiki** and may induce a SVM re-ranking.

5 Benchmark Corpora

Recent work published a variety of benchmark corpora for Wikification, most of them consisting of English newspaper articles from different time periods. Table 1 gives an overview of the corpora treated in this paper. The major difference between these corpora is the annotation scheme. Cucerzan, Ratinov et al., Milne and Witten and Kulkarni et al. treated mentions of all types on **MSNBC**, **ACE**, **AQUAINT**[3] and **IITB**[4] respectively. Hoffart et al. considered only named entity mentions in the **CoNLL** corpus[5]. Additional to differing mention types, there are also annotation differences that render comparison difficult. For instance, in **CoNLLb** the mention "Taiwan" is linked to *Republic of China*, while in **ACE** it is linked to *Taiwan*. We also observed

[3] **MSNBC**, **AQUAINT** and **ACE** are publicly available and described in detail in (Ratinov et al., 2011).
[4] **IITB** is publicly available and described in detail in (Kulkarni et al., 2009).
[5] **CoNLL** is publicly available and described in detail in (Hoffart et al., 2011b), we consider **CoNLL testb** called **CoNLLb** in the following.

Algorithm 2: Disambiguation algorithm

Input: List of mentions $\mathbf{M} = m_1, \ldots, m_k$, where each m_i has name, type, context & *bestFit* attributes.
Output: List of disambiguated mentions $\mathbf{M} = \{(m_1, \hat{c}_1), \ldots, (m_k, \hat{c}_k)\}$

2.1 **for** $m_i \in \mathbf{M}$ **do** // **step 4**
2.2 $p_{in} = 5$ // initialize inlink prior
2.3 $\tilde{c} = \text{directMatch}(m_i)$ // c.f. Sec. 4.2
2.4 **if** $\tilde{c} \neq \emptyset$ **then** add boosted query terms for attributes of $\tilde{c}$
2.5 **if** $bestFit_i \neq \emptyset$ **then** add boosted query terms for attributes of $bestFit_i$
2.6 $p_{in} = \min\left(p_{in}(\tilde{c}), p_{in}(bestFit_i), p_{in}\right)$ // reduce prior on inlinks
2.7 *query*.addNameQuery(name(m_i)) // add name attributes to the query terms
2.8 **if** *type* **then** *query*.addTypeQuery(type(m_i)) // add type attributes to the query terms
2.9 **if** *context* **then** *query*.addContextQuery(context(m_i)) // add context attributes to the query terms
2.10 $\mathbf{C}_i^* =$ search **Wiki** using *query* and inlink prior p_{in}
2.11 **if** $\max_{c_j^* \in \mathbf{C}_i^*} s_L(c_j^*) \leq \tau$ **then**
2.12 $\mathbf{C}_i^* =$ search **Wiki** using *query* without inlink prior
2.13 **if** $\mathbf{C}_i^* \neq \emptyset$ **then** $\{\mathbf{C}_i^*\} \cup \mathbf{C}_i^*$ **else** $\{\mathbf{C}_i^*\} \cup c_0$ // add $\mathbf{C}_i^*$ to concept set $\{\mathbf{C}_i^*\}$ or add c_0 if the search yields no results
2.14 **for** $i = 1, \ldots, k$ **do**
2.15 **for** $c_{ij}^* \in \mathbf{C}_i^*$ **do**
2.16 compute cross coherence$(c_{ij}^*, \{\mathbf{C}_l^*\}_{l=1}^k)$ according to Eq. 4 and set other features (c.f. Sec. 4.2)
2.17 $\hat{c}_i = \arg\max_{c_{ij}^* \in \mathbf{C}_i^*} \text{SVM rank}(c_{ij}^*)$ // **step 5**: rank candidates by trained SVM for final concept prediction
2.18 **return** $\mathbf{M} = \{(m_1, \hat{c}_1), \ldots, (m_k, \hat{c}_k)\}$

corpus	#documents	#Wikipedia concepts (unique)	$\#c_0$	$\#c \in$ **Wiki**	$\#\tilde{c}_0 \notin$ **Wiki**	$\varnothing\lvert\mathbf{M}\rvert$ per doc.
MSNBC	20	658 (279)	97	640	18	37.75
ACE	36	257 (185)	49	254	3	8.5
AQUAINT	50	727 (572)	0	702	25	14.54
CoNLLb	228	4363 (1527)	0	4317	46	19.13
IITB	104	11185 (3755)	0	9439	1746	107.54

Table 1: Benchmark corpora with number of documents, the number of (unique) Wikipedia concepts and nil-concepts c_0, the number of linkable concepts $c \in$ **Wiki**, the number of Wikipedia concepts $\tilde{c}_0$ missing in **Wiki** and the average number of mentions per document.

some inconsistencies in the **CoNLL** training corpus that are presumably due to inter-annotator disagreement (20%) or candidate selection: while the phrase "European Union" is linked to the appropriate Wikipedia concept, it's acronym "EU" is linked to c_0. While Hoffart et al. neglected nil-concepts for evaluation on **CoNLLb**, these inconsistencies might be harmful for the SVM training of our approach. Moreover, **CoNLLb** contains many news articles about sport events. These are often not truly natural language texts, but more table-like. These variations make it challenging to apply the same system to different corpora.

For all corpora we proceed as follows: given the input mention, we first check if the ground truth concept is linkable, i.e. contained in our index. If this is not the case, but the mention is linked to some $c \neq c_0$, we change the ground truth to $\tilde{c}_0$ which is always considered during evaluation. Since we also resolve redirects, the number of distinct concepts in Tab. 1 may differ from the one published in the respective paper. Note that the overall number of mentions remains unchanged. The procedure is the same for concepts that do no longer exist in Wikipedia.
For a consistent set of named entity tags, we run the Apache OpenNLP NER[6] on all corpora.

[6] `http://opennlp.apache.org/`

6 Performance Measures for Wikification

In the following, we discuss different Wikification evaluation techniques. While in many areas performance measures are defined by the task at hand and used thoroughly by most authors, this is not the case in the field of concept disambiguation or Wikification. Consequently, published results are often hard to comparable.

Following Milne and Witten (2008), Ratinov et al. used **Bag-of-Titles** (**BOT**) evaluation which compares the predicted set of titles (i.e. concepts) with the ground truth set of concepts, ignoring duplicates in either set, and further utilizes standard Precision, Recall, and F1. For discussion, we take the example from Ratinov et al. (2011). Let the ground truth be $truth =$ {("China", *People's Rep. of China*), ("Taiwan", *Taiwan*), ("Jiangsu", *Jiangsu*}, with $truth_{BOT} =$ {*People's Rep. of China*, *Taiwan*, *Jiangsu*}. Assume the system predicts {("China", *People's Rep. of China*), ("China", *History of China*), ("Taiwan", c_0), ("Jiangsu", *Jiangsu*)}, with associated BOT $pred_{BOT} =$ {*People's Rep. of China*, *History of China*, *Jiangsu*}. According to Ratinov et al., both Precision and Recall for $pred_{BOT}$ are 0.66. Consequently, the nil prediction c_0 for *Taiwan* is not counted as a false positive, since we already observe *History of China* as a false positive, with two true positives from *People's Rep. of China* and *Jiangsu* resulting in $P = 0.66$.

The first remarkable point is the ignorance of duplicate concepts which obscures both erroneous as well as correct predictions: if a concept appears 5 times in the ground truth annotation, and the disambiguation model fails to resolve it correctly, the number of false negatives is only 1 in BOT, whereas it would be 5 if all instances were considered. Analogously this holds for the number of true positives. Second, nil predictions are not counted as false positives, which renders Precision less comparable. In our implementation of BOT, we assume that the sequential input order is taken into account.

The performance measure used by Hoffart et al. (2011b) is **Mean Average Precision** (**MAP**) which is defined as $MAP = \frac{1}{m}\sum_{i=1}^{m} p@\frac{i}{m}$, where $p@\frac{i}{m}$ is the Precision at a specific Recall level. Here, the model output is ranked according to the model's confidence s, i.e. mention-concept pairs with high model confidence are ranked at leading positions, pairs with low confidence at late positions. Consider the following prediction $\{s(m_3, c_3) = 0.9, s(m_2, c_2) = 0.8, s(m_1, c_1) = 0.2\}$, that is sorted by some confidence s instead of order of appearance. If c_1 is an incorrect prediction, the associated Precision values are $\{p@1 = \frac{1}{1}, p@2 = \frac{2}{2}, p@3 = \frac{2}{3}\}$. According to the above definition, the MAP of this example is $\frac{1+1+2/3}{3} = \frac{8}{9}$. If in contrast, we followed the sequential input order, the MAP would be $\frac{0+1/2+2/3}{3} = \frac{7}{9}$. Note that the interpretation of Recall differs from that in BOT since it is related to the position in the output list and not the number of false negatives. In terms of BOT, the performance result for this example is $P = R = \frac{2}{3}$.

Assuming that incorrect predictions have in general a low confidence, MAP shuffles erroneous predictions to the end of the ranked output list. Then the sum is dominated by correct predictions (high confidence) at the top of the ranking, which are propagated through the whole list. This is of great importance, if the number of mentions in a document is especially large. In our implementation of MAP, the model confidence is represented by the SVM's prediction, i.e. the instance's hyperplane offset.

The most crucial difference between current systems is the treatment of covered and uncovered concepts. Hoffart et al. decided to ignore nil-concepts during evaluation and hence roughly 20% of the mentions. To compare our method with AIDA, we follow this restriction when applying our method on **CoNLLb** and ignore nil-concepts for this corpus as well. Using the

search coverage	no *bestFit*	*bestFit* via NGD	*bestFit* via Δ_{topics}NGD	*bestFit* via $\Delta_{\cos}$NGD	*bestFit* via $\cos(c_i, c_j)$
mention (exp.)	87.69/94.55	86.83/94.70	88.12/95.58	88.96/95.94	86.73/94.44
+type	86.10/94.32	88.79/95.60	88.22/95.96	89.53/95.98	88.46/95.73
+context	86.43/94.30	89.50/96.10	89.30/96.60	89.95/96.54	89.20/96.45
+topics	87.59/95.16	89.47/96.46	89.50/96.48	**89.95**/96.81	89.60/96.67
⌀ cross coherence		0.381	0.179	0.104	0.215

Table 2: $F1_{BOT}$/MAP of our system on **MSNBC** for different configurations (all values in %).

search coverage	no *bestFit*	*bestFit* via NGD	*bestFit* via Δ_{topics}NGD	*bestFit* via $\Delta_{\cos}$NGD	*bestFit* via $\cos(c_i, c_j)$
mention (exp.)	84.46/92.74	86.18/93.23	87.91/94.02	87.02/93.01	86.70/92.95
+type	83.30/91.62	87.23/93.43	87.75/93.51	87.18/93.21	87.23/93.10
+context	86.49/93.28	86.76/92.98	88.40/93.80	88.85/94.12	87.75/93.54
+topics	86.50/91.16	86.97/91.25	88.44/94.67	**89.01**/94.33	88.24/93.68
⌀ cross coherence		0.354	0.139	0.096	0.211

Table 3: $F1_{BOT}$/MAP of our system on **ACE** for different configurations (all values in %).

AIDA online version that treats nil-concepts, we can evaluate this system on the other corpora. Kulkarni et al. also used a different evaluation scheme (KUL_{F1}) that is comparable to BOT but takes incorrect nil predictions into account. For more details, we refer to the respective paper.

7 Evaluation

As most alternative approaches rely on different versions of large-scale knowledge bases, it is practically not feasible to re-implement every competitor system. GLOW is publicly available, but we decided against using it, since we could not reproduce the results published in (Ratinov et al., 2011) and assumed that there was a crucial difference we could not solve[7]. Hence we compare our system to the figures reported by Ratinov et al. (2011) both for GLOW as well as the M&W system (Milne and Witten, 2008). For comparison with AIDA, we used the online interface $AIDA_{web}$ which was kindly provided to us by the authors[8] and run on all corpora. To give a unified view, we report the results for our system in all performance measures outlined in the previous section.

We ran initial experiments on all corpora to evaluate the effect of the mention expansion described in Sec. 3.2 and found that it increased $F1_{BOT}$ on all corpora by about 2%, when only the mention's name was considered in the search. We report the effect of different *search coverages* and show that in many cases results can be improved when we extend searches relying only the expanded mention name by additional type and context information. For all coverages, we show the effect of *bestFit* configurations: weighting NGD with Δ_{topics}, $\Delta_{\cos}$ and replacing NGD by the cosine distance over article texts. The influence of the topic feature computed from the Hellinger distance (Eq. 5) of $(topics(m_i), topics(c))$ is reported as well since it is computationally the most expensive SVM feature.

Tables 2 to 6 show the results obtained for the different configurations of our system. The corresponding performance figures of of GLOW, M&W and $AIDA_{web}$ are given in the text. For **MSNBC** (Tab. 2), the best configuration of our system (complete coverage, topics, *bestFit* candidate via $\Delta_{\cos}NGD$) achieves a $F1_{BOT}$ of 89.95%, which is 15% higher than that of GLOW (74.88%) and 20% higher than for M&W (68.49%). Also, the MAP of our system is with 96.81% more than 25% higher than that of $AIDA_{web}$ (69.52%). We found that the same configuration

[7]Thanks to Lev-Arie Ratinov for his useful comments on this.

[8]We use the most current version of July 30th, 2012.

search coverage	no *bestFit*	*bestFit* via NGD	*bestFit* via Δ_{topics}NGD	*bestFit* via $\Delta_{\cos}$NGD	*bestFit* via $\cos(c_i, c_j)$
mention (exp.)	84.77/94.50	84.71/94.83	85.07/94.47	85.45/94.65	84.53/94.65
+type	84.41/94.69	84.93/94.87	85.61/95.02	85.43/94.73	84.41/94.57
+context	84.81/92.92	84.50/93.83	84.19/94.10	84.59/93.70	82.95/93.36
+topics	**86.81**/91.97	84.46/91.97	84.33/93.87	84.94/93.53	83.20/93.55
⌀ cross coherence		0.31	0.119	0.07	0.161

Table 4: $F1_{BOT}$/MAP of our system on **AQUAINT** for different configurations (all values in %).

search coverage	no *bestFit*	*bestFit* via NGD	*bestFit* via Δ_{topics}NGD	*bestFit* via $\Delta_{\cos}$NGD	*bestFit* via $\cos(c_i, c_j)$
mention (exp.)	84.89	85.03	85.71	85.75	85.12
+type	85.36	86.72	88.13	87.26	87.44
+context	86.04	88.23	89.25	88.70	88.80
+topics	87.56	88.65	**89.32**	89.13	89.12
⌀ cross coherence		0.402	0.208	0.134	0.262

Table 5: MAP of our system on **CoNLLb** for different configurations (all values in %).

also yields the best result on **ACE** (Tab. 3). On this corpus, our system achieves a $F1_{BOT}$ of 89.01%, which outperforms GLOW (77.25%) and M&W (72.67%) by more than 12%. Also, the MAP of our system is with 94.33% about 9% higher than that of $AIDA_{web}$ (86.14%).
For **AQUAINT** (Tab. 4), the best configuration of our system is complete search coverage and using the topic feature in SVM ranking. Here, *bestFit* candidate generation did not increase performance. We argue that this is due to the rather low average cross coherence over the ground truth concepts. Without the usage of collective information, our system achieves a $F1_{BOT}$ of 86.81%, which outperforms GLOW (83.94%) and M&W (83.61%) by 3%. Note that even the slightly worse results using collective search are higher. Also, the MAP of our system is with 91.97% about 30% higher than that of $AIDA_{web}$ (58.61%). For all of the above corpora, we found that not using the SVM for candidate re-ranking and nil-concept detection reduces the $F1_{BOT}$ of our system between 5 and 10%, which shows the usefulness of a supervised classifier.
For **CoNLLb** (Tab. 5), the best configuration of our system is the complete search coverage, the topic feature in SVM ranking, and *bestFit* candidate generation via Δ_{topics}NGD. This corpus has the highest avg. cross coherence over the ground truth concepts. Our system achieves a MAP of 89.32% (with corresponding $F1_{BOT}$= 82.16%), which is only slightly better than the figures published for AIDA (89.05%) but about 4% higher than that of $AIDA_{web}$ (85.66%). Without SVM application, the MAP of our system would be reduced to 86.70%. This indicates the necessity of features that are not graspable by **Wiki** but available in SVM candidate re-ranking.
We found that on **IITB** (Tab. 6), the best results can be achieved when we use only name and type based queries in combination with *bestFit* candidate selection via Δ_{topics}NGD. Although we found that this corpus has the lowest avg. cross coherence over the ground truth concepts, the collective search increases performance. We argue that this result is due to the very high number of mentions per document, which has a diminishing effect on the avg. cross coherence. Our system achieves a KUL_{F1} of 75.26%, which is 5% better than the result published by Kulkarni et al. (2009) (69.69%). Note that the performance of $AIDA_{web}$ on **IITB** is only MAP=43.62%, whereas the corresponding MAP of our system is 90%. We are aware that the performance reported by Han et al. (2011) is with KUL_{F1}=78.95% about 4% higher than that of our system. Still, even though our system was not tuned on specific data sets, we achieve a high performance on all of the 5 different benchmark corpora. We argue that this makes our system the most stable compared to other approaches both in terms of generalizability and applicability.

search coverage	no *bestFit*	*bestFit* via NGD	*bestFit* via Δ_{topics}NGD	*bestFit* via $\Delta_{\cos}$NGD	*bestFit* via $\cos(c_i, c_j)$
mention (exp.)	73.81/89.92	74.74/89.91	**75.26/89.95**	74.68/89.67	73.89/89.32
+type	73.96/89.93	74.90/89.94	**75.10/90.00**	74.85/89.68	74.08/89.40
+context	72.57/88.01	69.07/85.69	69.81/85.59	68.54/86.23	69.29/86.14
+topics	71.10/87.26	68.74/85.41	69.41/86.31	68.35/86.10	69.13/85.83
⌀ cross coherence		0.224	0.087	0.041	0.112

Table 6: KUL_{F1}/MAP on **IITB** for different method configurations (all values in %).

To summarize, we observe for all corpora a positive correlation between the avg. cross coherence of ground truth concepts and the effect of the collective search. The influence of confidence sorting in MAP becomes obvious on **MSNBC** and **ACE**: while F1_{BOT} differs only by 1%, the associated MAP value can differ by 4%. This can be the case, when the average number of mentions per document differs (8.5 on **ACE** and 37.75 on **MSNBC**).

The concept-mention probability $p(c|m)$ is a very strong feature as often the name is sufficient for disambiguation. This is most obvious on **IITB**, where the incorporation of context features even decreased performance. We also found that this prior-like attribute can mislead the SVM re-ranking on **CoNLLb**. This corpus contains many sport statistics that, for instance, mention countries participating in a match. As an example, even though the *bestFit Japan National Football Team* is correct due to high cross coherence, the SVM re-ranked the output to *Japan*, since the concept-mention probability $p(\textit{Japan}|\text{"Japan"}) = 0.97$ is much higher than $p(\textit{Japan National Football Team}|\text{"Japan"}) = 0.0063$. While Hoffart et al. (2011b) thus did not always use this feature, we could not find an appropriate threshold for our system.

Especially for **CoNLLb** we found that our system suffered from annotation scheme differences. While our system links mentions like "British" to concepts such as *English language* or *British people*, the ground truth concept in **CoNLLb** is always *United Kingdom*. An investigation showed that these annotations are often correct but in general depend on the gusto of the annotator. Also, we observed for **IITB** that many ground truth concepts are disambiguation pages. These are not contained in our index and thus treated as $\tilde{c}_0$. For example, we observed a document on sports mentioning the word "fitness" which was linked to the disambiguation page *Fitness* by the **IITB** annotators. While our system predicted the suitable concept *Physical Fitness*, this was still treated as an error since we relinked the disambiguation page *Fitness* to $\tilde{c}_0$.

Conclusion

In this paper we described a novel algorithm for concept disambiguation through concept assignment to Wikipedia articles. We exploit the coherence of Wikipedia concepts and take into account a variety of features to perform the assignment. The algorithm also estimates if a concept is not covered by Wikipedia. It turned out that the collective search is more efficient, if the average coherence between the concepts is higher.

We analyzed different evaluation criteria and discussed their relative strengths and weaknesses. We evaluated various configurations of our approach on five benchmark corpora and compared the results to four competitor systems. For some benchmark data sets our system is dramatically better than the other approaches, while for other corpora the differences are not so pronounced. We observed that these benchmark corpora are not error free, which can limit their usability. Except for one case our system always has better performance figures than the competitor systems and therefore can be considered as a stable alternative ready for practical application. In future work we will consider the assignment of concepts not described in Wikipedia.

References

Blei, D. and Lafferty, J. (2009). Topic models. In Srivastava, A. and Saham, M., editors, *Text Mining: Theory and Applications*. Taylor and Francis.

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

Cucerzan, S. (2007). Large-scale named entity disambiguation based on wikipedia data. In *Proc. 2007 Joint Conference on EMNLP and CoNLL*, pages 708–716.

Han, X., Sun, L., and Zhao, J. (2011). Collective entity linking in web text: a graph-based method. In *Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval (SIGIR '11)*, pages 765–774. ACM.

Hatcher, E., Gospodnetic, O., and McCandless, M. (2010). *Lucene in Action*. Manning, 2nd revised edition edition.

Hoffart, J., Suchanek, F. M., Berberich, K., Kelham, E. L., de Melo, G., and Weikum, G. (2011a). Yago2: Exploring and querying world knowledge in time, space, context, and many languages. In *Demo paper in the proceedings of the 20th International World Wide Web Conference (WWW 2011)*.

Hoffart, J., Yosef, M. A., Bordino, I., Fürstenau, H., Pinkal, M., Spaniol, M., Taneva, B., Thater, S., and Weikum, G. (2011b). Robust disambiguation of named entities in text. In *Conference on Empirical Methods in Natural Language Processing*, pages 782–792.

Kulkarni, S., Singh, A., Ramakrishnan, G., and Chakrabarti, S. (2009). Collective annotation of wikipedia entities in web text. In *15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*.

Milne, D. N. and Witten, I. H. (2008). Learning to link with wikipedia. In *Proceedings of the ACM Conference on Information and Knowledge Management (CIKM'2008)*, pages 509–518.

Pilz, A. and Paass, G. (2011). From names to entities using thematic context distance. In *Proceedings of 20th ACM Conference on Information and Knowledge Management (CIKM)*, Glasgow, UK.

Ratinov, L.-A., Roth, D., Downey, D., and Anderson, M. (2011). Local and global algorithms for disambiguation to wikipedia. In *ACL*, pages 1375–1384.

Schuetze, H. (1998). Automatic word sense discrimination. *Computational Linguistics - Special issue on word sense disambiguation*, 24:97–123.

Song, D. and Heflin, J. (2011). Automatically generating data linkages using a domain-independent candidate selection approach. In *Proceedings of International Semantic Web Conference (ISWC)*, pages 649–664.

Suchanek, F., Kasneci, G., and Weikum, G. (2008). Yago - a large ontology from wikipedia and wordnet. *Elsevier Journal of Web Semantics*, 6(3):203–217.

Who's (Really) the Boss?
Perception of Situational Power in Written Interactions

Vinodkumar Prabhakaran[1] *Owen Rambow*[2] *Mona Diab*[2]
(1) Department of Computer Science, Columbia University, New York, NY 10027, USA
(2) Center for Computational Learning Systems, Columbia University, New York, NY 10115, USA
vinod@cs.columbia.edu, {rambow,mdiab}@ccls.columbia.edu

Abstract
We study the perception of situational power in written dialogs in the context of organizational emails and contrast it to the power attributed by organizational hierarchy. We analyze various correlates of the perception of power in the dialog structure and language use by participants in the dialog. We also present an SVM-based machine learning system using dialog structure and lexical features to predict persons with situational power in a given communication thread.

Keywords: Computational Sociolinguistics, Social Networks, Power Analysis, Dialog Acts, Dialog.

Title and Abstract in German

Wer ist (wirklich) der Chef?
Situative Macht in schriftlichen Dialogen

Wir untersuchen, wie situative Macht in schriftlichen Dialogen wahrgenommen wird. Situative Macht ist Macht, die nicht beständig ist, sondern nur zielbedingt während einer (vielleicht längeren) Interaktion existiert. Unsere Studie beruht auf Geschäftsemails. Wir kontrastieren situative Macht mit der Macht, die durch die organisatorische Hierarchie entsteht. Wir identifizieren verschiedene Korrelate der Wahrnehmung situativer Macht in der Dialogstruktur und im Sprachgebrauch der Dialogteilnehmer. Wir stellen ausserdem einen SVM-basierten maschinellen Lernalgorithmus vor, der Dialogstruktur und Wörter in den Emails benutzt, um Dialogteilnehmer mit situativer Macht zu identifizieren.

Keywords in German: Komputationalle Soziolinguistik, Soziale Netzwerke, Macht, Dialogakte, Dialog.

Proceedings of COLING 2012: Technical Papers, pages 2259–2274,
COLING 2012, Mumbai, December 2012.

1 Introduction

Within an interaction, there is often a power differential between the interactants. This differential is often drawn from a static source external to the interaction, such as a formal or informal power structure or hierarchy. Most computational studies (Creamer et al., 2009; Bramsen et al., 2011; Gilbert, 2012) that analyze power within interactions have used such an external power structure (namely, a corporate hierarchy) as the definition of the power differential. However, the power differential may also be dynamic and specific to the situation of the interaction. We define a person to have **situational power** if there is another person such that he or she has power or authority over the first person in the context of a situation or task. Such situational power may not always align with the external power structures, if one exists. For example, a Human Resources department employee within an organization will have power over an office manager when the interaction is about enforcing some HR policy. But the direction of power will be reversed if the interaction is about allocating office space for a new hire. Neither of these directions of power may be captured in the organizational hierarchy chart. In some cases, the situational power may even be in the opposite direction to the power relation reflected in the hierarchy — e.g., a subordinate organizing an office event.

Although power is a difficult concept to define, it is often recognizable from an interaction. As we show in this paper, one of the primary ways power is recognized in dialog is by the manner in which people participate in the dialog. Power relations sometimes constrain how one behaves when engaging in dialog; in some other cases, they enable one to constrain another person's behavior. And in some cases, the dialog behavior becomes a tool to express and even pursue power. By dialog behavior, we mean the choices a discourse participant makes while engaging in dialog. It includes choices with respect to the message content, like lexical choice or degree of politeness. It also includes choices participants make in terms of dialog structure, such as the choice of when to participate with how much and what sort of contribution, the number of questions to ask and which of those questions to answer. These manifestations may differ depending on whether the power differential is entirely hierarchical or situational or a mix of both - hierarchical power may be inert in a particular interaction, but situational power is a rather active form of power.

In this paper, we focus on situational power, how an outsider perceives it and what attributes of the interaction contributes to that perception. More specifically, we focus on why a dialog participant is perceived to have situational power within an interaction. Another related problem is whether a participant is perceived to have situational power over a *specified person* or not. We do not address it in this paper. We first analyze the notion of situational power in detail and find that its perception is subjective. We then define a set of features describing the choices participants make, including the dialog structure and the verbosity of dialog participants, and study how they correlate with the perception of situational power. We build a system to detect participants with situational power using a supervised machine learning approach. The main findings of this study are: a) situational power is a very different form of power than the power ascribed by hierarchy; b) the perception of situational power correlates with dialog structure as well as content-based features. The contributions of this paper also include an automatic situational power tagger based on dialog structure and content features. This study is conducted on email threads; however, we do not use any aspects of written dialog that are specific to email. Hence, we expect the methodology we use and the insights we gain to be applicable to other genres of online social media such as online discussion forums.

The rest of the paper is structured as follows: Section 2 presents related work in this field. Section 3 presents the data and describes various annotations present in the data, including situational power

annotations. Section 4 compares the perception of situational power with hierarchical power. Section 5 defines the problem formally and Section 6 describes all dialog structure and verbosity features we used. Section 7 presents statistical significance measures of these features with respect to persons perceived to have situational power. Section 8 presents the machine learning experiments conducted on the data and discusses results. We finally conclude and discuss future work.

2 Related Work

Most definitions of power in the sociology literature — e.g., (Dahl, 1957; Emerson, 1962; Bierstedt, 1950) — include "an element indicating that power is the capability of one social actor to overcome resistance in achieving a desired objective or result" (Pfeffer, 1981). The five bases of power proposed by French and Raven (1959) (Coercive, Reward, Legitimate, Referent, and Expert) and their extensions are widely used in sociology to study power. Wartenberg (1990) makes the distinction between two notions of power: power-over and power-to. Power-over refers to hierarchical relationships between interactants, while power-to refers to the ability to exercise power an interactant possesses (maybe temporarily) and can use within the interaction. Our notion of situational power roughly corresponds to Wartenberg's notion of power-to, while the power attributed to an interactant by an organizational hierarchy can be considered an instance of power-over. Both can be considered special cases of French and Raven's notion of legitimate power.

It has long been established that there is a correlation between dialog behavior of a discourse participant and how influential he or she is perceived to be by the other discourse participants (Bales et al., 1951; Scherer, 1979; Brook and Ng, 1986; Ng et al., 1993, 1995). Specifically, factors such as frequency of contribution, proportion of turns, and number of successful interruptions have been identified as being important indicators of influence. Reid and Ng (2000) explain this correlation by saying that "conversational turns function as a resource for establishing influence": discourse participants can manipulate the dialog structure in order to gain influence. This echoes a starker formulation by Bales (1970): "To take up time speaking in a small group is to exercise power over the other members for at least the duration of the time taken, regardless of the content." Simply successfully claiming the conversational floor represents a feat of power. The previous work just discussed was conducted entirely on spoken dialog. In this paper, we show that the core insight — conversation is a resource for power — carries over to written dialog. This means that we can predict powerful people from studying the discourse structure. However, some of the characteristics of spoken dialog do not carry over straightforwardly to written dialog, most prominently among them the important issue of interruptions: there is no interruption in written dialog. Our work draws on findings for spoken dialog, looking at correlates for written dialog.

We now turn to the computational literature. Several studies have used Social Network Analysis (Diesner and Carley, 2005; Shetty and Adibi, 2005; Creamer et al., 2009) or email traffic patterns (Namata et al., 2007) for extracting social relations from online communication. These studies use only meta-data about messages: who sent a message to whom when. For example, Creamer et al. (2009) find that the response time is an indicator of hierarchical relations; however, they calculate the response time based only on the meta-data, and do not have access to information such as thread structure or message content, which would actually verify that the second email is in fact a response to the first. In fact, using NLP to deduce social relations from online communication is relatively a new area which has only recently become an active area of research (Bramsen et al., 2011; Gilbert, 2012; Danescu-Niculescu-Mizil et al., 2012).

Bramsen et al. (2011) and Gilbert (2012) address the same problem we are trying to solve: identifying social power relationships from online written communication. They both also use the Enron email

corpus for their experiments. Using knowledge of the actual organizational structure, Bramsen et al. (2011) create two sets of messages: messages sent from a superior to a subordinate, and *vice versa*. Their task is to determine the direction of power (since all their data, by design in the construction of the corpus, has a power relationship). They approach the task as a text classification problem and build a classifier to determine whether the set of all emails (regardless of thread) between two participants is an instance of up-speak or down-speak. Similarly, Gilbert (2012) considers a message to be *upward* only when every recipient of that message outranks the sender. Any message that is not an *upward* message is labeled *non-upward*. This formulation is slightly different from that of (Bramsen et al., 2011) which considers only those messages that have a power relationship upward or downward. Gilbert (2012) extracts a list of phrases that signal *upward* messages using penalized logistic regression model. While the objectives of both these studies and our work are the same, there are major differences. Firstly, our focus is on situational power and how it is perceived by an outsider, while they focus on power from the organizational hierarchy. We show in Section 4 that the perception of situational power is not aligned with the organizational hierarchy. Secondly, our data unit is a naturally occurring thread, not data units assembled by the researchers, and a thread may or may not include a person who has power. Also, we focus on the structure of the dialog (which we can do since our unit is a thread, as opposed to a single message or an arbitrary aggregation of single messages).

Danescu-Niculescu-Mizil et al. (2012) study the notion of language coordination — a metric that measures the extent to which a discourse participant adopts another's language — in relation with various social attributes such as power, gender, etc. They perform their study on Wikipedia discussion forums and Supreme Court hearings. They also look into situational power; however they define situational power in terms of the dependence between interactants: "x may have power over y in a given situation because y needs something that x can choose to provide or not". They model this dependence "using the exchange-theoretic principle that the need to convince someone who disagrees with you creates a form of dependence." We adopt a broader definition of situational power based on context and perception. Strzalkowski et al. (2010) are also interested in power in written dialog. However, their work concentrates on lower-level constructs called *Language Uses* which will be used to predict power in subsequent work. They model power using notions of topic switching, exploiting mainly complex lexical features. Peterson et al. (2011) focus on formality in Enron email messages and relates it to social distance and power.

3 Data

3.1 Corpus

We use the email corpus presented in (Prabhakaran et al., 2012a) which contains manual annotations for various types of power relations between participants. In this study, we focus only on Situational Power (SP) and leave the other types of power for future work. The corpus contains 122 email threads with a total of 360 messages and 20,740 word tokens. This set of email threads is chosen from a version of the Enron email corpus with some missing messages restored from other emails in which they were quoted (Yeh and Harnly, 2006). Most emails are concerned with exchanging information, scheduling meetings, and solving problems, but there are also purely social emails. Table 1 presents some statistics on participants and messages in the corpus. We define an active participant of a given thread as someone who has sent at least one email message in the thread.

Apart from the thread level annotations for different types of power, the corpus also contains utterance level annotations for overt displays of power. The same corpus has been previously annotated with dialog act annotations (Hu et al., 2009). We utilize these annotations in our study

Statistic	Count / Mean (SD)
Number of email threads	122
Number of participants	1033
Ave. Participants / thread	8.47 (13.82)
Number of active participants	221
Ave. Active participants / thread	1.81 (0.73)
Number of messages	360
Ave. Messages / thread	2.95 (2.24)
Ave. Messages / active participant	1.45 (1.01)
Number of word tokens	20,740
Situational Power (SP)	81

Table 1: Corpus statistics

and will describe them in more detail in the following sections. We give an example thread and corresponding situational power annotations in Table 2. The example shows annotations for overt display of power (ODP) which will be explained in Section 3.1.2. The email body also contains dialog act annotations which will be explained in Section 3.1.3.

3.1.1 Situational Power Annotations

Person_1 is said to have situational power over person_2 if person_1 has power or authority to direct and/or approve person_2's actions in the current situation or while a particular task is being performed, based on the communication in the current thread. Situational power is independent of organizational hierarchy: person_1 with situational power may or may not be above person_2 in the organizational hierarchy (or there may be no organizational hierarchy at all). For more details on the situational power annotations, see (Prabhakaran et al., 2012a), where we explain these annotations in more detail with examples and describe instructions given to the annotator.

In our example thread, the annotator judged *William* to be possessing situational power over *Barry* and *Barry* over *Stephanie*: in both cases, a task is assigned to another person, even if the language used is more direct in the case of *Barry* delegating his task to *Stephanie*.

3.1.2 Overt Display of Power (ODP) annotations

An utterer can choose linguistic forms in her utterance to signal that she is imposing constraints on the addressee's choice of how to respond, which go beyond those defined by the standard set of dialog acts. For example, if the boss's email is "Please come to my office right now", and the addressee declines, he is clearly not adhering to the constraints the boss has signaled, though he is adhering to the general constraints of cooperative dialog by responding to the request for action. This roughly captures the "restriction of an interactant's action-environment", suggested as one of the key elements to identify exercise of power in interactions by Wartenberg (1990). An utterance is defined to have an **overt display of power (ODP)** if it is interpreted as creating additional constraints on the response beyond those imposed by the general dialog act. Syntactically, an ODP can be an imperative, a question, or a declarative sentence. The presence of an ODP does not presuppose that the utterer actually possess social power: the utterer could be attempting to gain power. Out of the 1734 utterances in our corpus, 86 were annotated to have an expression of ODP. In our example thread, utterances M2.2 and M2.6 were labeled as instances of ODP.

From: William S Bradford To: Barry Tycholiz CC: Michael Tribolet Subject: Gas Inventories Time: 2001-09-24 10:45:00 ——————————————— M1.1. Barry, [Conventional] M1.2. Let me know if you have any time to review. [Inform] M1.3. Bill [Conventional] From: Barry Tycholiz To: Stephanie Miller Subject: Gas Inventories Time: 2001-09-24 11:11:05 ——————————————— M2.1. Steph, [Conventional] M2.2. further to our discussion, Pls review. [Request-Action] Flink2.2 M2.3. I took a quick look at the locations and most appear to be East based. [Inform] M2.4. You might want to use an analyst to figure this out. [Inform] M2.5. Also, they have valued the inventories off of the Nymex only (or so it appears) and I would have to believe that the value of these molecules is materially different than this. [Inform] M2.6. Pls review and let's discuss asap. [Request-Action] Flink2.6 M2.7. BT [Conventional]	
Situational Power	William S Bradford –> Barry Tycholiz
	Barry Tycholiz –> Stephanie Miller
Overt Display of Power	M2.2, M2.6

Table 2: Example thread and annotations; note that the dialog act for M1.2 appears to be incorrectly labeled as Inform, instead of Request-Action or perhaps Request-Information (we did not change any dialog act labels)

For a further discussion of the annotation of ODP, see (Prabhakaran et al., 2012b), where we further define the notion of ODP, give inter-annotator agreement numbers, and present initial work on building an automatic classifier for ODP.

3.1.3 Dialog Act annotations

The corpus we used also contains manual dialog act annotations as described in Hu et al. (2009). We use these annotations to model the dialog structure of the communication thread. Each message in the thread is segmented into Dialog Functional Units (DFUs). A DFU is a contiguous subset of a turn (i.e., in our corpus, of an email message) which has a coherent communicative intention. Each DFU is assigned a Dialog Act (DA) label which is one of the following: **Request for Action**, **Request for Information**, **Inform**, **Inform-Offline**,[1] **Conventional**, and **Commit**.

In addition, DFUs are interlinked by three types of links to reflect the dialog structure. These links capture the patterns of local alternation between an initiating dialog act and a responding one. A **forward link** (Flink) is the analog of a "first pair-part" of an adjacency pair, and is similarly restricted to specific speech act types. All Request-Information and Request-Action DFUs are assigned Flinks. The responses to such requests are assigned a **backward link** (Blink). If an utterance can be interpreted as a response to a preceding DFU, it gets a Blink even where the preceding DFU has no Flink. The preceding DFU taken to be the "first pair-part" of the link is assigned a **secondary forward** link (SFlink).

3.2 How Subjective is the Perception of Situational Power?

In this section, we investigate how subjective the perception of situational power is. We performed an independent study of annotator perceptions on a subset of 47 threads from the corpus. We trained two additional annotators — AnnA and AnnB — using the same annotation manual described in (Prabhakaran et al., 2012a) and compared the annotations they produced for situational power on the selected threads. Both AnnA and AnnB are undergraduate students, one from the Arts Department and the other from the Engineering Department.

The cognitive process behind labeling a participant to have situational power is not a binary decision the annotator makes for each participant. Annotators read the entire thread before performing the annotations. They are also asked to provide, in free-form English, a short "power narrative" which describes their perception of the overall power structure among the discourse participants of that thread. Annotators build a fairly consistent mental image of a power narrative — an outline of the power structure between the participants — based on various indicators from across the thread. Their annotations about situational power are based on this power narrative. Evaluating agreement on such a task is not trivial. For the purposes of this study, we port this task into a binary decision task of identifying whether participant X has situational power or not. There were 289 participants in the selected 47 threads. AnnA found 19 of these participants to have situational power while AnnB found only 13 to have situational power, 8 of which were also found by AnnA. We obtained a κ value of 0.472 which is only a moderate agreement. The fact that we don't obtain a higher agreement could be due to many reasons. Firstly, in porting the task to a binary labeling task, we are unnecessarily penalizing the annotators by introducing instances to represent judgments that the annotator never actually made. For example, if an email invite to a party was sent to 50 recipients, the annotator will not have considered each single recipient individually and made a choice about

[1] Sometimes, the Inform act refers to a previous act of communication that did not happen in the email thread itself. Such cases are marked as Offline.

him or her. However, these 50 recipients will be added as data points in our κ calculation, thereby increasing the expected agreement and decreasing the κ value. Another reason could be just that the task by itself is subjective. The indicators that are noticed by each annotator may underspecify how they can be interpreted in the power narrative (and subsequently the situational power annotation). The annotator's choices will then vary depending on the annotator's familiarity with corporate culture, or with other individual characteristic of the annotators.

We investigated the annotations further to confirm this. We found that there were many instances where different valid power narratives could be built based on the same email thread. For example, in our example thread, the message from Bill (first message) could be interpreted in isolation as a request from a peer or even a subordinate. However, if you take into consideration that Barry delegated the task to Stephanie upon receiving the message from Bill, the first message could be considered as Bill assigning a task to Barry. Either judgment is valid depending on the power narrative that one builds around the interaction within the thread. The original annotation in (Prabhakaran et al., 2012a) adopted the latter narrative whereas both AnnA and AnnB adopted the former. In our investigation of cases where AnnA and AnnB disagreed, we found many cases where both scenarios (person *X* having power and not having power) are plausible based on the annotators' power narrative.

The original annotations that were in the corpus are the perception of one particular annotator. We obtained similar moderate (>0.3) agreement between AnnA and AnnB and the original annotations for the subset of threads that were triply annotated. The moderate agreement suggests that there must be some core indicators of situational power that we could obtain by combining multiple perceptions. We leave that to future work. For the rest of this paper, we rely on the original annotations for the perception we are modeling. In Section 8.3, we explicitly address the issue of whether the original annotator's perceptions are internally consistent.

4 Are they really the bosses?

We explore how the perception of situational power compares with the organizational hierarchy. For this purpose, we utilize the gold organizational hierarchy for Enron released by (Agarwal et al., 2012). It contains relations between 1,518 employees, and 13,724 dominance pairs (pairs of employees such that the first dominates the second in the hierarchy, not necessarily immediately). We labeled a participant to have hierarchical power within a thread if there exist a dominance pair in the gold hierarchy where he/she is dominating over any other participant in the same thread.

According to the gold hierarchy, 113 out of the 1033 participants in our corpus have hierarchical power within the interaction. But only 12 of them (10.6%) were perceived to have situational power by our annotator. In other words, bosses act bossy rather rarely. Also, a total of 107 participants were judged to have situational power. The 12 of those who also had hierarchical power amounts to only 11.2% of them. In other words, you don't have to be the boss to be bossy.

The above findings are particularly important since most previous computational approaches have concentrated on modeling power purely in hierarchical terms. These findings show the importance of thinking beyond hierarchy and about other types of power.

5 Problem Definition and Approach

Now we move on to finding correlates in the interaction that could help predict participants with situational power. Given a communication thread $\mathcal{T}$ and an active participant $\mathcal{X}$, we would like

to predict whether $\mathscr{X}$ has situational power over some person $\mathscr{Y}$ in the thread.[2] In this paper, we restrict ourselves to features relative to messages sent by the participant $\mathscr{X}$. Hence, we consider each active participants in the thread as a data point and extract features with respect to them. There were 221 active participants in our corpus out of which 81 were annotated to have situational power. Our approach is to build a binary classifier predicting whether or not $\mathscr{X}$ has situational power based on features with respect to $\mathscr{X}$ in the context of the given thread $\mathscr{T}$.

6 Feature Sets

In this section, we describe six sets of features we use to capture the way interactants participate in dialog. The first four sets of features — dialog act (DAP), dialog link (DLC), positional (PST) features and verbosity (VRB) — relate to the whole dialog and its structure, whereas lexical (LEX) features and overt display of power (ODP) are features related to the form and content of individual messages. PST, VRB, and LEX are readily derivable from the data, while we use the gold annotation for DAP, DLC, and OSP.

6.1 Dialog act features - DAP

We have six features: **ReqAction**, **ReqInform**, **Inform**, **InformOffline**, **Conventional**, and **Commit**, denoting the percentage of each of these dialog act labels aggregated over all messages sent by the participant within the thread.

6.2 Dialog structure link features - DLC

We use counts of various types of dialog structure links between DFUs as features. We use absolute counts here rather than relative counts since there is no obvious maximal number of links against which to compare. **Flink**, **SFlink** and **Blink** denote the total number of Flinks, SFlinks and Blinks in messages sent by the participant. **Clink** denotes the number of Blinks by other people connected back to DFUs in messages sent by the participant. This includes both Flinks and SFlinks that are connected. **Dlink** denotes the number of Flinks by the participant that were not connected back via Blinks by other people ("dangling Flinks"). These are requests with no responses. **DlinkRatio** denotes Dlinks as a percentage of the number of Flinks by the participant.

6.3 Positional features - PST

We use features that denote the placement of the participant's messages relative to the thread. **Initiator** is a binary feature denoting whether the participant was the initiator of the thread. We used two other features: **FirstMsg** and **LastMsg**, to denote the position where the participant sent his/her first and last message, normalized by the total number of messages in the thread.

6.4 Verbosity features - VRB

We use features denoting how verbose the participant is within the thread. **MsgCount** denotes the number of messages sent by the participant. **MsgRatio** denotes the proportion of messages sent by the participant compared to the total number of messages in the thread. **TokenCount** denotes the number of tokens used by the participant. **TokenRatio** denotes the proportion of tokens used by the participant compared to the total number of tokens in the thread. **TokenPerMsg** denotes the average number of tokens per messages sent by the participant.

[2] The related problem to predict if person $\mathscr{X}$ has situational power over a specified person $\mathscr{Y}$ is not addressed in this paper.

Set	Features	SP
DAP	ReqAction	$\mathbf{0.07/0.01_{0.01}}$
	ReqInform	$0.10/0.12_{0.70}$
	Inform	$0.56/0.63_{0.10}$
	InformOffline	$0.003/0.005_{0.62}$
	Conventional	$0.25/0.23_{0.35}$
	Commit	$0.001/0.003_{0.51}$
DLC	Flink	$\mathbf{0.98/0.59_{0.03}}$
	SFlink	$\mathbf{0.49/0.24_{0.02}}$
	Blink	$0.72/0.59_{0.40}$
	Clink	$\mathbf{0.83/0.44_{7.1E-3}}$
	Dlink	$0.64/0.39_{0.08}$
	DlinkRatio	$\mathbf{0.33/0.21_{0.05}}$
PST	Initiator	$\mathbf{0.68/0.48_{3.3E-3}}$
	FirstMsg	$\mathbf{0.13/0.24_{1.1E-3}}$
	LastMsg	$0.41/0.36_{0.21}$
VRB	MsgCount	$\mathbf{1.68/1.32_{0.03}}$
	MsgRatio	$0.54/0.50_{0.18}$
	TokenCount	$\mathbf{113.04/74.19_{0.02}}$
	TokenRatio	$\mathbf{0.62/0.47_{2.1E-3}}$
	TokensPerMsg	$73.22/54.76_{0.07}$
ODP	ODPCount	$\mathbf{0.78/0.14_{6.0E-8}}$

Table 3: Statistical significance measures; values with $p \leq 0.05$ are boldfaced

6.5 Lexical features - LEX

The LEX feature set contains lexical features extracted from the content of messages sent by the participant. We aggregated all messages sent by the participant in the thread and extracted ngram counts for word lemmas. We experimented with **Unigram** counts, **Bigram** counts and a combination of both. We found that unigram counts performed better than the other two. Higher order ngrams were found to decrease the performance of the system.

6.6 Overt Display of power - ODP

We used a feature **ODPCount** to denote the number of instances of ODP in messages sent by the participant. For this study, we used the gold ODP tags (as we use the gold dialog annotations).

7 Statistical Significance Study

In this section, we present the statistical significance study of dialog features with respect to persons with situational power. We consider two populations of people who participated in the dialog – $\mathscr{P}_p$, those judged to have situational power and $\mathscr{P}_n$, those not judged to have situational power. Then,

for each feature, we performed a two-sample, two-tailed t-test comparing means of feature values of $\mathscr{P}_p$ and $\mathscr{P}_n$. Table 3 presents means of each feature value for both populations $\mathscr{P}_p$ and $\mathscr{P}_n$ (as mean($\mathscr{P}_p$)/mean($\mathscr{P}_n$)) along with the p-value associated with the t-test as the subscript. For p-values less than 0.05, we reject the null hypothesis and consider the feature to be statistically significant. We have highlighted the statistically significant features in Table 3.

7.1 Significant Features

We find many features to be statistically significant, which suggests that situational power is reflected in the dialog structure and content of messages. For example, persons with situational power tend to utter requests for action (ReqAction) significantly more than those without. They have significantly more connected links (Clink, SFlink); but the ratio of dangling links is also significantly higher for them, probably because they issue significantly more forward links (Flink). They also tend to be the initiators of the thread (Initiator) or start participating in the thread closer to the beginning (FirstMsg). They talk more within a thread (TokenRatio) and send significantly more (MsgCount) and longer (TokensCount, TokensPerMsg) messages. They also have significantly more instances of overt displays of power (ODPCount) than others.

7.2 Multiple Test Correction

The statistical measures presented in the previous section are exploratory in nature, presenting tests on all features, to gain a better understanding of their interaction with the situational power. We do not draw theoretical conclusions from the specific combination of interactions that are found statistically significant. Hence, we did not apply any corrections for multiple tests in statistical significance for individual features in the previous section. However, on applying the Bonferroni correction to adjust the p-value for the number of tests performed (threshold = 0.05/21 = 0.0024), three features (one each from PST (FirstMsg), VRB (TokenRatio) and ODP (ODPCount)) still remain statistically significant. Hence the global null hypothesis that the dialog structure and language use do not interact with situational power would still be rejected.

8 Automatic Situational Power Tagger

In this section, we present machine learning experiments to predict persons with situational power using features described in Section 6. We train a binary classifier to predict whether a participant has situational power or not.

8.1 Machinery

We used the ClearTK (Ogren et al., 2008) framework for extracting features and developing the classifier under the Apache UIMA framework. We used ClearTK's built-in tokenizer, POS tagger, lemmatizer and SVMLight (Joachims, 1999) wrapper. We balanced our dataset by up-sampling minority class instances in the training step. This has proven useful previously in cases of unbalanced datasets (Japkowicz, 2000). All results presented below have been obtained after balancing the training folds in cross validation; the test folds remain unchanged.

8.2 Experiments

Since we defined sets of dialog features (DAP, DLC, PST and VRB) in Section 6 based on separate aspects of communication they capture, we assume that these feature sets are reasonably coherent and individual features within a set interact with each other more strongly than with features from

Feature set	Features	P	R	F
Random		36.7	49.4	42.1
AlwaysTrue		36.7	100.0	53.6
VRB	TokenRatio, MsgRatio	43.9	70.4	54.0
PST	FirstMsg	45.1	67.9	54.2
DAP	ReqInform, Inform-Offline, Conventional, Commit	40.9	75.3	53.0
DLC	Blink, Flink, Clink, SFlink	49.6	**75.3**	59.8
LEX	Unigrams	54.9	55.6	55.2
ODP	ODPCount	**71.2**	51.9	**60.0**
BEST	DLC, ODP	59.4	70.4	64.4

Table 4: Cross validation results (P: Precision, R: Recall, F: F-measure) VRB: Verbosity, PST: Positional, DAP: Dialog acts, DLC: Dialog links, LEX: Lexical, ODP: Overt display of power

other feature sets. So, first we find the best performing subset of features for each feature set by exhaustive search within the set. The small cardinality of these feature sets (max of 6 for DAP and DLC) makes exhaustive search computationally feasible. For LEX, we found the best performing feature set to be Unigrams (Section 6.5), and ODP contains only one feature - ODPCount. Once we have the best subset of each feature set, we do another round of exhaustive search combining best performers of each set to find the overall best performing feature subset. We tried various feature selection methods such as information gain, which did not improve results. We believe this is because these methods rank each feature individually, and thus important interactions between features are not captured.

We used 5-fold cross validation on the data to evaluate the prediction performance for different feature subsets. The corpus was divided into 5 folds at the thread level. Active participants from 4 folds were used to train a model which was then tested on active participants in the 5th fold. We did this with all five configurations and all the reported results in this paper are micro-averaged results across 5 folds. We report (R)ecall, (P)recision and (F)-measure ($\beta = 1$). We experimented with a linear kernel and a quadratic kernel; the latter performed better. All results presented in this paper are obtained using a quadratic kernel.

8.3 Results

Table 4 shows cross validation results for each set of features. We present two baseline measures - **Random** and **AlwaysTrue**. In the Random baseline, we predict an active participant to have situational power at random. In AlwaysTrue baseline, we always predict an active participant to have situational power. The table 4 lists the best performing feature subset and corresponding precision/recall/f-measure for each set of features. As described in Section 8.2, the BEST feature set is found by doing exhaustive search on all combinations of best performers of VRB, PST, DAP, DLC, LEX and ODP.

The best performing individual feature sets are ODP and DLC, both at or near 60.0 F-measure. The random and AlwaysTrue baselines yield F-measures of 42.1 and 53.6, respectively. The best performers of all feature sets except DAP outperformed these baselines. The simple ngram based model obtained an F-measure of 55.2. While ODP results in a high precision (71.2) model, DLC yields a high recall (75.3) model; combination of both gave the best performing system with an

F measure of 64.4. The best performing single feature was ODPCount, which by itself gave an F measure of 60.0. The results we obtained are in line with the findings from the statistical significance study presented in Section 7. For example, DAP contained the least significant features while ODP contained the most significant feature. The tagger also performed worst when only DAP features were used and best when ODP was used. We assessed the statistical significance of F-measure improvements over baseline, using the Approximate Randomness Test (Yeh, 2000; Noreen, 1989).[3]. We found the improvements to be statistically significant (p = 0.001).

For the best performing feature set — DLC+ODP — we obtained a mean F-measure (macro-average) of 64.92 with a standard deviation of 8.82 (please note that Table 4 reports micro-averaged F-measure: 64.4). The low standard deviation suggests that the model built in this setting will obtain comparable performances for new unseen data. This also means that the data from which it was trained on the different folds of the cross-validation is sufficiently consistent to learn a model with predictive power. Put differently, our annotator's perception of situational power is coherent.

Conclusion and Future Work

In this paper, we studied the perception of situational power within written interactions. We have shown that situational power is not aligned with the organizational hierarchy. We have also shown that the perception of situational power correlates with various dialog structure and linguistic features. We presented an automatic situational power tagger to detect persons with power in written interactions. The methodology presented in this study can be applied to other forms of written interactions like discussions in online forums and blogs. We expect to find similar patterns of correlation in other genres.

As future work, we intend to study power relations between pairs of participants. It would be interesting to see how dialog features correlate with the other direction of power; that is from a submitter to an exerciser of power. Also, our current approach of aggregating features at the participant level is prone to noise. For example, let $\mathscr{X}$,$\mathscr{Y}$,$\mathscr{Z}$ be active participants such that $\mathscr{X}$ has power over $\mathscr{Y}$, who has power over $\mathscr{Z}$. When we aggregate features with respect to $\mathscr{Y}$, we are introducing noise from the part of communication between $\mathscr{X}$ and $\mathscr{Y}$. Extending our work to the person pair level would prevent this noise.

Our predictions are done using some gold features. Some features we used (verbosity, position) are readily derivable from the text; but others require processing. We will investigate using automatic taggers (such as a dialog act tagger and link predictor (Hu et al., 2009), an ODP tagger (Prabhakaran et al., 2012b)) to extract these features to predict power. However, one main contribution of this paper is to show the interaction between these dimensions of the dialog (like dialog structure and ODP) and situational power, which is an important first step towards solving the problem. Finally, in future work we will further study the manner in which different annotators interpret ambiguous threads in their power narratives, and identify different levels of certainty of situational power.

Acknowledgments

This work is supported, in part, by the Johns Hopkins Human Language Technology Center of Excellence. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the sponsor. We thank Weiwei Guo for his valuable suggestions on this paper. We also thanks several anonymous reviewers for their constructive feedback.

[3]http://www.clips.ua.ac.be/ vincent/scripts/art.py

References

Agarwal, A., Omuya, A., Harnly, A., and Rambow, O. (2012). A comprehensive gold standard for the enron organizational hierarchy. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 161–165, Jeju Island, Korea. Association for Computational Linguistics.

Bales, R. F. (1970). Personality and interpersonal behaviour. New York: Holt, Reinhart, and Winston.

Bales, R. F., Strodtbeck, F. L., M., M., T., and Roseborough, M. (1951). Channels of communication in small groups. *American Sociological Review*, pages 16(4), 461–468.

Bierstedt, R. (1950). An Analysis of Social Power. *American Sociological Review*.

Bramsen, P., Escobar-Molano, M., Patel, A., and Alonso, R. (2011). Extracting social power relationships from natural language. In *ACL*, pages 773–782. The Association for Computer Linguistics.

Brook, M. and Ng, S. H. (1986). Language and social influence in small conversational groups. *Journal of Language and Social Psychology*, pages 5(3), 201–210.

Creamer, G., Rowe, R., Hershkop, S., and Stolfo, S. J. (2009). Segmentation and automated social hierarchy detection through email network analysis. In Zhang, H., Spiliopoulou, M., Mobasher, B., Giles, C. L., Mccallum, A., Nasraoui, O., Srivastava, J., and Yen, J., editors, *Advances in Web Mining and Web Usage Analysis*, pages 40–58. Springer-Verlag, Berlin, Heidelberg.

Dahl, R. A. (1957). The concept of power. *Syst. Res.*, 2(3):201–215.

Danescu-Niculescu-Mizil, C., Lee, L., Pang, B., and Kleinberg, J. M. (2012). Echoes of power: language effects and power differences in social interaction. In Mille, A., Gandon, F. L., Misselis, J., Rabinovich, M., and Staab, S., editors, *WWW*, pages 699–708. ACM.

Diesner, J. and Carley, K. M. (2005). Exploration of communication networks from the enron email corpus. In *In Proc. of Workshop on Link Analysis, Counterterrorism and Security, SIAM International Conference on Data Mining 2005*, pages 21–23.

Emerson, R. M. (1962). Power-Dependence Relations. *American Sociological Review*, 27(1):31–41.

French, J. R. and Raven, B. (1959). The Bases of Social Power. In Cartwright, D., editor, *Studies in Social Power*, pages 150–167+. University of Michigan Press.

Gilbert, E. (2012). Phrases that signal workplace hierarchy. In *Proceedings of the ACM 2012 conference on Computer Supported Cooperative Work*, CSCW '12, pages 1037–1046, New York, NY, USA. ACM.

Hu, J., Passonneau, R., and Rambow, O. (2009). Contrasting the interaction structure of an email and a telephone corpus: A machine learning approach to annotation of dialogue function units. In *Proceedings of the SIGDIAL 2009 Conference*, London, UK. Association for Computational Linguistics.

Japkowicz, N. (2000). Learning from imbalanced data sets: Comparison of various strategies. In *AAAI Workshop on Learning from Imbalanced Data Sets*.

Joachims, T. (1999). Making Large-Scale SVM Learning Practical. In Schölkopf, B., Burges, C. J., and Smola, A., editors, *Advances in Kernel Methods - Support Vector Learning*, Cambridge, MA, USA. MIT Press.

Namata, Jr., G. M. S., Getoor, L., and Diehl, C. P. (2007). Inferring organizational titles in online communication. In *Proceedings of the 2006 conference on Statistical network analysis*, ICML'06, pages 179–181, Berlin, Heidelberg. Springer-Verlag.

Ng, S. H., Bell, D., and Brooke, M. (1993). Gaining turns and achieving high in influence ranking in small conversational groups. *British Journal of Social Psychology*, pages 32, 265–275.

Ng, S. H., Brooke, M., , and Dunne, M. (1995). Interruption and in influence in discussion groups. *Journal of Language and Social Psychology*, pages 14(4),369–381.

Noreen, E. W. (1989). *Computer-Intensive Methods for Testing Hypotheses : An Introduction*. Wiley-Interscience.

Ogren, P. V., Wetzler, P. G., and Bethard, S. (2008). ClearTK: A UIMA toolkit for statistical natural language processing. In *Towards Enhanced Interoperability for Large HLT Systems: UIMA for NLP workshop at Language Resources and Evaluation Conference (LREC)*.

Peterson, K., Hohensee, M., and Xia, F. (2011). Email formality in the workplace: A case study on the enron corpus. In *Proceedings of the Workshop on Language in Social Media (LSM 2011)*, pages 86–95, Portland, Oregon. Association for Computational Linguistics.

Pfeffer, J. (1981). *Power in organizations*. Pitman, Marshfield, MA.

Prabhakaran, V., Rambow, O., and Diab, M. (2012a). Annotations for power relations on email threads. In *Proceedings of the Eighth conference on International Language Resources and Evaluation (LREC'12)*, Istanbul, Turkey. European Language Resources Association (ELRA).

Prabhakaran, V., Rambow, O., and Diab, M. (2012b). Predicting overt display of power in written dialogs. In *Human Language Technologies: The 2012 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, Montreal, Canada. Association for Computational Linguistics.

Reid, S. A. and Ng, S. H. (2000). Conversation as a resource for in influence: evidence for prototypical arguments and social identification processes. *European Journal of Social Psychology*, pages 30, 83–100.

Scherer, K. R. (1979). Voice and speech correlates of perceived social influence in simulated juries. In *H. Giles and R. St Clair (Eds), Language and social psychology*, pages 88–120. Oxford: Blackwell.

Shetty, J. and Adibi, J. (2005). Discovering important nodes through graph entropy the case of enron email database. In *Proceedings of the 3rd international workshop on Link discovery*, LinkKDD '05, pages 74–81, New York, NY, USA. ACM.

Strzalkowski, T., Broadwell, G. A., Stromer-Galley, J., Shaikh, S., Taylor, S., and Webb, N. (2010). Modeling socio-cultural phenomena in discourse. In *Proceedings of the 23rd International Conference on COLING 2010*, Beijing, China. Coling 2010 Organizing Committee.

Wartenberg, T. E. (1990). *The forms of power: from domination to transformation*. Temple University Press.

Yeh, A. (2000). More accurate tests for the statistical significance of result differences. In *Proceedings of the 18th conference on Computational linguistics - Volume 2*, COLING '00, pages 947–953, Stroudsburg, PA, USA. Association for Computational Linguistics.

Yeh, J.-Y. and Harnly, A. (2006). Email thread reassembly using similarity matching. In *CEAS 2006 - The Third Conference on Email and Anti-Spam, July 27-28, 2006, Mountain View, California, USA*, Mountain View, California, USA.

Bilingual Lexicon Construction from Comparable Corpora via Dependency Mapping

QIAN Long Hua WANG Hong Ling ZHOU Guo Dong ZHU Qiao Ming[*]
NLP Lab, School of Computer Science and Technology
Soochow University, Suzhou China 215006
{qianlonghua,hlwang,gdzhou,qmzhu}@suda.edu.cn

ABSTRACT

Bilingual lexicon construction (BLC) from comparable corpora is based on the idea that bilingual similar words tend to occur in similar contexts, usually of words. This, however, introduces noise and leads to low performance. This paper proposes a bilingual dependency mapping model for BLC which encodes a word's context as a combination of its dependent words and their relationships. This combination can provide more reliable clues than mere context words for bilingual translation words. We further demonstrate that this kind of bilingual dependency mappings can be successfully generated and maximally exploited without human intervention. The experiments on BLC from English to Chinese show that, by mapping context words and their dependency relationships simultaneously when calculating the similarity between bilingual words, our approach significantly outperforms a state-of-the-art one by ~14 units in accuracy for frequently occurring noun pairs and similarly, though in a less degree, for nouns and verbs in a wide frequency range. This justifies the effectiveness of our dependency mapping model for BLC.

TITLE AND ABSTRACT IN ANOTHER LANGUAGE, CHINESE

应用依存映射从可比较语料库中抽取双语词表

从可比较语料库中抽取双语词表的基本思想是，双语相似的词语出现在相同的语词上下文中。不过，这种方法引入了噪声，从而导致了低的抽取性能。本文提出了一种用于双语词表抽取的双语依存映射模型，在该模型中一个词语的上下文结合了依存词语及其依存关系。这种结合方法为双语词表构建提供了比单一的词语上下文更为可靠的信息。我们还进一步展示了在没有人工干预的情况下可以产生和利用这种双语依存关系。从英文到中文的双语词表构建实验表明，通过在计算双语词语相似度时同时映射词语及其依存关系，同目前性能最好的系统相比，我们的方法显著提高了精度。对于经常出现的名词，精度提高了14个百分点；对于较大频率范围内的名词和动词，性能也提高了，尽管程度较小。这说明了依存映射模型对双语词表构建的有效性。

KEYWORDS: Bilingual Lexicon Construction, Comparable Corpora, Dependency Mapping
KEYWORDS IN CHINESE: 双语词表构建, 可比较语料库, 依存映射

[*] Corresponding author

Proceedings of COLING 2012: Technical Papers, pages 2275–2290,
COLING 2012, Mumbai, December 2012.

1 Introduction

Bilingual lexicons play an important role in many natural language processing tasks, such as machine translation (MT) (Och and Ney, 2003; Gong et al., 2011) and cross-language information retrieval (CLIR) (Grefenstette, 1998). Traditionally, bilingual lexicons are built manually with tremendous efforts. With the availability of large-scale parallel corpora, researchers turn to automatic construction from parallel corpora and achieve certain success (Wu and Xia, 1994). However, large-scale parallel corpora do not always exist for most language pairs. Therefore, researchers turn their attention to either pivot languages or non-parallel but comparable corpora.

Using pivot languages in BLC was pioneered by Tanaka and Umemura (1994). Thereafter, various studies have been done to take advantage of multiple paths (Mann and Yarowsky, 2001) and even multiple pivot languages (Mausam et al., 2009) between the source and target languages. Since such automatically constructed lexicons usually contain noisy and polysemous entries, corpus-based occurrence information has been widely used to help rank the candidate target words (Schafer and Yarowsky, 2002; Kaji et al., 2008; Shezaf and Rappoport, 2010).

Alternatively, extracting bilingual lexicons from comparable corpora assumes that words with similar meanings in different languages tend to occur in similar contexts, even in non-parallel corpora. Rapp (1999) and Fung (2001) proposed a bilingual context vector mapping strategy to explore word co-occurrence information. Both studies rely on a large, one-to-one mapping seed lexicon between the source and target languages. Koehn and Knight (2002) investigated various clues such as cognates, similar context, preservation of word similarity and word frequency. Garera et al. (2009) proposed a dependency-based context model and achieved better performance than previous word-based context models. Recent studies concentrate on automatic augmentation of the seed lexicon either by extracting identical words between two closely related languages (Ficšer and Ljubešić, 2011) or by aligning translation pairs from parallel sentences, which is mined in advance from a comparable corpus (Morin and Prochasson, 2011). The problem with above method is that they only consider the words involved in the contexts and ignore other rich information therein, such as syntactic relationships, thus usually suffering from low performance especially when they are applied to two distinct languages such as English and Chinese. For example, our preliminary experiment with the dependency-based model (Garera et al., 2009) shows that English source word "profit" matches wrongly with Chinese target word "企业" (enterprise), instead of the correct one "利润", due to the higher similarity score with the former than that with the latter. Further exploration shows that the word "企业" has a much higher frequency than the word "利润" in the adopted corpus, thus tends to have a higher similarity score due to richer (nevertheless noisy) contexts. We also find that some relevant contextual words with both target words, such as "实现" (realize) and "成本" (cost) etc., share the same or corresponding dependency relationships with "利润" (profit), i.e. *dobj* and *conj*, but not with "企业" (enterprise).

In order to take advantage of this observation, this paper proposes a bilingual dependency mapping model for BLC from a comparable corpus by extending the scope of a word's context from mere neighbouring words to both dependent words and their dependency relationships. The basic idea underlying our model is that bilingual similar words tend to occur within similar

bilingual contexts involving not only dependent words but also their relationships, and the similarity between bilingual words can be better calculated by considering the mappings of both context words and their relationships. Furthermore, while the mappings of bilingual words may suffer from the data sparseness problem due to the availability of only a small scale of given seed lexicon, the mappings of dependency relationships can be reliably generated from the seed lexicon without human intervention, making our method easily adapted to other language pairs and domains. Finally, the weights of different dependency mappings can be automatically learned using a simple yet effective perceptron algorithm.

The remainder of this paper is organized as follows. Section 2 reviews the related work. Section 3 introduces our comparable corpus and a strong baseline. Section 4 details our dependency mapping approach while the experimentation is described in Section 5. Finally, Section 6 draws the conclusion with future directions.

2 Related work

In this section, we limit the related work to BLC from comparable corpora between English and Chinese. For others, please refer to the general introduction in Section 1.

Due to distinct discrepancies between English and Chinese, BLC from comparable corpora between these two languages is challenging. Fung (2000) extracted word contexts from a comparable corpus, and calculated the similarity between word contexts via an online dictionary. Particularly she analyzed the impact of polysemous words, Chinese tokenization and English morphological information. Zhang et al. (2006) built a Chinese-English financial lexicon from a comparable corpus with focus on the impact of seed lexicon selection. Haghighi et al. (2008) proposed a generative model to construct lexicons for multiple language pairs, including English-Chinese, via canonical correlation analysis, which effectively explores monolingual lexicons in terms of latent matching.

In particular for BLC without any external lexicon, Fung (1995) focused on context heterogeneity in Chinese and English languages, which measures how productive the context of a word is, instead of its absolute occurrence frequency. She suggested that bilingual translation words tend to share similar context heterogeneity in non-parallel corpora. Specifically, she calculated the similarity between two bilingual words using the ratios of unique words in the right and left contexts. Yu and Tsujii (2009) proposed the notion of dependency heterogeneity, which assumes that a word and its translation should share similar modifiers and heads in comparable corpora, no matter whether they occur in similar contexts or not. In this sense, our approach is similar to theirs. However, while their distance measure of dependency heterogeneity is limited to three easily-mapping common relationships between two languages, namely SUB, OBJ and NMOD, we further generalize to automatic mappings of any bilingual dependency relationships. Another difference is that our method considers dependent words and their relationships simultaneously.

3 Corpus and baseline

This section introduces the comparable corpus and the bilingual seed/test/development lexicons used in this paper for evaluation as well as a state-of-the-art baseline for BLC.

3.1 Comparable corpus

In this paper, we generate a comparable corpus from the parallel Chinese-English Foreign Broadcast Information Service (FBIS) corpus, gathered from the news domain. This bilingual corpus contains about 240k sentences, 6.9 million words in Chinese and 8.9 million words in English. Similar to the way adopted in (Garera et al., 2009; Haghighi et al., 2008), we couple the first half of Chinese corpus and the second half on the English side as our comparable corpus.

For corpus pre-processing, we use the Stanford POS tagger (Toutanova and Manning, 2000) and syntactic parser (Marneffe et al., 2006) to generate the POS and dependency information for each sentence in both Chinese and English corpora. Particularly, English words are transformed to their respective lemmas using the TreeTagger package (Helmut, 1994).

3.2 Bilingual lexicons for evaluation: seed, test and development

In the literature, different scales of bilingual seed lexicons have been used. For example, Rapp (1999) and Fung (2000) used large-scale dictionaries of 10-20k word pairs while other studies (Koehn and Knight, 2002; Haghighi et al., 2008; Garera et al., 2009) used only small dictionaries of about 100-1000 word pairs.

In this paper, we adopt a small scale one. In particular, we use the GIZA++ package (Och and Ney, 2000) to extract the most frequently occurring 1000 word pairs as the bilingual seed lexicon (denoted as L_s) and the subsequent 500 noun pairs (denoted as LN_t) as the primary bilingual test lexicon. This way of generating the test lexicon for nouns has been commonly used in previous studies (Koehn and Knight, 2002; Haghighi et al., 2008; Garera et al., 2009). Besides, we frame a secondary test lexicon (denoted as LA_t) including 200 nouns, verbs and adjectives respectively, which spread evenly in the four ranges of 1001-2000, 2001-3000, 3001-4000 and 4001-5000. The goal of LA_t is to evaluate the adaptability of our method to words with different categories in a wide frequency range.

Different from other studies on context-based BLC which use the seed lexicon only for bilingual word projection, we set aside a bilingual development lexicon L_d of nouns, verbs and adjectives (denoted as LN_d, LV_d and LJ_d) respectively for fine-tuning our BLC system. This bilingual development lexicon is constructed by randomly selecting 200 nouns, 200 verbs and 100 adjectives[1] in the bilingual seed lexicon. Obviously we have $L_d = \{LN_d \bigcup LV_d \bigcup LJ_d\} \subset L_s$.

3.3 Baseline

As a state-of-the-art baseline, Garera et al. (2009) extracted the words from the dependency tree with a fixed window size of ±2 as the context. That is, given a word w in a sentence, all the words corresponding to its immediate parent (-1), immediate children (+1), grandparent (-2) and grandchildren (+2) are extracted as features to constitute a context vector. Specifically, each feature in the vector is weighted by its point-wise mutual information (PMI) with the word w, defined as:

$$PMI(w,c) = \log_2 \frac{N(w,c) * N}{N(w) * N(c)} \quad (1)$$

[1] In the bilingual seed lexicon there are about 500, 300, 200 English and Chinese noun, verb and adjective pairs respectively.

where *N(w, c)* is the co-occurrence frequency of word *w* and its context word *c*, *N(w)* and *N(c)* are the occurrence frequencies of the words *w* and *c* respectively, *N* is the number of all word occurrences in the corpus. Since PMI is usually biased towards infrequent words, we multiplied it with a discounting factor as described in (Lin and Pantel, 2002):

$$\frac{N(w,c)}{N(w,c)+1} \times \frac{\min(N(w),N(c))}{\min(N(w),N(c))+1} \tag{2}$$

Then, the pair-wise similarity scores between source word w_s and candidate target words w_t are computed using the cosine similarity measure as follows:

$$Sim_{DW}(w_s,w_t) = COS(W_s,W_t) = \frac{\sum_i W_s^i \times W_t^i}{\sqrt{\sum_i W_s^i} \times \sqrt{\sum_i W_t^i}} \tag{3}$$

where W_s and W_t are the dependent word vectors of source word w_s and candidate target words w_t respectively, W_s^i and W_t^i are the discounted PMI values of the *i*th features f_s^i and f_t^i, s.t. $(f_s^i, f_t^i) \in L_s$. We call Formula (3) the dependent word similarity as it is calculated solely on dependent words in the contexts.

Finally, all the candidate target words are ranked in terms of their dependency word similarity scores with source word w_s, and the top one $\hat{w}_t$ is selected as the translation word:

$$\hat{w}_t = \arg\max_{w_t \in GEN(w_s)} Sim_{DW}(w_s,w_t) \tag{4}$$

Where $GEN(w_s)$ is a function that enumerates a set of candidates for source word w_s. Here, as our goal is to build a lexicon of nouns occurring frequently in the source text, one reasonable assumption is that their translation counterparts also occur frequently in the target text. Therefore, in order to reduce the computation cost we limit the candidate words for source word w_s, $GEN(w_s)$ to the most frequently occurring nouns with the number set to 10 times of the size of the lexicon.

4 BLC via dependency mappings

In this section, we first present the dependency mapping model for BLC and then detail on how to manually and automatically generate dependency mappings via a given development lexicon.

4.1 Dependency mapping model

Following Garera et al. (2009) and Yu and Tsujii (2009), we further postulate that the mapping of dependency relationships can hold between two translation words in comparable corpora. It is worth noting that one dependency relationship in one language may not always be directly mapped to the same relationship in another language and there are even cases where one relationship may map to multiple relationships in another language (cf. Fig. 1).Generally, we can enumerate the mappings of dependency relationships between English and Chinese, either by crafting manually or generating automatically. Suppose that we already have such dependency mappings between English and Chinese at hand, denoted as Ψ. Compared with the baseline procedure in Subsection 3.3, bilingual lexicon construction via dependency mappings can be revised by generating dependency mapping context vectors, in which each feature combines a dependent word and its relationship. Here, the window size is fine-tuned to ±1 (using the

development lexicon LN_d) with the relationship direction not considered. Formally, the weight of each feature can be recast via PMI as:

$$PMI(w,ct) = \log_2 \frac{N(w,ct) * N}{N(w) * N(ct)} \quad (1')$$

where *ct* denotes the combination of the dependent word and its relationship. Similarly, this PMI value is also discounted according to Formula (2).

Likewise, the dependency mapping similarity between source word w_s and target candidate word w_t is calculated using the cosine similarity as follows:

$$Sim_{DM}(w_s, w_t) = COS(D_s, D_t) = \frac{\sum_i D_s^i \times D_t^i}{\sqrt{\sum_i D_s^i} \times \sqrt{\sum_i D_t^i}} \quad (3')$$

where D_s and D_t are the dependency mapping vectors of source word w_s and candidate target words w_t respectively, D_s^i and D_t^i are the discounted PMI values of the ith features f_s^i and f_t^i whose involved words are translation pairs in the seed lexicon and whose involved dependency types are bilingually mapped in Ψ, i.e. $(f_s^i.word, f_t^i.word) \in L_s$ and $(f_s^i.type, f_t^i.type) \in \Psi$.

Obviously, we can rank the candidate target words in terms of their dependency mapping similarity scores with the source word and select the top one as the translation word. However, this often leads to the data sparseness problem since the combination of a dependent word and its relationship occurs much less frequently than a dependent word alone. Therefore, the dependent word similarity and dependency mapping similarity are interpolated linearly for candidate ranking as follows:

$$Sim_T(w_s, w_t) = \alpha \times Sim_{DW}(w_s, w_t) + (1-\alpha) \times Sim_{DM}(w_s, w_t) \quad (5)$$

Where $Sim_T(w_s, w_t)$ denotes the overall similarity score between the words w_s and w_t, and α is a coefficient to balance these two similarity measures and can be fine-tuned using the development lexicon L_d.

4.2 Manually crafting dependency mappings

Considering the number of dependency relationships in both English and Chinese, e.g. 53 dependency relationships in Stanford encoding scheme, there are potentially thousands of possible mappings between these two languages. Fortunately, the distribution of various dependency relationships is severely skewed. Table 1 lists the statistics for the dependency relationships whose percentages are greater than 2% in the descending order, where the left three columns denote the English relationships, their short descriptions and percentages as well as the Chinese statistics on the right three columns. These statistics are obtained from 5000 most frequently occurring nouns in our English and Chinese corpora.

It shows that the top 8 types (*prep*, *conj*, *nsubj*, *nn*, *amod*, *dobj*, *dep* and *poss* for English and *nn*, *conj*, *dobj*, *assmod*, *nsubj*, *rcmod*, *amod* and *dep* for Chinese) account for 87.2% and 94.2% of total dependency relationships for English and Chinese respectively. This means when we consider dependency mappings between English and Chinese, we can safely ignore other relationships whose percentages are less than 2%. Furthermore, since the *dep* one in both English and Chinese denotes general dependency relationship that can not be nicely fitted into other more

specific ones, the mapping between *dep* and any other ones are not considered subsequently in this paper.

EN Rel.	Short Description	%	CN Rel.	Short Description	%
prep	prepositional modifier	35.7	nn	noun modifier	32.0
conj	conjunction	11.8	conj	conjunction	13.1
nsubj	nominal subject	9.3	dobj	direct object	11.8
nn	noun modifier	8.6	assmod	associative modifier	11.4
amod	adjectival modifier	8.1	nsubj	nominal subject	9.6
dobj	direct object	7.6	rcmod	relative clausal modifier	8.0
dep	general dependency	3.6	amod	adjectival modifier	4.5
poss	possessive modifier	2.5	dep	general dependency	3.8
Total		87.2	Total		94.2

TABLE 1 –Statistics on dependency relationships for English and Chinese nouns

Using linguistic knowledge from both English and Chinese languages, we manually craft 10 dependency mappings Ψ_M between these two languages, as shown in Fig. 1.

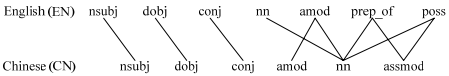


FIGURE 1 – Bilingual dependency mappings between English and Chinese

From the figure we can see that while some mappings, such as *nsubj* (EN) to *nsubj* (CN) and *dobj* (EN) to *dobj* (CN), capture common grammatical relationships in both languages, others, such as *poss* (EN) to *assmod* (CN), indicate the differences across the two languages. Particularly interesting is that *nn* (CN) can map to four relationships, namely *nn*, *amod*, *prep_of* and *poss* (EN) while *nn* (EN) has only one correspondence *nn* (CN). This indicates that *nn* (CN) is much more productive and ambiguous than *nn* (EN). This scenario can be illustrated by possible mapping of example Chinese phrase "中国银行" (*nn*) to English phrases "China Bank" (*nn*), "Chinese Bank" (*amod*), "Bank of China" (*prep_of*) , and "China's Bank" (*poss*), though only the third is correct while all the others are merely grammatically reasonable.

4.3 Automatically generating dependency mappings

While manually crafting dependency mappings between two languages do serve our purpose, its limitation exists in the need for bilingual knowledge and the lack of flexibility. One alternative is to automatically generate bilingual dependency mappings via a development lexicon. The idea behind is that not only the bilingual words with similar dependent words and dependency relationships tend to pair each other (cf. Section 4.1), but also the bilingual dependency relationships between similar dependent words and their context words tend to map to each other. Fig. 2 illustrates an algorithm to derive bilingual dependency mappings via a development lexicon.

In this figure, Ψ_A denotes the set of bilingual dependency mappings to be automatically generated while D_s and D_t are the respective dependency mapping vectors for source word s_i and target word t_i respectively.

Input: $L_d = \{(s_i, t_i)\}_{i=1}^{N}$

Output: Ψ_A

Initialize: $\Psi_A = NULL$

1. for i =1…N
2. extract D_s and D_t for s_i and t_i
3. for each feature f_s^j and f_t^j in D_s and D_t
4. if $(f_s^j.word, f_t^j.word) \in L_s$ then
5. add this mapping and its count to Ψ_A
6. end if
7. end for
8. end for
9. calculate the percentage for each mapping in Ψ_A
10. keep the top 30 most frequent mappings in Ψ_A

FIGURE 2 –Algorithm for automatically generating bilingual dependency mappings

Table 2 shows the derived bilingual dependency mappings from English to Chinese along with their percentages. Compared with Fig. 1, we can see that all the 10 manually crafted mappings (marked in italics fonts) in Ψ_M can be found in the top 30 automatically generated mappings Ψ_A, with 8 in top 10. This implies high consistency between Ψ_A and Ψ_M. A natural question one may ask is: are those extra mappings in Ψ_A but not in Ψ_M noisy for BLC?

No	EN-CN map.	%	No	EN-CN map.	%	No	EN-CN map.	%
1	*prep_of-nn*	7.3	11	nsubj-nn	1.7	21	prep_of-dobj	0.9
2	*nn-nn*	7.0	12	dobj-nn	1.6	22	nn-conj	0.9
3	*amod-nn*	6.2	13	conj-assmod	1.5	23	dobj-rcmod	0.8
4	*conj-conj*	5.5	14	nsubj-dobj	1.4	24	nsubj-assmod	0.8
5	*dobj-dobj*	5.4	15	*poss-nn*	1.4	25	amod-conj	0.8
6	conj-nn	4.9	16	prep_of-conj	1.2	26	amod-rcmod	0.8
7	*amod-amod*	3.2	17	amod-assmod	1.1	27	nn-assmod	0.8
8	*prep_of-assmod*	3.1	18	prep_for-nn	1.0	28	conj-dobj	0.8
9	*nsubj-nsubj*	2.5	19	dobj-nsubj	1.0	29	*poss-assmod*	0.7
10	prep_in-nn	2.4	20	prep_in-assmod	1.0	30	dobj-conj	0.7

TABLE 2 – Top 30 dependency mappings mined via the development lexicon

To answer this question is by no means a trivial task. Although we are quite sure that some of the mappings in Ψ_A are irrelevant such as *nsubj-assmod*, for others it's difficult to determine their relevancy with BLC from the linguistic perspective, just as we are not sure whether there are other useful mappings missing in Ψ_M. Therefore, we adopt an ablation testing strategy to progressively remove those mappings whose removal lead to better performance using the development lexicon L_d. Fig. 3 illustrates the algorithm, where P^i in Line 2 denotes the performance achieved on the development lexicon L_d using previous mappings Ψ_A^{i-1} and P_j in Line 4 denotes the performance on L_d using mappings Ψ_A^{i-1} minus the jth mapping. The output of this algorithm $\hat{\Psi}_A$ maximizes the performance on the development lexicon L_d.

Input: Ψ_A and L_d

Output: $\hat{\Psi}_A$

Initialize: $\Psi_A^0 = \Psi_A$

Steps:

1. for i =1… $|\Psi_A|$
2. calculate P^i using Ψ_A^{i-1} on L_d
3. for each mapping m_j in Ψ_A^{i-1}
4. calculate P_j using $\Psi_A^{i-1} - \{m_j\}$ on L_d
5. end for
6. $\hat{m} = \arg\max_{m_j} P_j$
7. $\Psi_A^i = \Psi_A^{i-1} - \{\hat{m}\}$
8. end for
9. $\hat{\Psi}_A = \arg\max_{\Psi_A^{i-1}} P^i$

FIGURE 3 –Algorithm for filtering the noisy mappings in Ψ_A

4.4 Weight learning via perceptron

While coefficient α in Formula (5) can be determined empirically using a development lexicon, it equally weighs different mappings. We argue that this may not be optimal since different mappings may have different contributions to BLC. That is, different mappings should have different weights to exploit such difference. Thus, Formula (5) can be recast as follows:

$$Sim_T(w_s, w_t) = \alpha^0 \times Sim_{DW}(w_s, w_t) + \sum_{i=1}^{|\Psi|} \alpha^i Sim_{DM}^i(w_s, w_t) \quad (5')$$

In this paper, we propose a simple perceptron algorithm to optimize those weights for different mappings using the development lexicon. Generally, the perceptron algorithm is guaranteed to find a hyper-plane that classifies all training points, if the data is separable. Even the data is non-separable as in most practical cases, the variants of perceptron (Freund and Schapire, 1999; Collins and Duffy, 2002), such as averaged perceptron (AP) or voted perceptron (VP), can generalize well.

Input: training examples (s_i,t_i)

Output: w

Initialize: w= w^0

Steps:

1. for i = 1…T
2. for j =1…N
3. Calculate $\hat{t}_j = \arg\max_{t \in GEN(s_j)} \Phi(s_j, t) \times w$
4. If $\hat{t}_j != t_j$ then $w = w + \Phi(s_j, t_j) - \Phi(s_j, \hat{t}_j)$
5. $u_{i,j} = w$
6. end for
7. end for
8. output $w = \sum_{i,j} u_{i,j} / NT$

Figure 4 – Perceptron algorithm for weight learning

Fig. 4 shows our averaged perceptron algorithm, where

- d, the number of features in a vector, is set to the order of the mapping set plus 1 for the dependency word similarity.
- Φ is a function which maps each bilingual word pair (s, t) to a feature vector $\Phi(s,t) \in R^d$.
- $w \in R^d$ is a weight vector for different mappings, and its initial value w^0 is set to the occurrence ratio for each mapping as in Table 2.

Here, the function Φ calculates the dependent word similarity score (cf. Formula (3)) and the similarity scores for each dependency mapping in Ψ (cf. Formula (3')), $u_{i,j}$ stores the weight vector in the ith iteration given the jth training example, and T denotes the number of iterations over the development lexicon[2].

5 Experimentation

This section systematically evaluates our approach for English-Chinese BLC. In this paper, precision (P) and mean reciprocal rank (MRR) are used as our evaluation metrics, as done in the literature (Koehn and Knight, 2002; Garera et al., 2009; Yu and Tsujii, 2009), where precision is the average accuracy within the top n (n=1 here) most similar words while MRR is the average of the reciprocal ranks for all the test words:

$$precision = \frac{count_{top1}}{N_w} \tag{6}$$

$$MRR = \frac{1}{N_w}\sum_{i=1}^{N_w}\frac{1}{rank_i} \tag{7}$$

where $count_{top1}$ is the number of correct translation words on the top one ranking, and $rank_i$ denotes the rank of the correct translation word for the ith test source word, and N_w is the total number of the test source words (e.g., 500).

5.1 Performance on the development noun lexicon LN_d

In order to determine the optimal subsets in automatic mappings Ψ_A and manual mappings Ψ_M respectively, we conduct ablation tests on the development lexicon LN_d (cf. Fig. 3) using proportional weights (PR) or automatic weights learned by the AP algorithm (cf. Fig. 4). Fig 6 depicts the MRR performance scores for 4 combinations, i.e. Auto-PR, Auto-AP, Manual-PR and Manual-AP. For Ψ_A, x-axis denotes the top 30 mappings (cf. Table 2) while for Ψ_M we present the data from the 20th iteration as there are only 10 mappings (cf. Fig. 1). At each iteration, the mapping whose removal causes the biggest performance increase is removed and the MRR score is measured using the remaining mappings. The integer numbers in Table 2 indicate the mappings to be removed at the corresponding iteration. Please note that the first iteration corresponds to all dependency mappings in Ψ_A or Ψ_M and the last one corresponds to the baseline without any dependency mapping. The figure shows that:

[2] T is experimentally tuned to 200 since the average perceptron algorithm converges after 200 iterations on the development lexicon LN_d.

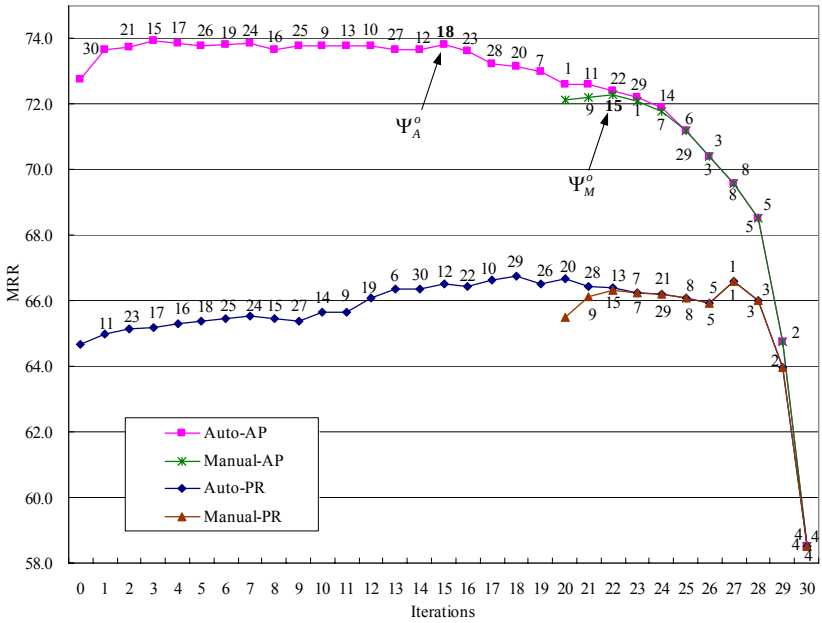


FIGURE 6 – MRR performance of ablation tests for Ψ_A and Ψ_M on the development noun lexicon

- The dependency mapping model with average perceptron significantly outperforms its counterpart with proportional weights. This suggests that weight optimization via perceptron algorithms substantially helps BLC. Therefore, the following experiments do not consider proportional weights.
- All the 4 combinations exhibit similar trends. That is, with the shrinkage of the mappings, the performance first slightly increases, then peaks at some iteration, afterwards decreases slowly, and finally drops significantly. This trend suggests that there do exist some noisy mappings, more or less in both Ψ_A and Ψ_M. However, removing them can only lead to limited performance improvement while too few mappings severely harm the performance.
- An interesting phenomenon is that the two combinations (Auto-AP and Manual-AP) converge at the last 7 iterations, so do the other two (Auto-PR and Manual-PR). This implies high consistency between manually crafted mappings and top automatically generated mappings no matter whether their weights are fine-tuned using a proper machine learning algorithm or fixed to their proportions in the corpus.

According to above observations, we select the most consistent mappings as the optimal ones in all the following experiments instead of choosing the best mappings for each combination. These sets, marked by the symbols Ψ_A^o (15 mappings for Auto_AP) and Ψ_M^o (8 mappings for Manual_AP) in the figure, include medium-sized mappings which perform the best or very

closely to the best. It is not surprising to find that 7/8 mappings in Ψ_M^o are included in Ψ_A^o because of their high consistency.

5.2 Comparison of mapping weights in Ψ_A^o and Ψ_M^o

Table 3 ranks the weights learned for each mapping in Ψ_A^o and Ψ_M^o in the descending order, where the mappings with negative weights are omitted. It shows that different mappings have very different weights, among which *conj-conj* and *poss-assmod* are ranked high while *nn-nn* and *prep_of-nn* are ranked low. This suggests that the former two are more discrimitive than the latter two in dependency mapping-based BLC, though the latter two occur more frequently as indicated in Table 2. In other words, the meaning of a noun depends more on its conjuncts and possessive/associative modifiers to a certain extent than on its prepositional phrases or noun premodifers.

Auto Ψ_A^o	Weights	Manual Ψ_M^o	Weights
conj-conj	0.2317	poss-assmod	0.3022
poss-assmod	0.2268	conj-conj	0.1866
prep_in-assmod	0.1738	amod-amod	0.1251
dobj-dobj	0.1597	prep_of-assmod	0.1141
prep_of-assmod	0.1264	*nn-nn*	0.1111
amod-nn	0.1201	dobj-dobj	0.1102
amod-amod	0.1161	amod-nn	0.1092
nn-nn	0.0982	*prep_of-nn*	0.0723
conj-dobj	0.0758	–	–
prep_of-nn	0.0501	–	–

TABLE 3 – Comparison of mapping weights in Ψ_A^o and Ψ_M^o

5.3 Performance on the test noun lexicon LN_t

Table 4 shows the performance of our dependency mapping model (DM) on the test noun lexicon LN_t. Three weighting strategies, i.e. DM (equal weights), DM_PR (proportional weights) and DM_AP (weight learning by the AP algorithm on LN_d) are used with four sets of dependency mappings (Ψ_M, Ψ_M^o, Ψ_A and Ψ_A^o). Please note that all the optimal mappings are tuned on the development lexicon as discussed in Fig. 6, though we don't present the results of ablation tests for DM there as they are similar to those of DM_PR. For comparison, the strong baseline as discussed in Section 3 is included at the top row. The table shows that:

- The AP[3] algorithm (DM_AP) achieves the best performance with the improvements of ~6 units in P and ~5 in MRR compared with DM. In most cases, DM_PR slightly underperforms DM, which means that simply using the occurrence ratio as each mapping's weight doesn't lead to performance improvement.
- In total, our method outperforms the strong baseline by ~14 units in both P and MRR across all the mapping sets. This suggests that via weight learning with a proper machine

[3] The voted perceptron algorithm achieves similar results, so here we omit them for brevity.

learning method (e.g. a simple perceptron algorithm), the dependency mapping model can dramatically improve the performance for BLC.

- A particularly important finding is that the optimal automatic mapping set Ψ_A^o performs comparably with the manual mapping set Ψ_M and slightly outperforms the complete automatic mapping set Ψ_A. This further justifies the appropriateness of automatically generating bilingual dependency mappings.

Systems	DM		DM_PR		DM_AP	
	P (%)	MRR(%)	P (%)	MRR(%)	P (%)	MRR(%)
Baseline	33.8	42.23	33.8	42.23	33.8	42.23
Manual Ψ_M	43.0	51.89	42.4	51.61	**49.2**	**57.18**
Manual Ψ_M^o	44.0	52.44	40.2	49.52	**50.0**	**57.62**
Auto Ψ_A	41.4	50.70	39.0	48.64	**46.2**	**55.50**
Auto Ψ_A^o	42.6	51.66	43.6	51.85	**48.6**	**56.43**

TABLE 4 – Performance of dependency mapping on the test noun lexicon LN_t

Above observations justifies that dependency mappings can significantly enhance the performance for BLC, and perform even better when the weight for each mapping is optimized. To explain how dependency mappings work, we take English word "profit" and its Chinese translation "利润" as an example, which is also mentioned in Section 1. Table 5 compares three kinds of similarity scores between "profit" and its two translation candidates, i.e., "利润" and "企业". It shows that due to Sim_{DM} ("profit", "利润") >Sim_{DM} ("profit","企业"), our method eventually acquires the correct translation pair of "profit" and "利润". The reason is that the dependency mapping context between "profit" and "利润", such as "实现_*dobj*" (realize_*dobj*) and "成本_*conj*" (cost_*conj*) etc., is more evidential than that between "profit" and "企业". In other words, the dependency mapping context contains more accurate bilingual corresponding words and ignores noisy ones than the dependent word context, thus leading to better performance for BLC.

Similarity	("profit", "利润")	Relationship	("profit", "企业")
Sim_{DW}	0.441	<	0.475
Sim_{DM}	0.460	>	0.393
Sim_T	0.456	>	0.409

TABLE 5 – Similarity comparison between "profit" and its two translation candidates

5.4 Performance on the general test lexicon LA_t

Table 6 compares the performance of different methods on the general test lexicon LA_t for words with different categories (i.e. nouns, verbs and adjectives) in a wide frequency range. Specifically, for the DM and DM_AP methods, the automatic mappings and their weights for nouns are the same as those in Table 2 while for verbs and adjectives, the mappings are first automatically generated and then filtered using the ablation tests with their weights learned on the development sets LV_d and LJ_d respectively. The table shows that;

- For nouns and verbs, both the DM and DM_AP methods with automatic dependency mappings outperform the baseline, though with the improvements in a less degree than those for nouns in LN_t, due to the much more data sparseness of both the dependent word context and dependency mapping context.
- For adjectives, however, the performances of three methods are quite similar. This suggests the non-effectiveness of dependency mappings on BLC for adjectives.

Parts of speech	Baseline		DM		DM_AP	
	P(%)	MRR(%)	P(%)	MRR(%)	P(%)	MRR(%)
Nouns	21.8	28.62	26.0	33.60	**28.3**	**35.02**
Verbs	19.5	27.07	23.5	31.69	**26.0**	**35.26**
Adjectives	35.5	46.22	**36.0**	**46.93**	34.5	45.63

TABLE 6 – Performance of different methods on the general test lexicon LA_t

Above observations demonstrate that our method can well adapt to nouns and verbs in a wide frequency range, but not to adjectives. Our exploration shows that, although the mappings for verbs are much different from those for nouns (cf. Table 2), both sets of mapping are diverse without a dominant single mapping. However, for adjectives dependency mapping *amod-amod* accounts for nearly 70% of total mappings. This reflects the fact that the dependency relationship between an adjective and its contextual words is much simpler than that for either a noun or a verb. The dominance of one mapping for adjectives makes the dependency mapping context highly correlated with the dependent word context, thus significantly weakens the effect of dependency mapping, while the diversity of dependency mappings for nouns and verbs ensures its efficacy.

Conclusion and perspectives

In this paper, we propose a bilingual dependency mapping model for bilingual lexicon construction from English to Chinese using a comparable corpus. When calculating the similarity between bilingual words, this model considers both dependent words and their relationships, thus providing more accurate and less noisy representation. Evaluation shows that our approach significantly outperforms a state-of-the-art baseline from English to Chinese on both nouns and verbs in a wide frequency range, though with the exception of adjectives. We also demonstrate that bilingual dependency mappings can be automatically generated and optimized without human intervention, leading to a medium-sized set of dependency mappings, and that their contributions on BLC can be fully exploited via weight learning using a simple yet effective perceptron algorithm, making our approach easily adaptable to other language pairs.

In future work, we intend to apply our method to BLC between other language pairs. Preliminary experiments show that the dependency mapping model can improve the precision for BLC by ~6 units from Chinese to English.

Acknowledgments

This paper is supported by Projects 60970056, 61273320, and 90920004 under the National Natural Science Foundation of China, Project 2012AA0111102 under the "863" National High-Tech Research and Development of China,, and Projects BK2010219, 11KJA520003 under the Provincial Natural Science Foundation of Jiangsu, China.

References

Collins, M., and Duffy, N. (2002). New Ranking Algorithms for Parsing and Tagging: Kernels over Discrete Structures, and the Voted Perceptron. ACL'2002.

Ficšer. D. and N. Ljubešić. (2011). Bilingual Lexicon Extraction from Comparable Corpora for Closely Related Languages. RANLP'2011: 125–131.

Fung, P. (1995). Compiling bilingual lexicon entries from a non-parallel English-Chinese corpus. In the Third Annual Workshop on Very Large Corpora: 173–183.

Fung, P. (2000). A statistical view on bilingual lexicon extraction: from parallel corpora to nonparallel corpora. In the Third Conference of the Association for Machine Translation in the Americas.

Freund, Y. and Schapire, R. E. (1999). Large margin classification using the perceptron algorithm. Machine Learning, 37(3): 277-296.

Garera, N., Callison-Burch, C., and Yarowsky, D. (2009). Improving translation lexicon induction from monolingual corpora via dependency contexts and part-of-speech equivalences. CoNLL'2009: 129–137.

Gong Z. X., Zhang M. and Zhou G. D. (2011). Cache-based document-level statistical machine translation. EMNLP'2011:909-919.

Grefenstette, G. (1998). The Problem of Cross-language Information Retrieval. Cross-language Information Retrieval. Kluwer Academic Publishers.

Haghighi, A., Liang, P., Berg-Krikpatrick, T., and Klein, D. (2008). Learning bilingual lexicons from monolingual corpora. ACL'2008: 771–779.

Helmut, S. (1994). Probabilistic part-of-speech tagging using decision trees. In Proceedings of the International Conference on New Methods in Language Processing (NeMLaP): 44–49.

Kaji, H., Tamamura, S. and Erdenebat, D. (2008). Automatic construction of a Japanese-Chinese dictionary via English. LREC'2008: 699–706.

Koehn, P. and Knight, K. (2002). Learning a translation lexicon from monolingual corpora. In Proceedings of ACL Workshop on Unsupervised Lexical Acquisition.

Lin, D. and Pantel, P. (2002). Concept Discovery from Text. COLING'2002: 42–48.

Mann, G. S., and Yarowski, D. (2001). Multipath translation lexicon induction via bridge languages. NAACL'2001: 151-158.

Marneffe, M.-C. de, MacCartney, B., and Manning, C. D. (2006). Generating typed dependency parses from phrase structure parses. LREC'2006.

Mausam, S. Soderland, O. Etzioni, D. S. Weld M. Skinner, and J. Bilmes. (2009). Compiling a Massive, Multilingual Dictionary via Probabilistic Inference. ACL'2009: 262–270.

Morin, E. and Prochasson, E. (2011). Bilingual Lexicon Extraction from Comparable Corpora Enhanced with Parallel Corpora. In Proceedings of the 4th Workshop on Building and Using Comparable Corpora, ACL'2011: 27–34.

Och, F. J. and Ney, H. (2000). Improved Statistical Alignment Models. ACL'2000: 440-447.

Och, F. J. and Ney, H. (2003). A Systematic Comparison of Various Statistical Alignment Models. Computational Linguistics, 29(1): 19-51.

Rapp, R. (1999). Automatic identification of word translations from unrelated English and German corpora. ACL'99: 519–526.

Schafer, C. and Yarowsky, D. (2002). Inducing Translation Lexicons via Diverse similarity Measures and Bridge Languages. CoNLL'2002.

Shezaf, D. and Rappoport, A. (2010). Bilingual Lexicon Generation Using Non-Aligned Signature. ACL'2010: 98–107.

Tanaka, K. and Umemura, K. (1994). Construction of a bilingual dictionary intermediated by a third language. COLING'94: 297-303.

Toutanova, K. and Manning, C. D. (2000). Enriching the Knowledge Sources Used in a Maximum Entropy Part-of-Speech Tagger. EMNLP/VLC'2000: 63-70.

Wu, D. and Xia, X. (1994). Learning an English-Chinese Lexicon from a Parallel Corpus. In Proceedings of the 1st Conference of the Association for Machine Translation in the Americas.

Yu, K. and Tsujii, J. (2009). Extracting bilingual dictionary from comparable corpora with dependency heterogeneity. NAACL-HLT'2009: 121–124.

Zhang, Y., Sun, L., Li, F. et al. (2006). Bilingual Dictionary Extraction for Special Domain Based on Web Data (in Chinese). Journal of Chinese Information Processing, 2006, 20(2): 16-23.

A MWE Acquisition and Lexicon Builder Web Service

Valeria Quochi[1] *Francesca Frontini*[1]
Francesco Rubino[2]
(1) ILC CNR, Pisa, Italy
(2) Synthema, Pisa, Italy
valeria.quochi@ilc.cnr.it, francesca.frontini@ilc.cnr.it,
francesco.rubino@synthema.it

ABSTRACT
This paper describes the development of a web-service tool for the automatic extraction of Multi-word expressions lexicons, which has been integrated in a distributed platform for the automatic creation of linguistic resources. The main purpose of the work described is thus to provide a (computationally "light") tool that produces a full lexical resource: multi-word terms/items with relevant and useful attached information that can be used for more complex processing tasks and applications (e.g. parsing, MT, IE, query expansion, etc.). The output of our tool is a MW lexicon formatted and encoded in XML according to the Lexical Mark-up Framework. The tool is already functional and available as a service. Evaluation experiments show that the tool precision is of about 80%.

KEYWORDS: Multiword extraction, lexical resources, LMF, web services.

Proceedings of COLING 2012: Technical Papers, pages 2291–2306,
COLING 2012, Mumbai, December 2012.

1 Introduction

Multi-word expressions (MWEs) nowadays still pose problems to most language technology and applications, e.g information retrieval, text mining, semantic web. In particular, they impact greatly on the performance of Machine Translation systems and automatic dictionary compilation. In Rule-based MT, many translation mistakes can be attributed to inadequate handling of MWEs. As for Statistical MT, they constitute a problem in the word alignment of parallel corpora. Especially for such applications MWEs can be thought of as a group of words constituting a single meaning unit and often they have an unpredictable, non-literal translation. From a practical point of view, if a group of words cannot be translated using a word-by-word translation, then it is treated as a MWE. If not recognized and handled properly, MWEs will result in mis-translations (or literal translations), and because many of them have an opaque meaning, i.e., the meaning of the unit cannot be derived by the meaning of the individual constituents that make up the unit, the translation will be incorrect and not understandable, thus hampering the overall text readability (see e.g. (Monti et al.), (Bilal, 2005)).

While many experiments on methods for the automatic acquisition of MWEs have been carried out in academic research for some decades now, readily available, and possibly customizable tools for the acquisition of MWEs in languages other than English are not so popular.

In this paper we describe the implementation of a tool for the automatic creation of MWE lexicons, integrated as a web service into a distributed platform, within the context of the PANACEA project[1]. As its goal is to set up a platform for the automatic acquisition of language resources and involves handling large or massive data, the tool described here has been implemented using robust and "computationally light" methods. The purpose is not so much to devise a new or innovative method (as the state-of-the-art seems to perform sufficiently well), but to provide a free to use tool that creates a full lexical resource from web crawled data: multi words with relevant and useful attached information that can be used for further processing (parsing, MT, information extraction, query expansion and the like). Our core extraction algorithm is language independent, making use of positional information only, and does not require a list of words to be used as seeds. The tool however has been tested and evaluated on Italian only[2].

2 Related works

Given the importance of MWEs for NLP applications, much research has been conducted for their automatic acquisition, with the aim of building or expanding lexica, i.e. mainly in support of lexicographic activities both for general and specific domains.

Common statistically-based acquisition approaches usually involve the following two steps: (1) the identification of candidates (usually based on *n*-grams or pattern matching)(2) filtering and candidate ranking according to some statistical score and an experimental threshold. Along these lines, different methods have been proposed in the existing literature. The oldest and simplest approaches made use of plain text corpora and identified candidates on the basis of (positional) n-grams (sequences of n adjacent words), optionally using POS filtering to clean the candidate lists and stop word lists to reduce the search space (e.g. (Choueka, 1988), (Smadja, 1993)). Also, the extraction is generally performed for a given set of seed words.

[1] see www.panacea-lr.eu

[2] In fact, recently it has been successfully run on a test Spanish corpus and it correctly produced an output lexicon, which however has not been evaluated.

More recent methods make use of tagged or parsed corpora to first identify relevant patterns in the attempt to improve precision, although for the latter the improvement is not clearly proven due to parsing errors (cfr. (Baldwin, 2005), also see the interesting review in (Seretan and Wehrli, 2009, 73-74)). The filtering and ranking of candidates is then achieved by applying some association measure (hereafter AM) calculated on the basis of co-occurrence frequency of the content words involved in candidates. Some of the most commonly used AMs are: (Pointwise) Mutual Information, Dice, Pearson's chi-squared, Log-Likelihood Ratio, Odds Ratio, Fisher's Exact tests, and various entropy measures. Several works have also carried out detailed comparisons of the methods used in the literature, evaluating the association measures used (among others, (Pearce, 2002),(Evert, 2004), (Evert and Krenn, 2005), (Pecina, 2010)). From these, we understand that the efficacy of a given AM may depend on factors like the language being analysed, the size of the corpus and the type of MWE that has to be identified. Overall, it seems that the simplest measures (frequency, MLE and Log Likelihood) perform best[3].

Although much of the work is on English data, research on MWE extraction has been carried out also for other languages, such as German (Krenn and Evert, 2001), Dutch (Villada Moiron, 2005), Czech (Pecina, 2010), French (Laporte et al., 2008), Portuguese (Villavicencio et al., 2010), among others. For Italian, the first work on collocation extraction used a window method for identifying candidates in a plain text corpus and Mutual Information for ranking (Calzolari and Bindi, 1990). Recent efforts towards the acquisition and/or production of MWE lexica for Italian are (Zaninello and Nissim, 2010), (Bentivogli and Pianta, 2002), and (Bonin et al., 2010). The former is an effort for the creation of a database of Italian MWEs annotated according to their morpho-syntactic pattern, where MWEs were given from pre-existing dictionaries. (Bentivogli and Pianta, 2002) extract from the Collins English-Italian dictionary *hidden* MWEs, i.e. MWEs not explicitly marked as such in the dictionary. (Bonin et al., 2010) extract MW terminology for two domains, adopting a contrastive approach in order to identify domain-specific multi-word terms.

Notwithstanding the vast literature, often evaluation is either not reported or not detailed enough. Also, it is often stressed that a righteous comparison of the performance is impossible due to the differences in: methods applied, targeted MWEs types, corpus used, and evaluation methodology (cfr. (Rayson et al., 2010, 3)). Precision and recall may vary considerably: for example Smadja's reports a precision of 80% for his XTRACT system on English texts (but of 40% before the syntactic-based filtering), with evaluation carried out by manual inspection by a lexicographer. (Seretan and Wehrli, 2009, 80) performed experiments in 4 languages and reported different figures for precision (English=0.42-0.58, Spanish=0.39-0.42, French=0.46-0.35, Italian=0.32-0.37).

3 The Panacea MW acquisition tool

Because of the need to operate in a web service distributed environment, where processing time is critical, and because of possible computing and memory limitations, it was decided to avoid computationally intensive methods. Also, as often reported in the literature, simpler methods seem to perform equally well, if not even better, in little constrained set-ups. Our MWE acquisition component is thus inspired by the seminal work by (Smadja, 1993), but integrates also more recently experimented statistical methods and association measures for filtering and ranking the acquired candidates and thus for producing a cleaner output lexicon, promoting

[3]For lack of space, we will not refer here to more sophisticated approaches that aim at measuring the degrees of fixedness and/or opacity, such as e.g. (Fazly et al., 2009).

precision over recall. The input data is part-of-speech tagged corpora in CoNLL format. The tool performs a sequence of steps each implementing different methods, in such a way as to be efficient in terms of processing time and memory usage. These steps are collocation extraction, pre-filtering and ranking by association measures, pattern extraction, pattern selection and lexicon building. In the following sections a description of each step is given.

3.1 Step 1: Window based collocation extractor

The search function requires no lemmas but the POS tags of the pair of tokens representing the first and last component of a multi-word, then all instances of the POS pair in the given window are retrieved. Because all words with the given POS tag will be retrieved, differently from other approaches (e.g (Smadja, 1993)), we only consider the right window. Both the POS tag pairs and the window size are passed as user-configurable parameters to the system. The output of this step is a list of candidate collocation pairs, with their related frequencies.

3.2 Step 2: Pre-filter and collocation ranking through association measures

A pre-filtering stage is applied based on the raw frequency distribution of the collocations for reducing memory load and discarding pairs that will provide no useful statistical evidence. This filters out pairs below a given proportional threshold. Assuming a zipfian distribution for collocation frequencies, a long tail of low frequency pairs and hapaxes[4] will normally be extracted, that needs to be filtered away. So far our algorithm allows for two kinds of filtering based on two different thresholds.

With the AverageFrequency PreFilter the threshold is set to (1):

$$\theta_{AF} = \overline{X} = \frac{\sum_{i=1}^{n} f_i}{n} \tag{1}$$

where f_i is the frequency of the ith collocation/pair and n is the number of collocations extracted. Thus the AverageFrequency PreFilter filters away all collocations with f below the mean $\overline{X}$. The MaxFrequency PreFilter instead sets the threshold to (2):

$$\theta_{MF} = \frac{f_1}{10} \tag{2}$$

where f_1 is the frequency of the most frequent collocation. This latter filter is more selective, in that it discards all pairs that have frequency less than 1/10th of the most frequent pair.

Notice that both thresholds are independent of the size of the corpus. Given the zipfian distribution for each extraction, independently from the corpus, they should identify more or less the same point in the distribution curve. As pre-filter (2) was observed to be too selective for our purposes, it will not be used in the experiments described in the rest of the paper.

After applying the pre-filter, various state-of-the-art Association Measures (AMs) can be calculated, that will be used for alternative rankings of the collocations and of the subsequently acquired full MWEs. Currently Log Likelihood and Pointwise Mutual Information have been implemented.

[4] By hapaxes here we mean word pairs of frequency 1.

3.3 Step 3: Pattern extraction

For each collocation in Step 2 the algorithm retrieves the complete patterns of tokens (i.e. word sequences of minimum length 2 and maximum the window size) with their raw and relative frequencies and the attached information about lemma and POS.

The rationale behind this step is to retrieve all possible intervening patterns between word A and word B. Thus, for each word pair AB, the intervening patterns are collected in a data structure that contains lemma, token and POS for each position, including A and B. This is what we shall call the “set of patterns" for the given collocations.

A distinct pattern will thus be a sequence of elements where each element is a combination of Tokens+Lemma+PoS. Frequencies for each pattern are also retrieved.

For instance, for the pair GAS-SERRA (‘gas-greenhouse’, combined frequency: 10353) the algorithm retrieves several patterns such as[5]:

a) *gas serra* (‘greenhouse gas’): frequency 7151
b) *gas ad effetto serra* (‘greenhouse effect gas’): frequency 1547
c) *gas a effetto serra* (‘greenhouse effect gas’): frequency 1365
...

3.4 Step 4: Pattern based collocation filtering and MWE selection

Now, the collocations retained after pre-filtering go through an additional filtering step where further collocations are discarded based on the distribution of their “set of patterns". More specifically the frequencies of all patterns in the sets are treated as vectors. The goal of the algorithm is to detect whether the vector has any significantly more frequent items; if not, the collocation itself is discarded, otherwise only the significant patterns (i.e. the outliers) are retained as good MWE candidates.

For example: GAS - SERRA produces a pattern vector *v1*:
v1 = [7151,1,1,1,1,51,21,2,43,1,1,38,1547,1,1,10,1,2,4,1,5,1,1,1,1,1,1,14,1,1,6,5,71,1365]

where all elements of *v1* are the frequencies of patterns extracted for GAS-SERRA. So, *v1* contains three clearly outstanding patterns for this collocation (with frequencies 7151, 1547, and 1365, corresponding to the above listed patterns A, B, C).

On the other hand, the collocation MARE-COSTA (‘sea - coast’) produces a long vector as in v2:
v2 = [1,1,1,1,8,1,1,1, ...]

with less clearly recognizable outstanding patterns. Given our approach, this can be seen as evidence for the lower fixedness of the second collocation with respect to the first, and thus as a criterion for rejecting the collcation altogether (i.e. the collocation and its set of patterns).

In order to quantify this intuition, mean ($\overline{X}$) and standard deviation (σ) are calculated for each set of patterns.

$$\sigma = \sqrt{\frac{\sum_{i=1}^{k}(f_i - \overline{X})^2}{k}} \tag{3}$$

[5]Here only the sequence of tokens is given

where f_i is the frequency of the ith pattern k is the number of patterns extracted.

We empirically assessed that only collocations whose vectors show $\sigma > 1$ have some chance of producing at least one significant pattern. This excludes those collocations that are normally made up by long series of very low frequency items, differing from each other very little in frequency. The only exception is when $\sigma = 0$ because the collocation extracts just one pattern of identical frequency. In this case the algorithm selects the pattern as a good MWE without further analysis[6].

Now that we have filtered out a lot noise and irrelevant collocations, the algorithm has to select the good patterns to be encoded in the output lexicon. This is done by using the same variance analysis of the distribution in the sets of patterns described above with the aim of selecting more than one relevant pattern. In particular, the presence of outliers is evaluated in terms of standard deviation above the mean. Empirical evidence showed that one σ above the mean is a good enough threshold in our case. This threshold normally extracts 1-3 significant MWEs per collocation.

Thus for each collocation A+B the algorithm (which we call henceforth SigmaPatternExtraction) runs as follows (figure 1):

```
f(A+B), the frequency of a collocation (A+B)
v(A+B), the vector of pattern frequencies for each collocation
p_i(A+B), a given pattern i of A+B
f(p_i(A+B)), the frequency of p_i(A+B)
θ = 1, c = 1

if f(A+B)== f(p(A+B))
      then return p_i(A+B)
elseif σ of v(A+B) > θ
      for each p_i(A+B)
        if the f(p_i(A+B)) > c * σ(v(A+B)) + X̄(v(A+B))
          then return p_i(A+B)
```

Figure 1: Outline of the SigmaPatternExtraction Algorithm.

Higher values for both θ and c result in more filtering and, possibly higher precision / lower recall.

Notice how this approach differs from the one used in (Smadja, 1993)). There the task was searching for collocates on the basis of a given list of words. Thus the vector was built in order to determine the position of any word W with respect to a word from the list. Thus, in Smadja's approach a high standard deviation indicates randomness, and therefore low association strength between the two words.

In our case the pair, not a single word, is considered. Consequently, the only case when a low sigma is indicative of a good candidate pair is when σ equals zero because its set of patterns actually contains only one element. Randomness however is null in this case. In all other cases, good pattern vectors will contain a strongly uneven distribution, with ideally one or few very frequent patterns and a long tail of very low frequency elements.

In order to evaluate the SigmaPatternExtraction Algorithm a simpler method was also devised (henceforth FirstPatternExtraction, figure 2), which consists in extracting only the most frequent

[6]This is the case for very fixed MWEs such as *stati membri*, "member states"

pattern for each collocation.

```
f(p1(A+B)), the highest value in v(A+B)

if f(A+B)== f(p(A+B)), then return p(A+B)
else return p1(A+B)
```

Figure 2: Outline of the FirstPatternExtraction Algorithm.

Notice that, so far, the system is language independent, as only distributional information is used[7]. Additional steps may be added as post-filters for further pruning the results. Such post-processors might be built ad-hoc for the language/domain in use: e.g. by using lists of stop words, or special heuristics to deal with language or tagger specific issues.

3.5 Step 5: The lexicon builder

The final step of the tool is lexicon building, that compiles the MWEs that were selected according to the steps/filters described above into a full XML-encoded lexicon, that conforms to the LMF standard (Gil Francopoulo, 2006)[8]. Figure 3 below exemplifies the representation of an entry[9].

```
<LexicalEntry id="0">
  <feat att="entryType" val="Multiword"/>
  <feat att="MWEPattern" val="S+E+S"/>
  <feat att="logLikelihood" val="110242.74923578261"/>
  <lemma>
    <feat att="writtenform" val="datore di lavoro"/>
  </lemma>
  <ListOfComponents>
    <Component entry="idEntry_datore">
      <feat att="rank" val="0"/>
      <feat att="pos" val="noun"/>
      <feat att="lemma" val="datore"/>
      <feat att="writtenform" val="datore"/>
    </Component>
    <Component entry="idEntry_di">
      <feat att="rank" val="1"/>
      <feat att="pos" val="prep"/>
      <feat att="lemma" val="di"/>
      <feat att="writtenform" val="di"/>
    </Component>
    <Component entry="idEntry_lavoro">
      <feat att="rank" val="2"/>
      <feat att="pos" val="noun"/>
      <feat att="lemma" val="lavoro"/>
      <feat att="writtenform" val="lavoro"/>
    </Component>
  </ListOfComponents>
</LexicalEntry>
```

Figure 3: Example of the encoding of an extracted MWE entry following LMF-XML.

At the entry level we record information on its type (MWE or NE[10]), on the POS pattern it instantiates, on the frequency and the Log Likelihood measure calculated during the acquisition process. Each entry is then characterized by a specification of the components forming the

[7] The tool is still inevitably format and tag set dependent. Still, we believe it is fairly general: the input data format is in fact CoNLL, a widely used de-facto standard, and the initial search input is a pair of POS tags which is passed as an external parameter set by the user.

[8] In fact, it is valid according to the LMF DTD-rev16.

[9] Notice that, for the sake of readability of the example, the single word LexicalEntries corresponding to each component have been omitted.

[10] NEs can be identified on the basis of simple heuristics exploiting the tagging of proper names, if available in the POS tagged corpus.

MWEs with related information: information about the rank or position of the components, their POS, orthographic form and lemma.

3.6 Requirements and deployment as a Web service

The tool was developed as a java application and it runs in less than 30 minutes any recent server, but it requires a maximum heap memory of 4GB for a corpus of 37 million words (250.3 MB) with window = 5. This is due to the remarkable size of the data structure containing the pairs and their intervening patterns before pre-filtering. Pre-filtering is therefore highly necessary in order to reduce the data-structure to a tractable size for further processing. The tool is then deployed as a web service using Soaplab on Apache Tomcat[11]. Services can be chained together and run as work-flows using a work-flow manager such as Taverna. A typical work-flow for our extractor will include a POS-tagger and a converter to CoNLL[12], as well as the extractor [13]. Post-filters might be developed as stand-alone services and chained to the MW_Extractor to obtain higher quality results.

4 Evaluation

In order to tune the extraction steps and to evaluate the tool, experiments have been performed on a domain specific corpus of Italian: the domain is ENVIRONMENT, one of the domains targeted in the PANACEA project. The tool is evaluated according the standard intrinsic evaluation procedure, against a reference resource that we will refer to as the gold standard.

Evaluation is split into three phases: 1) evaluation of the pre-filtering phase, where the removal of the long tail of low frequency co-occurrences is justified, followed by 2) the evaluation of the pattern extraction algorithm: automatic evaluation of the extracted patterns against the gold standard with the standard precision, recall and F-measures; 3) manual inspection of false positives.

This last step is necessary since, unsurprisingly, the gold standard is incomplete in terms of coverage at different levels: not only are good (domain) multi-word expressions missing, but in some cases, as an analysis of the false negatives has shown, the multi-word acquired from the corpus occurs in a different form than in the gold standard (usually the citation/lemmatised form). A manual exploration of the false positive results is thus necessary. As we are acquiring by POS patterns and not by a list of words, we are likely, and hopefully, extracting multi-word terms that are not present in existing resources, or were not considered interesting domain terms for the given glossary, but which may well be interesting and useful for NLP applications, and especially MT.

In the following sections we describe the data, the experiments and the results obtained.

4.1 The corpus and targeted MWEs

The (Environment) corpus used in the experiments was automatically produced with focussed monolingual crawling and cleaning services within the project[14]; its size is of about 37 million word tokens.

[11] `http://langtech3.ilc.cnr.it:8080/soaplab2-axis/services/panacea.estrattore_mw`

[12] See also (Rubino et al., 2012)

[13] The work-flows can be found on the PANACEA work-flow registry `http://myexperiment.elda.org/workflows/`

[14] http://registry.elda.org:3001/services/160, http://registry.elda.org/services/158

The evaluated extraction, henceforth called SIGMA extraction, was carried out by using the following parameters:
target = extraction of nominal Multiwords, i.e. multiwords whose first and last word is a noun (N-N henceforth)
window = 5 tokens including the first and last element (i.e. the extracted MWEs have a maximum length of 5 words)
prefilter = AverageFrequency PreFilter as in equation (1)
pattern extraction = using the SigmaPatternExtraction as in figure (1)

This extraction will be evaluated against a simpler one, henceforth FIRST extraction, where:
pattern extraction = uses the FirstPatternExtraction as in figure (2)

4.2 The gold standard

A gold standard, or reference resource, for the Environment domain has been created by semi-manually by collecting from several authoritative web glossaries and thesauri relevant nominal Italian MWEs (i.e. N-N MWEs). For each MWEs collected, its frequency in the corpus was computed using simple regular expressions to search for potential morphological variants, and never occurring MWEs are "discarded".

In the gold standard the citation forms were kept as they were found in the given resources. If the same multi-word was present in two sources with two different citation forms - e.g. singular and plural - they were not merged into one single entry in the gold standard nor was their relatedness marked.

4.3 Evaluation of the pre-filtering phase

Before we evaluate the core of the pattern extraction algorithm, it is important to analyze how much is lost in the pre-filtering phase. A MWE can never be extracted with our method if the corresponding collocation (that is the pair of lemmas corresponding to the first and last words of the MWE) is thrown away by the pre-filter. The extraction of the relevant collocation is thus a necessary pre-condition, although not a sufficient one, in that the pattern selection algorithm may then fail to extract the correct pattern, that is the one corresponding to the MWE.

For our evaluation we chose the Average Frequency pre-filter (1). Without this pre-filter our corpus of 37 million words produces 2,046,532 collocations, containing a long tail of hapaxes. With the Average Frequency pre-filter the collocations reduce to 259,848[15].

When evaluated against the gold standard, the non pre-filtered extraction contains 2710 eligible pairs, that is collocations whose first and last word are the same as a gold standard entry[16]. With pre-filtering eligible pairs are reduced to 1746. This may seem a heavy loss, but the 964 lost eligible pairs are to be found among 1,786,684 others. This means that the portion of the extraction that is filtered away (1,786,684 pairs) has a precision in terms of collocations of less than 0.0005 with respect to our gold standard, whereas the density in terms of eligible pairs of the remaining portion (259,848) is, 0.007 that is ten times higher[17]. Since our goal is to

[15]The lowest collocation has frequency 5, the highest has frequency 10,353

[16]The recall in terms of eligible pairs is maximum, which is not surprising, considering that the gold standard contains only MWEs that are present in the corpus.

[17]Although the gold standard is far from complete, these figures can help us getting a general picture of the distribution of our data.

achieve high precision in an acceptable processing time, this loss in recall can be considered acceptable. Notice that an eligible collocation may still not produce a genuine MWE, and that very low frequency collocations defy analysis with any association measure and are discarded in several approaches (Evert, 2004).

Evaluation of ranking is also important, since users may want to take into account only the top portion of the returned MWEs. We shall take ranking into account here, considering that association measures are a property of the collocation rather the MWE, since they are a measure of the association strength between the first and the last element of the MWE.

Two association measures - Pointwise Mutual Information and Log Likelihood - are calculated for each collocation. Collocations can be thus re-ranked by AM as well as by frequency. It is interesting to check which one is the best ranking by calculating the interpolated precision over a complete extraction. No matter what pre-filter is used to extract MWEs from collocations, raw frequency and Log Likelihood produce similar interpolated precision curves (see figure 4); both manage to rank eligible collocations at the top (frequency working slightly better), and both perform significantly better than Pointwise Mutual Information. In the figure the PMI line is almost invisible, being constantly below the baseline and close to zero in the portion taken into account. It is known in the literature (Pecina, 2010) that different kinds of AMs give different results depending on the kind of MWE extraction task. Frequency is also known to perform well (Justeson and Katz, 1995) when filtering on PoS is used.

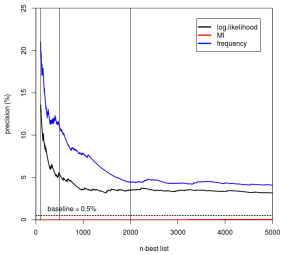


Figure 4: Interpolated precision graph for the different rankings (AverageFrequency pre-filter and SigmaPatternExtraction method). Baseline represents the average precision. The graph shows how precision progresses in the different rankings (only the first 5000 positions are shown).

4.4 Evaluation of MWE extraction

In this section the SigmaPatternExtraction Algorithm is evaluated. Thus two extractions are compared, the SIGMA extraction and the FIRST extraction as described above in 3.4, with FIRST acting as our baseline. Two kinds of evaluations are carried out:

SIMPLE: Simply check the extracted patterns against the gold standard. This evaluation is interesting for the recall; given that the gold standard is incomplete precision is not really significant

REDUCED: Only collocations that could produce patterns in the gold standard are selected

from the full extractions, and all MWEs related to those collocations used for evaluation. This is useful in order to evaluate precision more realistically and to allow an approximate comparison with other approaches described in the literature (see section 2 above on comparability). The "reduction" algorithm is a fairly simple one and implies some stemming, due to the fact that the gold standard is not lemmatised (and that, although UTF-8 is used, some issues related to accented chars still remain) and runs as follows 5:

```
for each mwe of length l in gold-standard
      select w_0 and the w_{l-1}
      remove the last two characters from w_0 and w_{l-1}
      add w_0 + w_{l-1} to eligible-pairs
for each mwe of length l in extraction
      select w_0 and the w_{l-1}
      remove the last two characters from w_0 and w_{l-1}
      if w_0 + w_{l-1} is in eligible-pairs
        add mwe to the reduced-set
evaluate reduced-set against the gold-standard
```

Figure 5: Outline of the reduction algorithm.

For example: if *fondo del mare* is in the gold standard, we search for pairs of the form *fon** - ma***, and return all MWEs that the algorithm has extracted for such pairs in order to evaluate them against the gold standard[18].

The results for both kinds of evaluation and for both methods are given below. Table 1 shows the results for the SIMPLE evaluation.

SIMPLE	**FIRST**	**SIGMA**
test	259848	209471
gold	2191	2191
precision	0.0038	0.0050
recall	0.4505	0.4770
F1	0.0075	0.01

Table 1: Precision and recall for the FIRST and the SIGMA extraction, with a SIMPLE evaluation method. 'Test' and 'gold' show the number of MWEs in, respectively, the evaluated extraction and the gold. While the number of MWEs in the gold remains the same, the number of MWEs in the test changes depending on the kind of algorithm used.

Notice how the test set (the number of MWEs in the extraction that are being evaluated) is smaller with SIGMA, since a number of "low sigma" pairs have been filtered out by the SigmaPatternExtraction Algorithm. Still both precision and recall are increased, that is more good patterns are extracted by not just stopping at the most frequent ones (see Figure 4 for the precision graph). Table 2 shows the results for the REDUCED evaluation.

While in the non-reduced evaluation the SIGMA method was increasing the precision due to the reduction of the test set, in the reduced evaluation FIRST seems to perform better in terms

[18] Notice how this stemming method is quite unsophisticated, and produces more pairs than it should. In this case it would also extract expressions such as *fonti nel mare*, if extracted. This means that the precision figures given for the REDUCED evaluation are probably slightly underestimated.

REDUCED	**FIRST**	**SIGMA**	**SIGMA** +edit	**SIGMA** manual
test	1746	2105	-	-
gold	2191	2191	-	-
precision	0.56	0.50	0.60	0.81
recall	0.45	0.48	0.58	0.60
F1	0.50	0.49	0.59	0.67

Table 2: Precision and recall for the FIRST and the SIGMA extractions, with a REDUCED evaluation method

of precision although SIGMA retrieves a higher absolute number of true positives (1043 vs 984) and thus improves in terms of recall. The reason for this is to be found in a higher test set for SIGMA. This is not surprising if we consider how the REDUCED evaluation is carried out. In this case many of the low sigma pairs that the SIGMA algorithm removes are filtered away by the reduction algorithm also for FIRST; at the same time SIGMA extracts more than one pattern per pair, thus ending up with a higher (reduced) test set. Given that it is infrequent for MWEs from the same collocation (same first and last word) to be present in the gold standard, extracting more than one pattern per collocation, as SIGMA does, is penalizing for the precision with respect to our gold standard. Still, a quick manual check of the false positives reveals that most of the extracted patterns are actually correct, in that they are variants of the first pattern, such as:

```
fonte di inquinamento > fonti di inquinamento
```

Thus, if a more flexible comparison is applied, such as allowing for edit distance (Damerau, 1964) up to 3 between the strings, these variants are recognized as true positives and for SIGMA improves by 10 points, as is shown in the fourth column of table 2.

As shown by the interpolated precision graph for this last evaluation (figure 6), when reducing the set of collocations, association measures actually perform better than simple frequency.

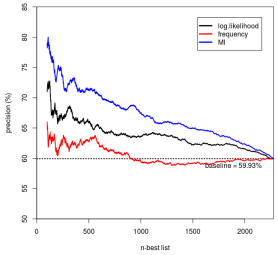


Figure 6: Interpolated precision graph for Pointwise Mutual Information, Log Likelihood and frequency for an extraction with AverageFrequency pre-filter and SigmaPatternExtraction. REDUCED evaluation method; match with edit distance = 3

4.4.1 Manual evaluation of MWE extraction

Further manual inspection of the false positives shows that precision is much higher in fact.

For instance the gold standard contains *zona di pressione* ('pressure zone'), which selects ZONA-PRESSIONE as an eligible pair. This collocation is thus retained in the REDUCED evaluation set, extracting from the corpus *zona di pressione* and *zona di bassa pressione* ('low pressure zone'). The latter is not contained in the gold standard, but is in fact a genuine MWE. By analysing false positives for the REDUCED evaluation and adding the good MWEs found to our gold standard, we obtain a precision of 81% (see table 2).

As the REDUCED evaluation method roughly simulates an extraction from a fixed, predefined set of targeted lemmas, as is usually the case with experiments reported in the papers, it allows for some comparison with other approaches.

Our result is thus in line with the precision performance of Smajda's XTRACT (Smadja, 1993)[19].

However, we observed that with the REDUCED evaluation, much of the noise present in the data (mainly due to the automatically crawled nature of the corpus) was filtered out through the reduction of eligible pairs (and therefore did not impact on the evaluation scores). To arrive at a more realistic assessment of the tool then a manual evaluation of the complete extraction (that is without evaluation filter) was also attempted. Considering the large amount of results obtained by our SIMPLE method, our priority was to verify the precision of the top portion of our extraction. This is meant to ensure that a user who blindly extracts MWEs from a (domain) corpus will get a significant amount of genuine results in the higher ranks. As we have seen when evaluating the pre-filter, our best ranking is raw frequency, followed by LogLikelihood. We thus evaluated the first 1000 highest frequency results against the gold standard, then we checked the false negatives and added such as turned out to be genuine to the gold standard for a second run.

$$\text{first run precision} = 0.40, \text{second run precision} = 0.78$$

The results of the first run tell us that 40% of the MWEs from the original gold are to be found in the top 1000 results. The second run, most significantly, tells us that 78% of the first 1000 returned results are correct, and that it is thus possible to extract MWEs without seed words and still get a precision that approaches the state of the art. Notice also how the "real" precision of the top 1000 portion of the extraction is just 2% lower than the one obtained with the REDUCED evaluation method, the latter thus providing a fairly good approximation for the first.

Conclusion and Future Work

We have presented a tool for the acquisition of multi-word expressions of various lengths and types that generates an LMF MWEs lexicon as output. The tools is already functional and has been deployed as a web service within a distributed platform that deals with large/massive data. First evaluation results (with a reduced extraction that is rendered comparable to the gold standard) are encouraging. If possible, we plan to extend the manual evaluation in order to have a more accurate estimate of the real precision.

Regarding the service, future improvements are possible especially based on users' feedback in particular regarding the properties to be left as configurable parameters and output formats. Regarding the tool, we intend to continue improving the method by:

[19] In fact, our system is likely even to outperform it given that the evaluation carried out is slightly penalizing for the system both because of the naive stemming and because the gold-standard also contains MWEs with frequency 1 and 2, which are most likely not retrieved by the system.

1. Adding further filtering/cleaning of the extracted MWEs. In particular we might experiment with merging patterns that are sub strings of others when they have the same frequency, as they create noise and lower precision. For instance: *sicurezza sul posto* (lit. 'safety on the place') and *sicurezza sul posto di lavoro* (lit. 'safety on the place of work') all have the same frequency (630); clearly, the genuine MWE is *sicurezza sul posto di lavoro* and the others are substrings thereof.

2. Extracting MWEs with PoS patterns by progressive test and reduction of patterns for learning the "free slots" in the multiwords. For instance we may want to derive a pattern of the form *articolo NUM della legge* ("article NUM of the law") from a series of patterns of the form: *articolo 6 della legge, articolo 12 della legge, articolo 23 della legge, ...*

3. Language specific fine tuning will also be implemented, in the form of post-filtering (e.g. via optional stop words list and legitimate patterns check), as well as of post-editing (e.g. with head detection heuristics).

4. Automatic conversion to, or direct output in-, an RDF format (e.g. according to the Lemon model), as to make the resource publishable as L(O)D potentially exploited for reasoning or by other web services[20].

Regarding the evaluation, the next step will be a task based evaluation. Interesting tasks could be: rule-based machine translation, syntactic parsing, or subcategorisation frame acquisition (which would be interesting to assess the impact of automatically acquired multi-word prepositions).

Acknowledgments

This work has been realized at CNR-ILC within the EU FP7 funded project PANACEA (Platform for Automatic. Normalized Annotation and Cost-Effective Acquisition of Language Resources for Human Language Technologies) under grant agreement n. 248064. We thank PANACEA reviewers and the COLING anonymous reviewers for their comments and incentives to improve. In particular, we want to thank Fabio Affé who helped us with the final modifications to the java code and with the deployment in Soaplab.

[20]We thank an anonymous reviewer for his/her suggestions concerning LOD and NIF. Although the NLP Interchange Format (`http://nlp2rdf.org/nif-1-0`) seems to be a format for corpus data, while our tool outputs a lexicon, in our future work within the LOD framework we will attempt to ensure compatibility with NIF in terms of basic vocabulary and data categories used.

References

Baldwin, T. (2005). Deep lexical acquisition of verb-particle constructions. *Comput. Speech Lang.*, 19(4):398–414.

Bentivogli, L. and Pianta, E. (2002). Detecting hidden multiwords in bilingual dictionaries. In EURALEX., editor, *Proceedings of the Tenth EURALEX International Congress*. Center for Sprogteknologi.

Bilal, K. (2005). Extracting Multiword Expressions in Machine Translation from English to Urdu using Relational Data Approach. *ENFORMATIKA International Transactions on Engineering, Computing and Technology*, 6:312–314.

Bonin, F., Dell'Orletta, F., Montemagni, S., and Venturi, G. (2010). A contrastive approach to multi-word extraction from domain-specific corpora. In Calzolari, N., Choukri, K., Maegaard, B., Mariani, J., Odijk, J., Piperidis, S., Rosner, M., and Tapias, D., editors, *Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC'10)*, Valletta, Malta. European Language Resources Association (ELRA).

Calzolari, N. and Bindi, R. (1990). Acquisition of lexical information: from a large textual italian corpus. In *Proceedings of the 13th conference on Computational linguistics - Volume 3*, COLING '90, pages 54–59, Stroudsburg, PA, USA. Association for Computational Linguistics.

Choueka, Y. (1988). Looking for needles in a haystack or locating interesting collocation expressions in large textual databases. In *Proceedings of the RIAO*, pages 38–43.

Damerau, F. J. (1964). A technique for computer detection and correction of spelling errors. *Commun. ACM*, 7(3):171–176.

Evert, S. (2004). The statistics of word cooccurrences: word pairs and collocations. *Unpublished doctoral dissertation, Institut fuer maschinelle Sprachverarbeitung, Universitaet Stuttgart*.

Evert, S. and Krenn, B. (2005). Using small random samples for the manual evaluation of statistical association measures. *Computer Speech and Language*, 19(4):450 – 466. Special issue on Multiword Expression.

Fazly, A., Cook, P., and Stevenson, S. (2009). Unsupervised type and token identification of idiomatic expressions. *Computational Linguistics*, 35(1):61–103.

Gil Francopoulo, Laurent Romary, M. M. N. C. (2006). Lexical Markup Framework (LMF). In *LREC 2006*.

Justeson, J. S. and Katz, S. M. (1995). Principled disambiguation: Discriminating adjective senses with modified nouns. *Computational Linguistics*, 21(1):1–27.

Krenn, B. and Evert, S. (2001). Can we do better than frequency? A case study on extracting PP-verb collocations. In *Proceedings of the ACL Workshop on Collocations*, Toulouse, France.

Laporte, E., Nakamura, T., and Voyatzi, S. (2008). A French Corpus Annotated for Multiword Nouns. In *Proceedings of the Language Resources and Evaluation Conference. Workshop Towards a Shared Task on Multiword Expressions*, pages 27–30, Marrakech, Maroc.

Monti, J., Barreiro, A., Elia, A., Marano, F., and Napoli, A. Taking on new challenges in multi-word unit processing for machine translation. In *Proceedings of the Second International Workshop on Free/Open-Source Rule-Based Machine Translation, net/10609/5646*, year=2011.

Pearce, D. (2002). A comparative evaluation of collocation extraction techniques. In *In Third International Conference of on Language Resources and Evaluation*, Las Palmas, Spain.

Pecina, P. (2010). Lexical association measures and collocation extraction. *Language Resources and Evaluation*, 44:137–158.

Rayson, P., Piao, S., Sharoff, S., Evert, S., and Moirón, B. n. V. (2010). Multiword expressions: hard going or plain sailing? *Language Resources and Evaluation*, 44:1–5.

Rubino, F., Frontini, F., and Quochi, V. (2012). Integrating nlp tools in a distributed environment: A case study chaining a tagger with a dependency parser. In Calzolari, N., Choukri, K., Declerck, T., Doğan, M. U., Maegaard, B., Mariani, J., Odijk, J., and Piperidis, S., editors, *Proceedings of the Eight International Conference on Language Resources and Evaluation (LREC'12)*, Istanbul, Turkey. European Language Resources Association (ELRA).

Seretan, V. and Wehrli, E. (2009). Multilingual collocation extraction with a syntactic parser. *Language Resources and Evaluation*, 43(1):71–85.

Smadja, F. (1993). Retrieving collocations from text: Xtract. *Comput. Linguist.*, 19(1):143–177.

Villada Moiron, B. (2005). *Data-driven identification of fixed expressions and their modifiability*. PhD thesis, University of Groningen.

Villavicencio, A., Ramisch, C., Machado, A., de Medeiros Caseli, H., and José Finatto, M. (2010). Identificação de Expressões Multipalavra em Domínios Específicos. *Linguamàtica*, 2(1):15–33.

Zaninello, A. and Nissim, M. (2010). Creation of Lexical Resources for a Characterisation of Multiword Expressions in Italian. In *LREC 2010*.

A Diverse Dirichlet Process Ensemble for Unsupervised Induction of Syntactic Categories

Roi Reichart[1] *Gal Elidan*[2] *Ari Rappoport*[3]
(1) The Computer Laboratory, University of Cambridge, UK
(2) Department of Statistics, The Hebrew University
(3) Institute of computer science, The Hebrew University
Roi.Reichart@cl.cam.ac.uk, galel@huji.ac.il, arir@cs.huji.ac.il

ABSTRACT
We address the problem of unsupervised tagging of phrase structure trees with phrase categories (parse tree nonterminals). Motivated by the inability of a range of direct clustering approaches to improve over the current leading algorithm, we propose a mixture of experts approach. In particular, we tackle the difficult challenge of producing a *diverse* collection of useful tagging experts, which can then be aggregated into a final high-quality tagging. To do so, we use the particular properties of the Dirichlet Process mixture model. We evaluate on English, German and Chinese corpora and demonstrate both a substantial and consistent improvement in overall performance over previous work, as well as empirical justification of our algorithmic choices.

KEYWORDS: Unsupervised parsing, Grammar induction, Non terminals, Dirichlet Process, Ensemble learning.

Proceedings of COLING 2012: Technical Papers, pages 2307–2324,
COLING 2012, Mumbai, December 2012.

1 Introduction

Grammar induction is the task of learning grammatical structure from plain text without human supervision. The task is valuable for the understanding of human language acquisition and its output can potentially be used by NLP applications, avoiding the costly and error prone creation of manually annotated corpora. The task has been widely explored (Klein, 2005) and its importance has increased due to the recent availability of huge corpora.

The induced grammar can be represented in various ways. Most work (e.g., (Klein and Manning, 2004; Smith and Eisner, 2006; Seginer, 2007; Headden et al., 2009)) annotate text sentences using an unlabeled hierarchical phrase or a dependency structure, and thus represent the induced grammar through its behaviour in a parsing task.

An important task in theory and practice that we consider in this work is how to enrich phrase structures with syntactic categories. The two grammars that have been widely explored by the NLP community in the last two decades, phrase structure grammars and dependency grammars, allow the induced structure to be either labeled or not and use the labeling to describe substantially different syntactic functions. In this paper we focus on the former formalism and induce parse tree nonterminals (e.g., 'NP', 'VP', 'PP') in an unsupervised manner. We keep the discussion of dependency parsing for future research.

Many linguistic theories posit a hierarchical labeled constituent (or constructional) structure, arguing that it has a measurable psychological reality (e.g., (Goldberg, 2006)). Practically, most of the syntactic annotation of corpora used by the NLP community comes in the form of labeled structures. Indeed, modern *supervised* syntactic parsers aim at learning labeled structures. Moreover, phrase categories are often used in a variety of NLP tasks, such as SRL (Gildea and Jurafsky, 2002; punyakanok et al., 2008), alignment in syntax-based machine translation (Zhang and Gildea, 2004), information extraction (Miyao et al., 2008), etc.

Phrase categories can be induced either jointly with the phrase structure (Haghighi and Klein, 2006) or given a previously induced structure (Borensztajn and Zuidema, 2007; Reichart and Rappoport, 2008). Reichart and Rappoport (RR08), which has the leading results, uses a two stage approach where the second stage clusters the phrases of the parse trees induced by an unsupervised parser (Seginer, 2007). This is done by inducing an over-expressive large number of categories using the BMM model of Borensztajn and Zuidema (2007), and then clustering these categories into a final set.

In this work we focus on improving the critical last stage of narrowing down the large number of induced BMM labels into a smaller set of informative categories[1]. Naively, one might think that simply replacing the simple clustering algorithm used by RR08 with a more elaborate approach would result in improved final categories. However, as we show in Section 3, a variety of clustering approaches did not lead to a noticeable improvement.

To overcome this difficulty, we adopt a qualitatively different solution and tackle this task using a multiple experts approach (Dietterich, 2000). Intuitively, averaging over multiple predictions (experts) is useful when the output of the various experts obeys two requirements: (1) the output of *each* expert is of useful quality; (2) the experts are sufficiently different from each other.

[1] It is also possible to directly cluster phrases without first inducting BMM categories. This, however, leads to inferior overall results which we do not report for clarity.

The central challenge in building such ensembles is in ensuring that different experts capture different characteristics of the problem. To do so, we build on the Dirichlet Process Mixture Model (DPMM) (Ferguson, 1973; Antoniak, 1974) which relies on the Bayesian framework to induce an a posteriori clustering. Each DPMM expert is characterized by a concentration parameter, and we vary this parameter to induce different experts (clusterings). As our experiments show, it is the particular properties of the DPMM that lead to a diverse ensemble.

Finally, with a diverse ensemble at hand, we aggregate the experts into a single coherent phrase structure tagging. Qualitatively, the tendency of two phrases to share a category should increase with the number of DPMM experts that independently cluster them together. We formalize this idea as a global optimization problem and use the k-way normalized cut algorithm (Yu and Shi, 2003) to solve it.

We evaluate our algorithm on English, German and Chinese, using various tag set sizes and evaluation measures. Our results justify our reliance on DPMM and normalized-cut, and demonstrate consistent improvement over previous work.

2 Previous Work

Unsupervised parsing attracts researchers for many years (see reviews in (Clark, 2001; Klein, 2005)). In recent years efforts have been made to evaluate the algorithms on manually annotated corpora such as the WSJ PennTreebank (Klein and Manning, 2002, 2004; Dennis, 2005; Bod, 2006; Smith and Eisner, 2006; Seginer, 2007; Cohen and Smith, 2009; Headden et al., 2009; Berg-Kirkpatrick and Klein, 2010; Blunsom and Cohn, 2010; Gillenwater et al., 2010; Spitkovsky et al., 2010a,b, 2011b,a). All these works induce unlabeled phrase or dependency structures.

In this paper we focus on the induction of syntactic categories for unlabeled phrase structures (parse tree nonterminals) and its evaluation on corpora annotated with a similar representation. There are three previous papers we are aware of that address this problem. Haghighi and Klein (2006) presented two models: $PCFG \times NONE$ and $PCFG \times CCM$. These models use the inside-outside and EM algorithms to induce bracketing and labeling simultaneously [2].

Borensztajn and Zuidema (2007) presented the Bayesian Model Merging (BMM), a framework for inducing a PCFG containing both a bracketing and a labeling. They use Stolcke's algorithm (Stolcke, 1994; Stolcke and Omohundro, 1994) with features from Petasis et al. (2004). The BMM model uses an iterative greedy search for an optimal PCFG according to the Bayesian criterion of maximum posterior probability. The data likelihood is proportional to the data description length according to the grammar and the prior distribution on the grammar is proportional to the grammar description length.

The number of categories induced by the BMM model is very large. For example, for the 7422 sentences of the WSJ10 corpus it induces 4944 categories. To enable generalization over the unlabeled bracketing, a much smaller number of final categories should be induced.

RR08 proposed a clustering algorithm for the BMM categories. They map each of the BMM categories to one of the R most frequent categories produced by the algorithm. Frequency was determined according to the number of phrases each BMM category tags. To perform the mapping, they construct a feature vector representation for each BMM category. The vector consists of $3|M| + |S|$ features, where M is the set of BMM categories and S is the set of POS

[2] Their other models, which were the core of their paper, are semi-supervised.

tags in the corpus. For each category l, they compute the cosine metric between its vector and that of every category among the R most frequent BMM categories. l is mapped to the category with which it obtains the highest cosine score.

RR08 applied their clustering scheme to the bracketing produced by the unsupervised parser of Seginer (2007). The labeled phrase structure trees induced by RR08 are better than those induced by (Haghighi and Klein, 2006) and (Borensztajn and Zuidema, 2007) which motivates separate learning of the phrase structures and their categories. In this work we provide an alternative algorithm for BMM categories clustering. We provide a detailed comparison to the work of RR08 and show superior results.

Sangati and Zuidema (2009) proposed head assignment algorithms. The concept of a head is related to syntactic categories. Their algorithms are trained on data without head annotations, but, unlike our unsupervised approach, requires manually created labeled phrase-structure trees as input.

The concept of head naturally connects to dependency parsing (Kubler et al., 2009) which has been extensively studied in the last decade. While the categories assigned to dependency structures are different in nature from the phrase categories explored in this paper, our algorithm may be applicable to this case as well. We keep this question for future research.

DP has been used for unsupervised syntactic acquisition tasks. Finkel and Manning (2007) Used DP for unsupervised POS induction from dependency structures. Liang et al. (2009) proposed a nonparametric Bayesian generalization of PCFG, based on the hierarchical Dirichlet Process, and applied it to supervised parsing. DP has been used for many other NLP tasks as well (e.g. (Goldwater et al., 2006; Johnson et al., 2007; Haghighi and Klein, 2007; Johnson and Goldwater, 2009)). However, we are not aware of works that explored DP as a model for creating a diverse ensemble of experts for clustering tasks.

3 Building a Diverse Clustering Ensemble

As discussed, to induce phrase categories we adopt a two stage approach where we first induce a large number of categories and then narrow this into a final set. The first stage, leading to a collection of BMM categories is similar to RR08. Our novelty is in taking a qualitatively different approach to the critical stage of narrowing down the large number of categories into a small informative set.

Motivation For the Ensemble Approach. To motivate our ensemble approach we must first consider more straightforward alternatives. Probably the simplest one is to use a one stage approach where we cluster phrases directly using several clustering algorithms (K-means, complete link, single link and average link) and distance metrics (Euclidean and cosine). Another alternative, is to keep running the BMM until the desired number of categories is obtained (that is, by selecting the least harmful update of its objective when no further improvement is possible) . In preliminary experiments, these approaches resulted in inferior results to RR08. For example, for WSJ10 with 26 clusters, the best of these algorithms (K-means with cosine distance), achieves F-score with many-to-one mapping of 38.7 compared to 58.9 of RR08 (see Table 2).

Building on the good performance of RR08, we next tried to simply replace its final clustering algorithm (of BMM labels). We tried the K-mean algorithm both with a random starting point and an informed starting point with the cluster centers initialized as the K most frequent BMM

	V	NVI	Many–to–1
DP	0.26	1.53	34.1
	0.23	1.59	37
	0.2	1.68	44.1
KM	0.65	0.74	69.4
	0.6	0.83	65.2
	0.65	0.75	66.8
RR08	0.67	0.71	73.1
	0.64	0.74	71.6
	0.61	0.84	68

Table 1: Average pairwise similarity between the clustering experts induced by the different clustering algorithms we use in this paper. KM is K–means. RR08 is the algorithm of Reichart and Rappoport (2008) run each time with a different number of induced clusters. For each clustering algorithm, the first line is for WSJ10 (English), the second line is for NEGRA10 (German) and the third line is for CTB10 (Chinese). Higher V and Many–to–1 and lower NVI scores imply that the clusterings are more similar. DP produces the least similar clusterings.

categories. We tried both variants with several different cluster set sizes. In both cases, the resulting final clustering was not superior to that of RR08.

Our next step was to adopt a Bayesian approach where multiple clusterings are considered. Concretely, the Dirichlet process mixture model (DPMM) defines a distribution over clusterings that is governed by a concentration parameter α. Given α, to get a specific clustering, the *maximum a-posteriori* clustering is a natural choice. One can also consider defining a prior over α to lessen the arbitrariness of the choice of parameters. We tried DPMM with a uniform prior over α which did not lead to an improvement in the results. Furthermore, individual clusterings for a range of α values, were all of similar quality. Thus, we cannot expect a more informed prior over α to lead to better performance[3].

Ensemble Construction: First Approach. With the above evidence as to the need of an ensemble approach, we are still left with the challenge of constructing diverse clustering experts. Intuitively, if we allow a different number of clusters for each experts we would often get qualitatively different solutions as, for example, the best three cluster solution is typically not a simple refinement of the best two cluster solution. The simplest approach to carry this out is to run different K-means, each with a different number of target clusters. Another possibility is to use the method of RR08, again with a different number of target clusters. Unfortunately, as Table 1 shows, in both cases the similarity of the different experts (clusterings) is quite high. Indeed, as we report in Section 7, this did only result in small improvement of the performance.

Part of the difficulty with this approach for constructing a clusterings ensemble is that both clustering methods used highly depend on the initial (informed) starting point. Unfortunately, as discussed above, starting with random initializations leads the K-means algorithm to low-quality clusterings.

Ensemble Construction: Improved Approach. To overcome the sensitivity to the initial clustering, we consider the Bayesian DPMM framework. Using standard MCMC techniques

[3]The effective range of α values in our experiments is $[10^{-4}, 10]$. We run the DP algorithm 1000 times with α values changed from 10^{-4} to 10 in steps of 10^{-3}.

(e.g., Gibbs sampling), the resulting clustering converges to the mode of the posterior solution, which is *independent* of the initialization of the procedure[4]. We start by describing the DPMM framework and then explain how it can be used to construct multiple diverse clusterings.

The DPMM defines a prior over the number of clusters that can be described via the so-called Chinese restaurant process (CRP). In this metaphor we have a Chinese restaurant with an infinite number of tables (clusters), each of which can seat an infinite number of customers (BMM categories). The first customer enters the restaurant and seats at the first table. When a new customer (BMM category) arrives at the restaurant, s/he either sits at an existing table with probability proportional to the number of customers already seated at the table (cluster size), or at a new table with probability proportional to α. This process defines a coherent probabilistic prior over the number of clusters (Ferguson, 1973).

Formally, let G be a collection of likelihood parameters, sampled for each cluster. Parameters $\theta_{1:M}$, one for each sample, are drawn from G and, finally, observations, $x_{1:M}$ (the BMM categories) are drawn from a distribution associated with the parameters. Each observation x_i is represented by a vector in N^d, and the values in its corresponding parameter vector, $\theta_i \in R^d$, sum to 1 (d is fixed for all observations). The probability of the i-th observation x_i is associated with the i-th parameter θ_i by a likelihood function $F(\theta_i)$. Now, since the number of clusters is unknown a-priori, the DP defines a distribution over G, governed by a base measure G_0 (parameter factory) and the concentration parameter α. The entire generative process is defined by the equations:

$$G|\alpha, G_0 \sim DP(\alpha, G_0)$$

$$\theta_m|G \sim G$$

$$x_m|\theta_m \sim F(\theta_m)$$

Our likelihood function $F(\theta_i)$ is a multinomial (see below) with parameters $\theta_{1:M}$. G_0 is chosen to be the Dirichlet distribution, the conjugate of the multinomial. The DP concentration parameter, α is kept fixed during the clustering process.

Given a DPMM, a concrete clustering is defined by the mode of the posterior distribution. Each cluster is assigned a different parameter from the collection defined by G, and observations (x_i, BMM labels) belonging to the same cluster (final phrase category), share this parameter.

Constructing multiple experts using the DPMM framework is simply done by varying the concentration parameter α. That is, each expert is associated with a different α value, resulting in a different clustering. While the quality of each such expert is not substantially different than RR08, as Table 1 shows, the resulting experts (clusterings) are substantially more varied than the alternatives considered above.

4 Model Averaging

We now face the task of aggregating the individual clusterings induced by the different DP experts. Intuitively, if several experts independently cluster together two BMM categories, our belief that these categories belong in the same cluster should increase.

[4]In practice, there are deterministic effective alternatives to the stochastic Gibbs procedure which are highly effective. In this work we use the one proposed in Daume (2007).

We now formalize this idea using the k-way normalized cut clustering algorithm (Yu and Shi, 2003). Its input is a undirected graph $G = (V, E, W)$ where V is the set of vertices, E is the set of edges and W is an edge weight matrix assumed to be non-negative and symmetric. For $A, B \subseteq V$ define:

$$links(A, B) = \sum_{i \in A, j \in B} W(i, j).$$

Using this definition, the normalized link ratio of A and B is defined to be:

$$NormLinkRatio(A, B) = \frac{links(A, B)}{links(A, V)}.$$

The k-way normalized cut problem is to minimize the links that leave a cluster relative to the total weight of the cluster. Denote the set of clusterings of V that consist of k clusters by $C = \{c_1, \ldots c_t\}$ and the j-th cluster of the i-th clustering by c_{ij}. Then

$$c^* = \underset{c_i \in C}{\operatorname{argmin}} \sum_{j=1}^{k} NormLinkRatio(c_{ij}, V - c_{ij})$$

To apply an algorithm that solves this clustering problem to our task, we construct the input graph G from the clusterings contained in the ensemble. The graph vertices V correspond to the BMM categories and the (i, j)-th entry of the matrix W is the number of ensemble members that cluster the ith and jth BMM categories together.

A low edge weight implies that a small number of ensemble members cluster together the BMM categories represented by the vertices connected by the edge. To reduce noise, we exclude edges whose weight is less than 3 from the graph.

5 Feature Representation

To complete the picture, we now describe the feature representation of the BMM categories. We create for each BMM category a vector $x \in N^{6 \cdot |S|}$ where S is the set of POS tags in the corpus. The first $2S$ features correspond to the appearance of a POS tag in the leftmost/rightmost position of a constituent labeled by the represented BMM category. Specifically, the i-th feature ($i \in \{1, \ldots, |S|\}$) is the number of times the i-th POS tag appears in the leftmost position of a constituent labeled by the represented BMM category, and the $(|S| + i + 1)$-th feature is the number of times that tag appears in the rightmost position of a constituent labeled by the BMM category.

Similarly, the next $2|S|$ features correspond to the appearance of a POS tag in the leftmost/rightmost position of a leftmost sibling of a constituent annotated by the BMM category, and the last $2|S|$ features correspond to the appearance of a POS tag in the leftmost/rightmost position of a rightmost sibling. A constituent $C1$ is defined to be the leftmost sibling of a constituent $C2$ iff $C1$ is an immediate left neighbour of $C2$, and $C1$ and $C2$ have the same parent. A rightmost sibling is defined accordingly.

Note that DP does not force a specific parametric family for the likelihood. Our decision to use a multinomial likelihood function is due to the fact that our features are counts of events.

6 Experimental Setup

Overall Setup. We evaluated our algorithm on English, German and Chinese corpora: the WSJ Penn Treebank, the Negra corpus (Brants, 1997), and version 5.0 of the Chinese Penn Treebank (Nianwen et al., 2002). In each corpus, we used the sentences of length at most 10[5], numbering 7422 (WSJ10), 7542 (NEGRA10) and 4626 (CTB10). We used the gold standard POS tag annotation of these corpora.

To initialize the clustering algorithms, we sort the BMM categories according to the number of constituents they label, and use the most frequent ones. For K-means, this provides an informed starting point, which proved crucial to the performance of the algorithm. For DPMM, although in theory the algorithm does not depend on its starting point, such initialization is helpful in practice[6].

We induce $D = 10$ different experts. For the K-means and RR08 baselines we do so by changing the number of induced clusters from 5 to 50, in steps of 5. For DPMM we use different values of α sampled log-uniformly in the range $[10^{-4}, 10]$ (5 orders of magnitude). The DP search procedure we use is that of (Daume-III, 2007)[7], whose good convergence properties have been demonstrated. The k-way normalized cut code was written by Jianbo Shi[8]. The code of RR08 was provided to us by the authors[9].

Number of Final Categories. As discussed, the quality of the different experts (clusterings) induced by the DPMM was not substantially different than RR08. Thus, despite the fact that the number of clusters for each expert is inferred automatically, we are still faced with the problem of choosing the final number of categories that will be inferred using the expert ensemble. We face the same problem when considering the baseline ensembles.

Reichart and Rappoport (2008) induced for each corpus two sets of clusters. A first set consists of T clusters, where T is the number of gold categories in the experimental corpus. For the second set size they observed that in all three corpora about 95% of the constituents are covered by 23% – 37% of the categories, and the curve rises very sharply until that 95% value. Therefore, the number of clusters in the second set is the number of categories that cover at least 95% of the constituents in the corpus (denoted by P, for *prominent* categories). Following their work, we induce for each corpus T and P categories according to the values they defined.

The specification of syntax annotation schemes, including the number of categories, usually involves arbitrary decisions (see (Klein and Manning, 2003) for an example and its effects on parsing). We thus induce for each corpus 5 different sets of clusters. Two of these are the set consisting of T clusters and the set consisting of P clusters. The other set sizes are the 3 values in $\{5, 10, \ldots 25\}$ that are not the two closest values to T and P (see Table 2).

Evaluation Measures. The induced labels have arbitrary names. To evaluate them against a manually annotated corpus, a proper correspondence with the gold standard labels should be established. We explore two types of evaluation measures, one is based on mapping between the induced and gold labels and one is based on information theory (IT) concepts. All measures

[5] Excluding punctuation and null elements, as in (Klein, 2005) and other previous work.

[6] Note that this is true even when stochastic algorithms are used to infer the clusterings since convergence time *strongly* depends on the starting point.

[7] http://www.cs.utah.edu/∼hal/DPsearch

[8] http://www.cis.upenn.edu/∼jshi/

[9] We thank the authors for letting us use their code.

are based on the co-occurrence matrix between the induced and gold labels defined as follows: given a corpus tagged once with the n_1 gold standard labels and once with the n_2 induced labels, the co-occurrence matrix has $n_1 \times n_2$ and the number in the (i, j)-th entry is the number of times the i-th gold cluster and the j-th induced cluster annotate the same constituent.

We evaluate with two mapping schemes: greedy *many–to–1* and greedy *1–to–1* mappings. In both cases we find the mapping between the induced and gold clusters which maximizes the co–occurrence between the clusterings. In the first mapping two induced clusters can be mapped to the same gold standard cluster, while in the latter each and every induced cluster is assigned a unique gold cluster. Under both mapping schemes, if the number of induced clusters is lower than the number of gold clusters, there will be gold clusters to which no induced cluster is mapped. Computing the greedy 1–to–1 mapping is equivalent to finding the maximal weighted matching in a bipartite graph, whose weights are given by the co-occurrence matrix. We use the Kuhn–Munkres (Kuhn, 1955; Munkres, 1957) algorithm to solve this problem.

For these measures, we follow the previous works and apply labeled parse trees evaluation by first mapping the induced labels to the gold labels and then computing the standard labeled parsing F–score [10]. While the labeling accuracy after mapping is not explicitly given, it can computed by dividing the unlabeled F–score with the labeled F–score.

The IT based measures provides a way to evaluate the induced clustering without performing a direct mapping to the gold standard. They are based on the observation that a good clustering reduces the uncertainty of the gold standard cluster given the induced cluster and vice-versa. Several such measures exist, we use two widely–accepted ones, the **V** (Rosenberg and Hirschberg, 2007) measure and the **VI** (Meila, 2007) measure.

The V measure is defined as follows:

$$V = \frac{2hc}{h+c}$$

$$h = 1 - \frac{H(G|T)}{H(G)}, c = 1 - \frac{H(T|G)}{H(T)}$$

For the VI measure, we report its normalized version, **NVI**. NVI and VI induce the same order over clusterings but NVI values for good clusterings ranges in $[0, 1]$ (Reichart and Rappoport, 2009). The NVI measure is defined to be:

$$NVI = \frac{H(G|T) + H(T|G)}{H(G)}$$

Note that V scores are in [0, 1] and the higher the score, the better the clustering. For NVI, the scores are non-negative and lower scores imply improved clustering quality. We use e as the base of the logarithm. Many other clustering evaluation measures exist. The ones we use here are well accepted in the literature. For a recent review see (Reichart and Rappoport, 2009).

7 Results

In this section we demonstrate the effectiveness of our DPMM ensemble for the task of unsupervised induction of syntactic categories (parse tree non-terminals) for three different languages. We start by demonstrating an overall and consistent improvement over RR08 and then provide evidence that justify the specific algorithmic choices.

[10] $f = \frac{2*LP*LR}{LP+LR}$, LP and LR are labelled precision and recall.

	English, WSJ10, IT measures									
	$\|C\| = 5$		$\|C\| = 8(P)$		$\|C\| = 15$		$\|C\| = 20$		$\|C\| = 26(T)$	
	NVI	V	NVI	V	NVI	V	NVI	V	NVI	V
DP+NC	**0.98**	**0.5**	**1.02**	**0.5**	1.2	0.47	**1.22**	**0.47**	**1.3**	**0.47**
RR08	1.2	0.38	1.15	0.4	**0.89**	**0.51**	1.33	0.44	1.44	0.44

	German, NEGRA10, IT measures									
	$\|C\| = 6(P)$		$\|C\| = 10$		$\|C\| = 15$		$\|C\| = 22(T)$		$\|C\| = 25$	
	NVI	V	NVI	V	NVI	V	NVI	V	NVI	V
DP+NC	**0.85**	**0.53**	**0.87**	**0.54**	**0.94**	**0.53**	**0.92**	**0.55**	**0.98**	**0.52**
RR08	1.05	0.44	1.09	0.46	1.06	0.5	1.14	0.48	1.18	0.48

	Chinese, CTB10, IT measures									
	$\|C\| = 5$		$\|C\| = 9(P)$		$\|C\| = 15$		$\|C\| = 20$		$\|C\| = 24(T)$	
	NVI	V	NVI	V	NVI	V	NVI	V	NVI	V
DP+NC	**0.9**	**0.47**	**0.92**	**0.47**	**0.95**	**0.47**	**1**	**0.46**	**1**	**0.46**
RR08	0.96	0.44	1	0.44	1.18	0.41	1.21	0.42	1.26	0.42

	English, WSJ10, Mapping measures, (UF = 74.6)									
	$\|C\| = 5$		$\|C\| = 8(P)$		$\|C\| = 15$		$\|C\| = 20$		$\|C\| = 26(T)$	
	F(m:1)	F(1:1)	F(m:1)	F(1:1)	F(m:1)	F(1:1)	F(m:1)	F(1:1)	F(m:1)	F(1:1)
DP+NC	**59.7**	**50.8**	**61**	**48.9**	**60.6**	42.6	**61.8**	**42.3**	**61.4**	**38**
RR08	50	42.7	49.6	42.5	55.9	**42.8**	60.5	38.9	58.9	33.2

	German, NEGRA10, Mapping measures, (UF = 58.1)									
	$\|C\| = 6(P)$		$\|C\| = 10$		$\|C\| = 15$		$\|C\| = 22(T)$		$\|C\| = 25$	
	F(m:1)	F(1:1)	F(m:1)	F(1:1)	F(m:1)	F(1:1)	F(m:1)	F(1:1)	F(m:1)	F(1:1)
DP+NC	**44.9**	**44.6**	**44.7**	**43**	45	**41.1**	46.6	**41.1**	45.4	**40.6**
RR08	42.6	37.7	43.6	37	**48.1**	36.7	**48.1**	35.2	**48.2**	34.9

	Chinese, CTB10, Mapping measures, (UF = 51.8)									
	$\|C\| = 5$		$\|C\| = 9(P)$		$\|C\| = 15$		$\|C\| = 20$		$\|C\| = 24(T)$	
	F(m:1)	F(1:1)	F(m:1)	F(1:1)	F(m:1)	F(1:1)	F(m:1)	F(1:1)	F(m:1)	F(1:1)
DP+NC	**35**	**29.8**	**35.5**	**29.8**	**35.8**	**29.6**	35.9	**28**	36	**28**
RR08	33.6	28.5	34.7	27.1	35.2	23.1	**36.2**	21.5	**36.4**	22

Table 2: Comparison between our main clustering ensemble model and the model of Reichart and Rappoport (2008) (RR08). DP: Dirichlet process. NC: normalized cut. The top three tables are for information theoretic based measures. The bottom three tables are for mapping-based measures. The columns of each table represent the specified number of induced clusters. F(m:1) and F(1:1) are labeled F-score values computed after the induced categories were mapped to the gold categories with many-to-one and 1-to-1 mappings respectively. Higher V, F(m:1) and F(1:1) and lower NVI scores imply the induced clustering to be more similar to the gold standard. The clustering ensemble model (DP+NC) provides considerable improvement over RR08.

Our Expert Ensemble Approach. We start by comparing our DPMM with normalized-cute approach (DP+NC) to RR08. Table 2 shows that DP+NC clearly outperforms the RR08 model. For IT measures it is better in 28 out of 30 experimental conditions and for F(1:1) it is better in 14 of 15 conditions. For F(m:1), DP+NC is better for English while for German and Chinese it is better when the number of induced clusters is small.

RR08 showed that their algorithm is superior to an algorithm that performs random labeling or replaces their final BMM mapping with a random mapping. Since our algorithm is shown to be superior over theirs, it is also better than these random baselines.

Note that RR08 proposed a representation vector of $3|M| + |S|$ features where M is the number of BMM categories. Our DPMM algorithm with this feature representation is very slow due to the high values of $|M|$ (between 2299 and 5559 for our experimental corpora). Consequently, we used a different representation (Section 5). To make sure that our results are not due to this change of features, we also ran their algorithm using our feature set. The superiority of our approach relative to RR08 using this setting was essentially similar to that reported above.

DPMM Clustering. To justify our selection of a DPMM for clustering we compare our results to a variant where DPMM is replaced by K-means (KM) with a cosine similarity measure. KM is known to be very sensitive to its initialization. We ran it once where the cluster centers are randomly initialized and once where, like our DPMM models, cluster centers are initialized to be the k most frequent BMM categories. Since the former model has a substantially inferior performance, we only compare to the latter. We also compare to an ensemble of experts, where each expert is a run of the BMM mapping scheme of RR08 with a different number of target clusters. For both K-means and RR08, we averaged the resulting experts using the same normalized cut algorithm used by our method.

Due to the large number of experimental conditions (3 models and 60 setups: 3 corpora, 4 measures and 5 label set sizes), for clarity of exposition, we only provide a summary of the results. For V, NVI and F(1:1), DP+NC achieves the best score in 39 out of 45 cases. For F(m:1), DP+NC is the best performing model for WSJ10. For NEGRA10 and CTB10, RR08+NC and KM+NC often provide the best performance, but DP+NC is superior to RR08 when the number of induced clusters is small.

Normalized Cut Model Averaging. To justify our selection of the normalized cut algorithm for model averaging, we experimented with various variants of our algorithm (DP+NC) and of the baseline ensemble of experts algorithm where the experts are induced by the informed KM algorithm (KM+NC). In these variants only the model averaging component (NC) is changed. We experimented with several linkage clustering algorithms (complete, single and average) and distance functions (cosine, sample correlation between observations, sample Spearman's correlation between observations and Euclidean). In these experiments each BMM category j is represented by a vector in which the value of the ith coordinate is the number of ensemble members that cluster the ith and jth categories together. We report only the results of the complete link (CL) with cosine distance that provides the best results[11].

In 52 out of the 56 cases where a clustering ensemble model outperforms RR08 NC is the algorithm that is used by the best performing model. Over all cases, DP+NC produces better clustering than DP+CL in 53 of the 60 cases.

[11] In summary, we now discuss five ensemble models. The three that were defined above: DP+NC, KM+NC, and RR08+NC, and the DP+CL and KM+CL defined here.

	No. of Constituents	1	2	3	4	5	6
NP	7898	40.4	59.76	70.4	80.3	88	90.1
VP	6758	60.18	74.77	85.35	90.2	94.24	95.96
PP	3234	44.77	45.92	63.73	71.8	79.22	85.93
S	1076	25.84	43.22	58.74	70.07	81.13	89.22
SBAR	492	34.15	47.56	60.57	72.97	80.78	87.2
ADJP	331	47.13	67.98	74.02	79.76	85.2	90.7
ADVP	226	43.8	66.8	77.88	84.96	89.82	92.48
SQ	119	45.38	85.7	96.6	98.3	100	100

Table 3: Performance of the DP+NC model on the 8 most frequent WSJ10 syntactic categories. For each gold category (lines) we show the fraction of the constituents it annotates that are labeled in the induced annotation by the *k* induced categories that label the most of these constituents (column headed with '*k*').

Measure	V	NVI	F(m:1)	F(1:1)
English	0.034	0.12	0.013	0.117
German	0.048	0.049	0.017	0.040
Chiense	0.031	0.119	0.032	0.123

Table 4: The standard deviation to mean performance ratio of the DP+NC model when the number of induced clusters is varied. The effect of the change in the number of clusters on the F(1:1) and NVI measures is an order of magnitude larger, for English and Chinese.

The Mixture of Experts Approach. Finally, we want to demonstrate the importance of adopting an ensemble approach regardless of the specifics of the experts or the aggregation algorithm. Looking at a total of 60 experiments for corpora (3), evaluation measures (4) and number of induced clusters (5), we see a clear advantage of the ensemble approach: in 56 of the 60 cases the ensemble models substantially outperform the RR08 model, and are competitive in the other 4 experiments.

For F(m:1) the difference is up to 11.4% (WSJ10), up to 4.3% (NEGRA10) and up to 1.9% (CTB10). For V it is up to 12%, up to 9% and up to 6% for these corpora respectively. Results are even more impressive for F(1:1) and NVI. For F(1:1), improvement is up to 8.1% (WSJ10), 6.9% (NEGRA10) and 6.5% (CTB10). For NVI, the error reduction[12] is up to 18.3%, 20.2% and 20.6% respectively.

8 Qualitative Analysis

To get a better understanding of the quality of the syntactic categories induced by our model, we provide in this section a qualitative analysis of the performance of our model. We first provide a detailed error analysis of the performance of one of our models, the DP+NC model on the WSJ10 corpus when 15 categories are induced [13]. We then analyze the cross-lingual effect of an important aspect of our model – the number of induced clusters.

English Error Analysis The WSJ10 gold annotation obeys the Zipf law according to which most of the constituents (phrases) of the corpus are annotated with a small number of categories

[12] NVI values are not limited to [0,1], we thus report error reduction, computed as: $\frac{NVI_{model} - NVI_{baseline}}{NVI_{baseline}}$

[13] In order to better analyze the ability of our algorithm to detect the 'S' category, in the analysis of this section we do not count the sentence level constituent which is annotated with this category in 84.7% of the cases.

and the rest of the constituents are annotated by a larger number of much smaller categories. Concretely, while this gold annotation consists of 26 categories, 97.6% (91.8%) of the corpus constituents are labeled with the 8 (4) categories that annotate the highest number of constituents (referred to as 'the most frequent 8(4) categories').

A similar pattern is observed in the categories induced by the DP+NC model: the 8(4) most frequent categories annotate 88.5% (62.2%) of the constituents. The stronger magnitude of the Zipfian effect in the gold annotation suggests that biasing our model towards a stronger Zipfian pattern (e.g. by adding a normalization term to the NC optimization problem) may improve its performance.

Table 3 presents the distribution of each of the 8 most frequent gold categories between the 6 most frequent induced categories that annotate most of its constituents. The table shows that for 6 of these categories (all categories except from 'S' and 'SBAR') at least 40% of the constituents are annotated by the same induced category and 63.7%-96.6% of the constituents are annotated by 3 induced categories. The algorithm is shown to performs especially well in detecting the 'VP' and 'SQ' categories (60.18%-85.35% and 45.38%-96.6% of the constituents in 1-3 induced categories respectively). Performance on the 'SBAR' and 'S' categories are somewhat lower (at least in terms of overlapping with their 3 most overlapping categories).

Cross-Lingual Analysis Here we provide cross-lingual error analyse for one of the choices made by our model, the number of induced clusters. Table 4 shows the standard deviation to mean performance ratio for the DP+NC model in all three languages. While for German, all measures are relatively indifferent to the number of clusters, for Chinese and English the ratio for the F(1:1) and NVI measures is an order of magnitude larger than for F(m:1) and V.

This pattern leads us to two interesting observations that may guide future research in the field. First, a large number of clustering evaluation measures have been proposed in the literature (Reichart and Rappoport, 2009). Our experiments suggest that F(1:1) and NVI are more sensitive to a change in the number of induced clusters. Second, the performance of our model on German is mostly indifferent to the number of clusters, according to all measures. This calls for a deeper investigation of the properties of our algorithm especially with respect to languages that are typologically similar to German.

Conclusion and perspectives

We presented a novel clustering ensemble model for unsupervised induction of syntactic categories. Our model uses the Dirichlet process mixture model for expert induction and normalized-cut model averaging, providing a substantial improvement over previous works in English, German and Chinese.

Our contribution is two-fold. First, we bring the idea of ensemble learning into the task of unsupervised induction of syntactic categories, leading to substantial performance improvement. Second, and more importantly, we show how to construct a diverse ensemble of experts using the Dirichlet Process mixture model.

In future work we intend to experiment with more languages. The hierarchical generalization of the Dirichlet Process offers an opportunity for future joint learning of the syntactic structures and its annotation. On an orthogonal axis, the output of our algorithm can be used to train supervised parsers.

Acknowledgments

The work in this paper was funded by the EU grant 7FP-ITC-248064.

References

Antoniak, C. (1974). Mixture of dirichlet processes with applications to bayesian non–parametric problems. *The Annals of Statistics*, 2(6):1152–1174.

Berg-Kirkpatrick, T. and Klein, D. (2010). Phylogenetic grammar induction. In *Proc. of ACL*.

Blunsom, P. and Cohn, T. (2010). Unsupervised induction of tree substitution grammars for dependency parsing. In *Proc. of EMNLP*.

Bod, R. (2006). Unsupervised parsing with u-dop. In *Proc. of CoNLL-X*.

Borensztajn, G. and Zuidema, W. (2007). Bayesian model merging for unsupervised constituent labeling and grammar induction. In *Technical Report, ILLC*.

Brants, T. (1997). The negra export format. In *CLAUS Report, Saarland University*.

Clark, A. (2001). *Unsupervised Language Acquisition: Theory and Practice*. PhD thesis, University of Sussex.

Cohen, S. and Smith, N. (2009). Shared logistic normal distributions for soft parameter tying in unsupervised grammar induction. In *Proc. of NAACL*.

Daume-III, H. (2007). Fast search for dirichlet process mixture models. In *Proc. of AISTAT*.

Dennis, S. (2005). An exemplar-based approach to unsupervised parsing. In *Proc. of CogSci*.

Dietterich, T. (2000). Ensemble methods in machine learning. *Lecture Notes in Computer Science*, 1857:1–15.

Ferguson, T. (1973). A bayesian analysis of some non-parametric problems. *The Annals of Statistics*, 1(2):209–230.

Gildea, D. and Jurafsky, D. (2002). Automatic labeling of semantic roles. *Computational Linguistics*, 28:245–288.

Gillenwater, J., Ganchev, K., Jo~ao V. Grac¸a, n. T., and Preira, F. (2010). Sparsity in dependency grammar induction. In *Proc. of ACL*.

Goldberg, A. (2006). *Constructions at Work*. Oxford University Press.

Goldwater, S., Griffiths, T., and Johnson, M. (2006). Interpolating between types and tokens by estimating power-law generators. In *Proc. of NIPS*.

Haghighi, A. and Klein, D. (2006). Prototype-driven grammar induction. In *Proc. of ACL*.

Haghighi, A. and Klein, D. (2007). Unsupervised coreference resolution in a nonparametric bayesian model. In *Proc. of ACL*.

Headden, W., Johnson, M., and McClosky, D. (2009). Improving unsupervised dependency parsing with richer contexts and smoothing. In *Proc. of NAACL*.

Johnson, M. and Goldwater, S. (2009). Improving nonparametric bayesian inference: Experiments on unsupervised word segmentation with adaptor grammars. In *Proc. of NIPS*.

Johnson, M., Griffiths, T., and Goldwater, S. (2007). Adaptor grammars: a framework for specifying compositional nonparametric bayesian models. In *Proc. of NIPS*.

Klein, D. (2005). *The unsupervised learning of natural language structure*. PhD thesis, Stanford University.

Klein, D. and Manning, C. (2002). A generative constituent-context model for improved grammar induction. In *Proc. of ACL*.

Klein, D. and Manning, C. (2003). Accurate unlexicalized parsing. In *Proc. of ACL*.

Klein, D. and Manning, C. (2004). Corpus-based induction of syntactic structure: Models of dependency and constituency. In *Proc. of ACL*.

Kubler, S., McDonald, R., and Nivre, J. (2009). *Dependency Parsing – Synthesis Lectures on Human Language Technologies*. Morgan and Claypool.

Kuhn, H. (1955). The hungarian method for the assignment problem. *Naval Research Logistics Quarterly*, 28:245–288.

Meila, M. (2007). Comparing clustering – an information based distance. *Journal of Multivariate Analysis*, 98:873–895.

Miyao, Y., Satre, R., Sagae, K., Matsuzaki, T., and Tsujii, J. (2008). Task-oriented evaluation of syntactic parsers and their representations. In *Proc. of ACL*.

Munkres, J. (1957). Algorithms for the assignment and transportation problems. *Journal of the SIAM*, 5(1):32–38.

Nianwen, X., Chiou, F.-D., and Palmer, M. (2002). Building a large–scale annotated chinese corpus. In *Proc, of ACL*.

punyakanok, V., Roth, D., and tau Yih, W. (2008). The importance of syntactic parsing and inference in semantic role labeling. *Computational Linguistics*, 24(1):257–287.

Reichart, R. and Rappoport, A. (2008). Unsupervised induction of labeled parse trees by clustering with syntactic features. In *Proc. of COLING*.

Reichart, R. and Rappoport, A. (2009). The nvi clustering evaluation measure. In *Proc. of CoNLL*.

Rosenberg, A. and Hirschberg, J. (2007). Entropy-based external cluster evaluation measure. In *Proc. of EMNLP-CoNLL*.

Seginer, Y. (2007). Fast unsupervised incremental parsing. In *Proc. of ACL*.

Smith, N. and Eisner, J. (2006). Annealing structural bias in multilingual weighted grammar induction. In *Proc. of ACL*.

Spitkovsky, V., Alshawi, H., and Jurafsky, D. (2011a). Lateen em: Unsupervised training with multiple objectives, applied to dependency grammar induction. In *Proc. of EMNLP*.

Spitkovsky, V., Alshawi, H., Jurafsky, D., and Manning, C. D. (2010a). From baby steps to leapfrog: How less is more in unsupervised dependency parsing. In *Proc. of NAACL-HLT*.

Spitkovsky, V., Alshawi, H., Jurafsky, D., and Manning, C. D. (2010b). Viterbi training improves unsupervised dependency parsing. In *Proc. of CoNLL*.

Spitkovsky, V., Chang, A., and Jurafsky, D. (2011b). Unsupervised dependency parsing without gold part-of-speech tags. In *Proc. of EMNLP*.

Stolcke, A. (1994). *Bayesian Learning of Probabilistic Language Models*. PhD thesis, University of of California at Berkeley.

Stolcke, A. and Omohundro, S. M. (1994). Inducing probabilistic grammars by bayesian model merging. In *Grammatical Inference and Applications, Second International Colloquium*.

Yu, S. and Shi, J. (2003). Multiclass spectral clustering. In *Proc, of ICCV*.

Zhang, H. and Gildea, D. (2004). Syntax-based alignment: Supervised or unsupervised? In *Proc, of COLING*.

From Finite-State to Inversion Transductions: Toward Unsupervised Bilingual Grammar Induction

Markus SAERS *Karteek ADDANKI* *Dekai WU*
Human Language Technology Center
Hong Kong University of Science and Technology, Hong Kong
masaers@cs.ust.hk, vskaddanki@cs.ust.hk, dekai@cs.ust.hk

ABSTRACT
We report a wide range of comparative experiments establishing for the first time contrastive foundations for a completely unsupervised approach to bilingual grammar induction that is cognitively oriented toward early category formation and phrasal chunking in the bootstrapping process up the expressiveness hierarchy from finite-state to linear to inversion transduction grammars. We show a consistent improvement in terms of cross-entropy throughout the bootstrapping process, as well as promising decoding experiments using the learned grammars. Rather than relying on external resources such as parses, POS tags or dictionaries, our method is fully unsupervised (in the way this term is typically understood in the machine translation community). This means that the bootstrapping can only rely on information gathered during the previous step, which necessitates some strategy for expanding the expressiveness of the grammars. We present principled approaches for moving from finite-state to linear transduction grammars as well as from linear to inversion transduction grammars. It is our belief that early, integrated category formation and phrasal chunking in this unsupervised bootstrapping process is better aligned to child language acquisition. Finally, we also report exploratory decoding results using some of the learned grammars. This is the first step towards an end-to-end grammar-based statistical machine translation system.

KEYWORDS: Grammar & Formalisms, Empirical machine translation, Multilinguality and Bilingual grammar induction.

Proceedings of COLING 2012: Technical Papers, pages 2325–2340,
COLING 2012, Mumbai, December 2012.

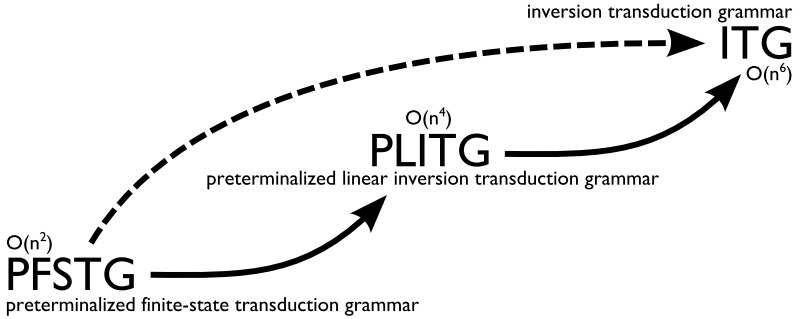


Figure1: Bootstrapping paths through the grammar hierarchy along with biparsing time complexities of grammar formalisms.

1 Introduction

We report a wide range of comparative experiments establishing for the first time contrastive foundations for a completely unsupervised approach to bilingual grammar induction that is cognitively oriented toward early category formation and phrasal chunking in the bootstrapping process up the expressiveness hierarchy from finite-state to linear to inversion transduction grammars.

In the context of bilingual grammar induction, "unsupervised" means that a transduction grammar (or synchronous grammar) is learned without using any external resources such as parses, POS tags, or dictionaries. In contrast, many, if not most, tree-based statistical MT approaches rely on monolingually parsed, chunked, and/or tagged parallel corpora (e.g., Galley et al. (2006)), from which a transduction grammar is extracted. Such approaches must compensate for monolingual analyses that often are not designed optimally for expressing the relationship between, say, English and Chinese. Exceptions include the hierarchical phrase-based SMT method of learning ITGs (Chiang, 2005), which does not rely on external resources.

However, unlike Chiang (2005) where huge numbers of (linguistically questionable) phrase translations are essentially memorized, our present work aims at inducing syntactic categories at an early stage in the learning, as occurs in child language acquisition. While only time will tell whether our more cognitively motivated approach to the induction of structural and lexical relationships between two languages will lead to better machine translation models, it is our belief that ultimately, learning the correct categories as early as humans do will provide a more accurate generalization bias for subsequent learning of more complex transduction rules.

In this paper, we report a large body of foundational experiments that, for the first time, illuminate how EM-based bilingual grammar induction behaves as we move up the complexity hierarchy of transduction grammars. We investigate various bootstrapping approaches to induction of finite-state transduction grammars (FSTGs), linear transduction grammars (LTGs), and inversion transduction grammars (ITGs). We then turn to various combinations of phrasal chunking to induce segmental transduction grammars, which can represent various classes of phrase-based translations. We then integrate category induction methods in various combinations.

This research thread represents a long term effort to investigate how a more principled, generalization-oriented model of bilingual grammar induction could gradually replace various parts of the long chain of heuristics used in the more memorization-oriented approaches. Although we do also look at preliminary BLEU and NIST scores from MT decoding using the induced transduction grammars, we believe it is premature to place excessive emphasis on such scores because too many other factors (such as the language model, to name one) are entangled. At this early stage of establishing the foundations for understanding category formation in bottom-up bilingual grammar induction, it is more important to compare more *directly* how well the relationships between the two languages are learned by the very many possible various combinations of bootstrapping between transduction expressivity levels, phrasal chunking, and category formation. Therefore, we focus more on the cross entropy of the induced bilingual grammars—which is exactly the bilingual form of the standard way to evaluate how well monolingual grammar induction captures monolingual data.

Note that, as observed in Wu (1997), this way of analyzing bilingual grammar induction can also be viewed as the problem of *bilingual language modeling*—modeling two languages simultaneously. We attack this problem of bilingual language modeling by extending a surprisingly effective finite-state baseline in two different ways: adding reordering capabilities and adding context information in the form of categories. The ultimate aim of this research direction is to have a grammar-based end-to-end statistical machine translation system, which would enable the usage of the same model in training and decoding instead of relying on a pipeline of different mismatched models trained independently. The bilingual language models we learn in this paper are grammar-based, and they thus represent a step towards this goal. Our new model is still extremely simple, even with the two proposed extensions. Reordering is added by first promoting the finite-state model to a linear model, which allows reading from the beginning or end of the two sentences independently and then by promoting the linear model to a fully nesting model. Contextual information is added in the form of a handful of categories, which are learned exclusively from the raw data.

We start by allowing any input token to translate into any output token, we also allow the input or output token to be empty. The lexical rules can contain input and output strings of length zero or one, but at least one of them must be nonempty. This initial search space is reigned in by training, which eliminates some of the (very unlikely) rules, but it is mainly expanded by allowing observed sequences to be taken up as lexicalized sequences (chunking), and by allowing category diversification (splitting). This initial stage of search space expansion is carried out in a very weak grammar formalism, which forces it to focus on learning lexical rules, since the weak structure of the grammar makes it rely heavily on surface form. Once the lexical chunks have been found and tentatively categorized, we will move the grammar into a more expressive formalism, and further train it. The idea is that we will have found enough good rules in the expansion phase that training with the more expressive formalism is mainly about establishing the structure and getting rid of useless rules. This approach also has the benefit that a lot of the "heavy lifting" can be done in the weaker grammar formalism, which is also less expensive to compute.

We will consistently use *preterminalized finite-state transduction grammars* (PFSTGs) as the weak grammar formalism, and *preterminalized linear inversion transduction grammars* (PLITGs) and *inversion transduction grammars* (ITGs) as the more expressive grammar formalisms.

The PFSTG's weakness comes from the fact that they have no structural way to reorder the input into the output. Instead they have to rely on singletons – lexical rules that delete from the input or insert into the output. These rules allow the PFSTG to disregard parts of a sentence pair, but still account for the parts that it can account for. It also means that PFSTGs need to keep a lot of singleton rules around that a more expressive grammar could do without.

ITGs are capable of considerable structural reordering, they are in fact the most expressive transduction grammar that can be used to parse a parallel corpus in polynomial time. They are, in other words, the most expressive grammar that can possibly be afforded.

PLITGs are wedged between PFSTGs and ITGs in terms of expressiveness. They do allow for some structured reordering. Like PFSTGs, the parse trees are chains rather than trees, but unlike PFSTGs, the links in these chains do not have to be physically adjacent. Allowing the first input token to be lexically associated with the last output token is an example of the kind of reordering that PLITGs allow, which make them more expressive than PFSTGs. The fact that these reorderings are limited to lexical units only is what makes them less expressive than ITGs (Saers et al., 2011) .

The rest of the paper is structured so that we begin by describing the initial finite-state transduction grammar that we will start our bootstrapping sequence (Section 2). We then describe how the lexical chunking (Section 3) and category splitting (Section 4) is carried out before moving on to expressivity expansion: first from finite-state to linear transduction grammars (Section 5), and then from linear to inversion transduction grammar (Section 6). We then move on to an illustrative example of what the bootstrapping process actually entails (Section 7), along with some tentative decoding results, before offering some concluding remarks (Section 8).

2 Initial grammar

As a grammar-based translation model baseline, we will use the simplest possible transduction grammar: the finite-state transduction grammar. A finite-state transduction grammar is the grammar form of a finite-state transducer. The transformation into a grammar is trivial and gives us FSTGs.

2.1 Definition 1

A FSTG over languages L_0 and L_1 is a tuple $G = \langle N, \Sigma, \Delta, S, R \rangle$, where N is a finite nonempty set of nonterminal symbols, Σ is a finite nonempty set of L_0 tokens, Δ is a finite nonempty set of L_1 tokens, $S \in N$ is the designated start symbol and R is a finite nonempty set of finite-state transduction rules on the forms:

$$A \rightarrow e/f\,B, \quad A \rightarrow \epsilon/\epsilon$$

where $A, B \in N$ *and* $e/f \in (\Sigma^* \times \Delta^*) - \{\epsilon/\epsilon\}$.

To harmonize this kind of grammar with future extensions to it, we will preterminalized it. A *preterminalized* FSTG has a special class of nonterminals called *preterminals*, which have a monopoly on rewriting to biterminals. This does not change the expressivity of the grammar, nor its asymptotic time complexity. To see why, simply imagine the degenerate case where every biterminal has exactly one unique preterminal symbol associated with it. In this paper, the only FSTGs we will use are *preterminalized finite-state transduction grammars*, or PFSTGs.

2.2 Definition 2

A PFSTG over languages L_0 and L_1 is a tuple $G = \langle N, P, \Sigma, \Delta, S, R \rangle$, where N is a finite nonempty set of nonterminal symbols, P is a finite nonempty set of preterminal symbols, disjoint from N, Σ is a finite nonempty set of L_0 tokens, Δ is a finite nonempty set of L_1 tokens, $S \in N$ is the designated start symbol and R is a finite nonempty set of preterminalized finite-state transduction rules on the forms:

$$A \to QB, \quad A \to \epsilon/\epsilon, \quad Q \to e/f$$

where $A, B \in N$, $Q \in P$ and $e/f \in (\Sigma^* \times \Delta^*) - \{\epsilon/\epsilon\}$.

As a baseline, we will choose the simplest possible PFSTG: a *bracketing* PFSTG. The bracketing PFSTG has only one nonterminal symbol and one preterminal symbol. As such it relies solely on lexical information to find translation correspondences. The lexical information is surprisingly powerful when both sentences are given.

To assign scores to the sentence pairs, we need a weighting function for the rules, making it a *weighted* PFSTG. We will go one step further, and require the weighting function to define proper probability distributions such that:

$$p(\psi \to \varphi) \equiv Pr(\varphi \mid \psi)$$

which makes the grammar *stochastic*. These kind of conditional probabilities over rules can be tuned towards a corpus of examples (a training corpus) through expectation maximization, or EM. Expectation maximization can only be used to tune existing parameters, so to have something to tune, we initialize the grammar using the training corpus. The structural rules will have uniform probability, and the lexical rules will have a portion of the probability mass relative to their cooccurrence in the training corpus. Specifically, we will initialize a stochastic bracketing FSTG such that:

$$\begin{aligned} p(S \to A) &= 1, \\ p(A \to QA) &= 0.5, \\ p(A \to \epsilon/\epsilon) &= 0.5, \\ p\left(Q \to e/f\right) &= \frac{c\left(e/f\right)}{\sum_{e' \in \Sigma, f' \in \Delta} c\left(e'/f'\right)} \end{aligned}$$

where $c\left(e/f\right)$ is the cooccurrence count for the biterminal $e/f \in ((\Sigma \cup \{\epsilon\}) \times (\Delta \cup \{\epsilon\})) - \{\epsilon/\epsilon\}$.

To calculate the expectations for the expectation maximization, we will parse the training corpus with the grammar we have, using an adaptation of the parsing algorithm described in Saers et al. (2010). The adaptation consists of allowing multiple categories (not simply bracketing grammars), and to handle preterminals. Although this algorithm was designed for the linear family of transduction grammars, the PFSTGs are merely a restricted form of PLITGs, which means that the algorithm applies to them as well.

Training this grammar on IWSLT07 ChineseEnglish data (Fordyce, 2007) gave us a sentence-level cross-entropy of 110.2. The results of all training runs can be found in Table 1, where the baseline model is designated `fstg`.

3 Chunking helps

We can induce new lexical rules by allowing lexical entities to combine in order to form larger lexical entities. The end results are similar to phrase-based machine translation, but the method of arriving at the segments (chunks or "phrases") is very different.

To apply chunking to our PFSTG, we use the method described in Saers and Wu (2011). The method was originally developed for PLITGs, which is a superset of PFSTGs, so it can be applied as is. The gist of the method is to allow two bitokens that are observed next to each other to combine into a new, larger bitoken. The process is thus limited to produce bitokens of twice the length of existing bitokens. There is, however, nothing stopping us from applying the method several times, getting larger and larger chunks. The maximum bitoken length of the baseline grammar is 2 (one input token and one output token), which we can double by chunking.

By applying chunking to the baseline PFSTG, and training with the chunks (model `fstg_c` in Table 1), we get a cross-entropy of 73.0, which is significantly better than the 110.2 that the baseline grammar (model `fstg`) scored. If we repeat the chunking (model `fstg_c_c`), we go all the way down to 45.0. A third round of chunking and training (model `fstg_c_c_c`) gets us a small improvement down to 43.9. This is interesting, since going from segments of length 1 to segments of length 2 helps tremendously, and proceeding to segments of length 4 goes a long way indeed. Pushing it up to segments of length 8 (which is close to the default "maximum phrase length" of Moses: 7), does surprisingly little. Either way, chunking was responsible for cutting the cross-entropy in more than half.

4 Splitting helps

We can introduce new nonterminal or preterminal symbols by splitting off some of the probability mass of an existing symbol to one or more new symbols. This gives us the possibility of diversifying the categories used by the grammar, which is necessary in order to move away from bracketing grammars to more interesting types of grammars.

Splitting into new symbols is a problem that can be formulated as splitting the probability mass of one symbol to several existing symbols. This requires the new symbols to be inserted into the grammar before the splitting takes place, which is a reasonable prerequisite. The assumption that the probability mass of the symbol being split could end up in any of the existing symbols makes it necessary to have some mechanism to control the destination of the probability mass. Furthermore, it is desirable to have some mechanism to introduce perturbations into the splitting of the probability mass, since exactly splitting it uniformly will inhibit learning. In addition to this wish list, we also wanted to differentiate between the case when the nonterminal was on the left-hand side of a rule, and when it was on the right-hand side. Remember that we are dealing with preterminalized grammars, and being able to have different splitting policies depending on where in the rule the symbol occurs is important. We want preterminals to be heavily perturbed on the left-hand side, but lightly perturbed on the right-hand side. When the preterminal is on the left-hand side, it is assigning a category to the terminal pair it rewrites to; by having larger differences in probability mass, we are essentially assigning a category at random, but keeping some of the mass in the other category, just in case the decision was wrong. When the preterminal is on the right-hand side, it determines the contexts in which a particular category can occur, which we do not want to randomize too much; relying on training is better. It turns out that all these desiderata can be addressed in a

very simple way.

To split a nonterminal x, we give a set of target nonterminals $y_0, y_1, ..., y_k$ paired up with a kind of pseudo counts $a_0, a_1, ..., a_k$ that represent "how often x was split into y_i". The as are used as parameters to a Dirichlet distribution, from which a categorical distribution is randomly drawn every time x has to be split.

If any of the ys do not exist in the grammar, they are added to it. If no pseudo count is given for a nonterminal, it is assumed to be zero. This means that the Dirichlet distribution will be well-defined over the set of nonterminal symbols in the grammar. The expected categorical distribution drawn from the Dirichlet distribution will be "relative frequency" over the pseudo counts, but since we are drawing the categorical distribution at random, there will be some perturbation to it. The magnitude of the perturbation will be inversely proportional to the total number of pseudo counts in the Dirichlet distribution. Intuitively, the more pseudo counts we have, the more certain we can be that they are correct. Conversely, with few pseudo counts, there is a lot of uncertainty, and therefore a lot of variation in the categorical distributions we draw.

Once we have a categorical distribution over the set of nonterminals, we can distribute the probability mass of a rule containing x, into the expected portion according to the categorical distribution. To keep x as a symbol, it needs to retain some of the probability mass, and the simplest way to do this is to have non-zero pseudo counts for it; this is the approach we took.

To allow for different behavior depending on where in the rule x occurs, we simply supply two Dirichlet distributions: one to be used when x is on the left-hand side of a rule, and one to be used when it is on the right-hand side.

The pseudo code for the algorithm is give below.

4.1 General nonterminal (and preterminal) splitting algorithm

To refresh memory, a rule consists of a single nonterminal (or preterminal) symbol (i) on the left-hand side, and a sequence of nonterminal (or preterminal) and biterminal symbols on the right-hand side (ϕ). Furthermore, there is a permutation over the right-hand side (π). The rule form is thus $i \rightarrow \phi ; \pi$. There is also a probability function that assigns a probability to each rule (p).

Input: a set of rules R, a nonterminal to split x, a probability function p over R, a left-hand side Dirichlet distribution over nonterminals D_l, a right-hand side Dirichlet distribution over nonterminals D_r.

Output: a new set of rules R' and a new probability function p' over R'.

$R' \leftarrow ()$
for all $i \rightarrow \phi ; \pi \in R$ **do**
  $R'' \leftarrow ()$
  if $i = x$ **then**
    Draw the parameters to a categorical distribution over nonterminals $\alpha \sim D_l$
    for all nonterminal y **do**
      $R'' \leftarrow (R'', [y \rightarrow ; \pi, \alpha_y p(i \rightarrow \phi ; \pi)])$
    end for

else
$R'' \leftarrow ([i \rightarrow; \pi, p(i \rightarrow \phi; \pi)])$
end if
for all $0 \leq k < \left|\phi\right|$ **do**
$R''' \leftarrow ()$
for all $[z \rightarrow \psi_{0..k}; \pi, p''] \in R''$ **do**
if $\phi_k = x$ **then**
Draw the parameters to a categorical distribution over nonterminals $\alpha \sim D_r$
for all nonterminals y **do**
$R''' \leftarrow (R''', [z \rightarrow \psi_{0..k} y; \pi, \alpha_y p''])$
end for
else
$R''' \leftarrow (R''', [z \rightarrow \psi_{0..k} \phi_k; \pi, p''])$
end if
$R'' \leftarrow (R'', R''')$
end for
end for
for all $[r, p''] \in R''$ **do**
$R' \leftarrow (R', r)$
$p'(r) += p''$
end for
end for

We consistently used pseudo counts of 1000 (both for left- and right-hand side) when splitting nonterminals, and 0.5 (for left-hand side) and 1000 (for right-hand side) when splitting preterminals.

So far, we only have four PFSTGs, with different segment lengths. We did not carry out all possible experiments, but the ones we did carry out all show that splitting helps (all models in Table 1 with an `s` followed by some number have had all their symbols split into the given number of symbols). Sometimes it helps as little as when moving from 73.0 for a bracketing PFSTG with segment length 2, to 72.5 for the same grammar with its pre- and nonterminal split in two (model `fstg_c` to `fstg_c_s2`). The same modest improvement was observed when splitting the grammar with segment length 4, where we moved from 45.0 cross-entropy to 44.5 (model `fstg_c_c` to `fstg_c_c_s2`). However, splitting again gave 42.9 (moving to `fstg_c_c_s2_s2`), and a third time gave 39.5 (moving to `fstg_c_c_s2_s2_s2`). Beyond the first split, we see a better improvement than chunking an extra time.

5 Moving to linear transductions helps

The *preterminalized finite-state transduction grammar* is very closely related to the *preterminalized linear inversion transduction grammar*. The only difference is that the latter has more structural rules. As a reminder, the PFSTG has rules on the following forms:

$$A \rightarrow QB, \quad A \rightarrow \epsilon/\epsilon, \quad Q \rightarrow e/f$$

Whereas the PLITG has rule on the following forms:

$$A \rightarrow [QB], \quad A \rightarrow [BQ], \quad A \rightarrow \langle QB \rangle, \quad A \rightarrow \langle BQ \rangle, \quad A \rightarrow \epsilon/\epsilon, \quad Q \rightarrow e/f$$

Where the square and angled brackets have their customary interpretation as straight and inverted permutations respectively. The difference is in the structural rules, and each of the structural rules in a PFSTG corresponds to four structural rules in a PLITG. To move from a PFSTG to a PLITG, we merely have to add these rules, and distribute the probability mass. We distribute the probability mass uniformly.

Another possibility that PLITGs have that PFSTGs do not is to canonize the singletons. In the PLITGs we have used so far, there is always some ambiguity due to the fact that singletons are empty in one of the languages, and there is no way to tell whether the empty string should attach to the front or the back of the string in that language. This ambiguity can be eliminated by insisting that it attach to the same end that the known string attaches to in its language. This enforces a canonical form for singletons, and reduces complexity.

To canonize a PLITG, start by creating two new preterminal symbols, one for input singletons and one for output singletons for every existing preterminal. Then move all the singleton realizations of that preterminal to the newly created symbols. Consider the preterminal P. We will split it into $P^{e/f}$ (non-singletons), P^{f} (input singletons) and P^{e} (output singletons), and divide the set of rules so that the singletons must be produced by the singleton symbols, and the non-singletons must be produced by the non-singleton symbol.

To produce the new preterminal symbols, we also need to account for the case when P is on the right-hand side of a structural rule, and withholding some of the possible new rules will make the grammar singleton canonical. In a PLITG in normal form there may be at most one preterminal symbol on the right-hand side. Consider the four kinds of rules where P could occur in a PLITG:

$$A \rightarrow [PB], A \rightarrow [BP], A \rightarrow \langle PB \rangle, A \rightarrow \langle BP \rangle$$

To keep the PLITG canonical, we would like to have the following rules:

$$A \rightarrow \left[P^{e/f} B \right], A \rightarrow \left[B P^{e/f} \right], A \rightarrow \langle P^{e/f} B \rangle, A \rightarrow \langle B P^{e/f} \rangle,$$
$$A \rightarrow [P^{e} B], A \rightarrow \langle P^{e} B \rangle, A \rightarrow \left[P^{f} B \right], A \rightarrow \langle B P^{f} \rangle$$

The probability mass is divided to reflect the probability of P rewriting to non-singletons and the singletons respectively.

Whether we use a canonical PLITG or not, it allows us to move from any of the grammar we have trained so far to a PLITG and resume training with this more expressive grammar formalism. This gives a consistent improvement in cross-entropy (this can be seen in Table 1, where the PLITG models end in `_ltg`, and are meaningful to compare to the models preceding them).

6 Moving to inversion transductions helps

Moving from a PFSTG or PLITG into an ITG is a much more complicated process than moving from a PFSTG to a PLITG. Since we have that first step covered, we will focus on moving from a PLITG to an ITG. Once this step is in place, we can move from a PFSTG to an ITG *via* a PLITG (with or without training at the PLITG stage).

Model	cross-entropy
fstg	110.2
fstg_ltg	108.9
fstg_ltg_itg	95.5
fstg_itg	93.2
fstg_c	73.0
fstg_c_ltg	72.5
fstg_c_ltg_itg	60.7
fstg_c_itg	60.7
fstg_c_s2	72.5
fstg_c_s2_ltg	70.5
fstg_c_s2_itg	60.8
fstg_c_c	45.0
fstg_c_c_ltg	44.5
fstg_c_c_itg	36.5
fstg_c_c_s2	44.5
fstg_c_c_s2_ltg	42.5
fstg_c_c_s2_itg	35.8
fstg_c_c_s2_s2	42.9
fstg_c_c_s2_s2_ltg	39.5
fstg_c_c_s2_s2_itg	33.3
fstg_c_c_s2_s2_s2	39.5
fstg_c_c_s2_s2_s2_ltg	39.5
fstg_c_c_s2_s2_s2_itg	**32.8**
fstg_c_c_s3	44.5
fstg_c_c_s3_itg	36.3
fstg_c_c_c	43.9
fstg_c_c_c_itg	35.9

Table1: The cross-entropy scores on the data for various models.

6.1 Definition 3

A syntax-directed transduction grammar (SDTG) in normal form is a tuple $\langle N, \Sigma, \Delta, S, R\rangle$ where N is a finite nonempty set of nonterminal symbols, Σ is a finite nonempty set of input language symbols, Δ is a finite nonempty set of output language symbols, $S \in N$ is the designated start symbol, and R is a finite nonempty set of syntax-directed transduction rules on the forms:

$$S \to A, \quad A \to \varphi; \pi, \quad A \to e/f$$

where $A \in N$, $\varphi \in NNN^*$, π is a permutation vector over φ, and $e/f \in (\Sigma^* \times \Delta^*) - (\epsilon/\epsilon)$.

6.2 Definition 4

An inversion transduction grammar (ITG) in normal form is an SDTG in normal form, where the number of nonterminals allowed on the right-hand side is exactly two for all nonterminals

except the start symbol (which may only have one). The rules are thus on the forms:

$$S \rightarrow A, \quad A \rightarrow BC; \pi, \quad A \rightarrow e/f$$

Since there are only two possible permutation vectors (straight and inverted), the rules are typically expressed as:

$$S \rightarrow A, \quad A \rightarrow [BC], \quad A \rightarrow \langle BC \rangle, \quad A \rightarrow e/f$$

where the square brackets represent straight order, and the angled brackets represent inverted order.

To make a PLITG in normal form into an ITG in normal form, all that has to be done is to eliminate the rules where a nonterminal is allowed to go to nothing (empty rules). This does, however, leave us with an ITG that generates the exact same transduction that the PLITG generated, which is not necessarily what we want. It is more likely that we want a grammar that is more expressive than the grammar we had. To this end, we will promote the preterminals to full nonterminals. Since the removal of the empty rules entails having the nonterminals behave as preterminals, and promoting the preterminals to full nonterminals entails having the preterminals behave as nonterminals, there is no longer any point in making the distinction between the two classes of symbols. Indeed, an ITG only has nonterminal symbols.

The theory of promoting preterminals to nonterminals is to insert unary rules, where the preterminal rewrites into exactly one nonterminal symbol. For every preterminal that is to be promoted, one such rule is generated for each nonterminal. To control how much of its "preterminal-ness" the preterminal retain, we employ a hyperparameter α. The probability mass of the preterminal is redistributed so that α of it is retained in its original rules, whereas $1 - \alpha$ is redistributed to the new unary rules. The last thing that needs to be determined is the preterminals affinity to specific nonterminals. We use the function $\beta\left(y\right)$ for this, which gives the probability of the nonterminal y given the preterminal that is being promoted. In this round of experiments, β was uniform over all nonterminals. Naturally, we want to keep the new ITG in normal form, so we will actually not insert the unary rules, but rather anything the nonterminal can expand into. This gives us the following new probability function (p') over the new and old rules:

$$p'(x \rightarrow \phi; \pi) = \alpha p(x \rightarrow \phi; \pi) + (1 - \alpha) \sum_{y \in N} \beta\left(y\right) p(y \rightarrow \phi; \pi)$$

Where x is the preterminal being promoted, N is the set of nonterminals, ϕ is a sequence of nonterminals and biterminals, and π is a permutation over ϕ.

To train at the ITG stage, we use the algorithm presented in Saers et al. (2009), which we generalize to handle multiple nonterminals rather than being restricted to bracketing ITGs.

It is clear from Table 1 that the ITG models explain the data better than any other kind of model, and that more induction steps are better (the best model is the most heavily processed one, with two chunking steps and three splitting steps before moving on to ITGs). In the cases where we move to ITGs via PLITGs, we also see some improvements, which is encouraging.

7 Qualitative analysis and translation assessments

In this section, we present a qualitative analysis of an example to illustrate the nature of alignments learned during various stages in the training. As a step forward towards our goal of purely inducing ITGs in an unsupervised manner for the purpose of translation, we report our initial findings on using these bootstrapped models in translation tasks.

7.1 Bootstrapping improves the alignments

Although the cross-entropy scores indicate an improvement in the quality of the grammars learned after LTGs and ITGs with FSTGs, we were interested in understanding the qualitative nature of generalizations the model was learning. Such an analysis would reveal how new generalizations might emerge as expressivity of models increases. They would also help identify overfitting of the model and how errors propagate from one stage to another.

An important consideration in bootstrapping would be to identify whether or not alignments that could not be learned in less expressive models could be learned by more expressive models after bootstrapping. In order words, whether bootstrapping enables learning of those alignments that would otherwise be impossible to learn with less expressive models. One should also consider the possibility that incorrect alignments in the initial stages might prevent the correct alignments from ever being learned.

With the above question in mind, we decided to investigate one of the simplest and most common manifestations of this scenario in Chinese-English parallel data. Typically, locatives in chinese appear before the verb whereas in English they appear after the verb. Hence, the alignments of bisentences which contain these locative markers require the model to permit a certain degree of reordering. We present below, a qualitative analysis of the translation rules learned for one such locative marker at various stages in our bootstrapping pipeline.

We choose the Chinese locative marker which occurs 67 times in our data set. typically translates to *inside* in English. The following are two sentences that occur in our training data.

Source:
Gloss: inside is what ?
Translation: what is inside ?

Source:
Gloss: inside is what thing ?
Translation: what does it contain ?

In the first example, the correct alignment would require an alignment permutation of $[2, 1, 0, 3]$ which is beyond the expressive power of the FSTG models. As the translation that is present in the training data for the second example is inexact, the locative marker is forced to align to . These examples are representative of the way in which appears in the training corpus.

Table 2 shows some of the best translations (in the decreasing order of probability) of the token learned after the FSTG, FSTG_LTG, FSTG_ITG and FSTG_LTG_ITG stages in our training pipeline, in contrast to directly inducing ITGs. It is not surprising that FSTGs fail to learn the correct translations of the locative as most alignments of sentences containing the

FSTG		FSTG_LTG		FSTG_LTG_ITG		ITG	
$P \rightarrow$	/	$P \rightarrow$	/inside	$P \rightarrow$	/inside	$A \rightarrow$	/inside
$P \rightarrow$	/there	$P \rightarrow$	/it	$P \rightarrow$	/	$A \rightarrow$	/in
$P \rightarrow$	/it	$P \rightarrow$	/there	$P \rightarrow$	/it	$A \rightarrow$	/
$P \rightarrow$	/what	$P \rightarrow$	/	$P \rightarrow$	/in	$A \rightarrow$	/it
$P \rightarrow$	/I	$P \rightarrow$	/what	$P \rightarrow$	/I	$A \rightarrow$	/I

Table2: The best translation rules for the Chinese locative marker at various stages in the training pipeline.

Model	Cross-Entropy	BLEU	NIST
FSTG	110.2	7.95	0.4752
FSTG_LTG	108.9	8.34	0.7466
FSTG_LTG_ITG	95.5	8.83	0.8554

Table3: The correlation between the cross-entropy and the BLEU and NIST scores for token based models.

token are non-monotonic. From the second column of the table we can observe that the correct translation is learned to be the best translation after bootstrapping with LTGs alone. However, singleton translation and other synonyms of do not appear in the top five translations. Upon bootstrapping ITG with the FSTG_LTG, we identify that the translation *in* makes it into the top translations. The translations learned at FSTG_LTG_ITG stage are almost comparable to directly inducing an ITG as shown in the rightmost column.

Although locatives represent an extreme case of the nature of generalizations that could be learned from bootstrapping, we find that this kind of learning applies to all alignments at the FSTG stage. The correct alignments tend to get rewarded through successive stages of bootstrapping while the noisy alignments are drowned out. The end result is comparable to that of inducing LTGs and ITGs directly without the overhead of such an induction.

7.2 Successive bootstrapping promises improvements in translation quality

The final goal of our approach is to build a theoretically principled SMT system with well-defined representations. Although cross-entropy is a good measure of gauging how our models explain the data, it is not sufficient to guarantee an SMT system with a good performance. It is important to observe an improvement in the performance on a translation task that correlates with the improvement in cross-entropy of the model.

We approach this promise with caution as our models in their current state, are not optimized to compete with the state of the art SMT systems. We report the results of our preliminary experiments on using our trained models for the task of SMT.

Table 3 shows the BLEU scores obtained using only token based models (without any sort of chunking) on the IWSLT07_CE test set. These scores gives a gist of how translation quality changes with the grammar formalism, but a token-based model will naturally perform poorly compared to the state of the art. We used an in-house decoder for the purpose of decoding using our models. We used a trigram LM trained using SRILM (Stolcke, 2002) on the IWSLT dataset and a part of the gigaword data set.

We can observe significant gains in the BLEU (Papineni et al., 2002) and NIST (Doddington, 2002) scores after training on LTGs and a further improvement after training on ITGs. Further, these scores seem to be reflecting our estimates about the goodness of our models using cross-entropy.

Due to time constraints, we could not perform extensive experiments on the quality of translations produced with other models discussed in the rest of the paper. We also want to emphasize that an exhaustive evaluation of the performance of these models on translation tasks would result in a combinatoric explosion of models where chunking and splitting could be applied at various stages in the training pipeline with different chunking and splitting strategies. Therefore, in order to realize a competitive SMT system using our bootstrapping technique, we would first need to understand the qualitative and quantitative effects of using different models and the respective chunking and splitting strategies. We intend to pursue this task more confidently given our encouraging results on cross-entropy and the BLEU score correlation.

8 Conclusions

We reported a wide range of comparative experiments for a completely unsupervised approach to bilingual grammar induction with early category formation and phrasal chunking in the bootstrapping process up the expressiveness hierarchy from finite-state to linear to inversion transduction grammars. We reported a consistent improvement in the cross-entropy as we move through FSTGs to LTGs and ITGs. We also saw a substantial improvement in cross-entropy scores upon chunking and inducing categories. We reported an illustrative example that indicates generalizations learned from bootstrapping are equivalent to inducing models with higher expressivity directly but at a much lower cost. Finally, we discussed some of the encouraging results of our translation assessment experiments using bootstrapped models.

Acknowledgements

This material is based upon work supported in part by the Defense Advanced Research Projects Agency (DARPA) under BOLT contract no. HR0011-12-C-0016, and GALE contract nos. HR0011-06-C-0022 and HR0011-06-C-0023; by the European Union under the FP7 grant agreement no. 287658; and by the Hong Kong Research Grants Council (RGC) research grants GRF621008, GRF612806, DAG03/04.EG09, RGC6256/00E, and RGC6083/99E. Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the RGC, EU, or DARPA.

References

Chiang, D. (2005). A hierarchical phrase-based model for statistical machine translation. In *Proceedings of the 43rd Annual Meeting of the ACL*, pages 263–270, Ann Arbor, Michigan.

Doddington, G. (2002). Automatic Evaluation of Machine Translation Quality using N-gram Co-occurrence Statistics. In *Proceedings of the 2nd International Conference on Human Language Technology Research (HLT-02)*, pages 138–145, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Fordyce, C. S. (2007). Overview of the IWSLT 2007 evaluation campaign. In *Proceedings of the International Workshop on Spoken Language Translation*, pages 1–12.

Galley, M., Graehl, J., Knight, K., Marcu, D., DeNeefe, S., Wang, W., and Thayer, I. (2006). Scalable inference and training of context-rich syntactic translation models. In *Proceedings of*

the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics, pages 961–968, Sydney, Australia.

Papineni, K., Roukos, S., Ward, T., and Zhu, W.-J. (2002). BLEU: A Method for Automatic Evaluation of Machine Translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL-02)*, pages 311–318.

Saers, M., Nivre, J., and Wu, D. (2009). Learning stochastic bracketing inversion transduction grammars with a cubic time biparsing algorithm. In *Proceedings of the 11th International Conference on Parsing Technologies*, IWPT '09, pages 29–32, Stroudsburg, PA, USA. Association for Computational Linguistics.

Saers, M., Nivre, J., and Wu, D. (2010). Word alignment with stochastic bracketing linear inversion transduction grammar. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, HLT '10, pages 341–344, Stroudsburg, PA, USA. Association for Computational Linguistics.

Saers, M. and Wu, D. (2011). Principled induction of phrasal bilexica. In *Proceedings of the 15th Annual Conference of the European Association for Machine Translation, Leuven, Belgium, May*.

Saers, M., Wu, D., and Quirk, C. (2011). On the expressivity of linear transductions. In *Proceedings of the 13th Machine Translation Summit (MT Summit XIII)*.

Stolcke, A. (2002). SRILM-an Extensible Language Modeling Toolkit. In *Proceedings of the 7th International Conference on Spoken Language Processing (Interspeech-02)*.

Wu, D. (1997). Stochastic Inversion Transduction Grammars and Bilingual Parsing of Parallel Corpora. *Computational Linguistics*, 23(3):377–403.

Underspecified Query Refinement via Natural Language Question Generation

Hassan Sajjad[1] *Patrick Pantel*[2] *Michael Gamon*[2]
(1) Qatar Computing Research Institute, Qatar Foundation, Doha, Qatar *
(2) Microsoft Research, Redmond, WA, USA
hsajjad@qf.org.qa, ppantel@microsoft.com, mgamon@microsoft.com

ABSTRACT
Underspecified queries are common in vertical search engines, leading to large result sets that are difficult for users to navigate. In this paper, we show that we can automatically guide users to their target results by engaging them in a dialog consisting of well-formed binary questions mined from unstructured data. We propose a system that extracts candidate attribute-value question terms from unstructured descriptions of records in a database. These terms are then filtered using a Maximum Entropy classifier to identify those that are suitable for question formation given a user query. We then select question terms via a novel ranking function that aims to minimize the number of question turns necessary for a user to find her target result. We evaluate the quality of system-generated questions for grammaticality and refinement effectiveness. Our final system shows best results in effectiveness, percentage of well-formed questions, and percentage of answerable questions over three baseline systems.

KEYWORDS: Query refinement, question generation, search as a dialog.

*This work was conducted at Microsoft Research.

Proceedings of COLING 2012: Technical Papers, pages 2341–2356,
COLING 2012, Mumbai, December 2012.

1 Introduction

Vertical search engines, i.e., domain-specific search engines, retrieve ranked entities from an underlying database, often via unstructured keyword queries. Popular engines, such as Yelp, Bing Travel, IMDB, and Amazon share a common aspect with niche engines, such as BlueWine and OpticsPlanet: user queries are often underspecified. Whether because of the seemingly infinite inventory in the larger engines, or because of the esoteric collections in niche engines, or simply because users seek to browse a collection of results, underspecification is pervasive. When the set of specified entities exceeds the common limit of ten blue links, finding the desired entity can be a long and frustrating process.

For example, consider the query “blue polo with purple stripes" issued to the Bing Shopping engine. Pages of results are returned with correctly matching products. Browsing through these, especially in a mobile or handsfree scenario, can be prohibitively difficult. In most result sets, however, entities can form natural clusters based on important attributes. In our example, clusters can be formed based on price, designer, shirt patterns, etc. Some vertical search engines leverage such editorially defined clusters by forming a faceted search experience, where users can refine search results by selecting attribute values. Such experiences are expensive to build, require a heavily curated ontology to define the important attributes, and consequently are only feasible for large sites with significant revenue streams.

In this paper, we explore methods to automatically discover important attributes from unstructured data associated with entities in a database. Further, in a runtime scenario, we propose algorithms for selecting the best candidate attribute to ask about, based on various criteria, and we construct a binary natural language question for the user to answer. The main idea is to let the data guide the user in her search such that she can more quickly and effectively find her targeted entity.

Our first challenge is to select attribute terms from unstructured text that are in general important for a particular class of entities. We employ a Maximum Entropy technique for recognizing strings that appear to be good attribute terms. Our second challenge is to select the most appropriate attribute term given a user query and result set. Here we propose a set of ranking functions that optimize against the number of necessary questions to find the desired entity. On average, our ranking functions ensure finding the target entity using $\log_2(m)$ questions where m is the number of entities in the original result set. Finally, we transform the selected attribute term into a well-formed natural language question by matching against a set of lexico-syntactic question templates. We evaluate our systems' ability to quickly and effectively find target entities in an Xbox Avatar Editor scenario. Question well-formedness (as a function of factors such as grammaticality and spelling) is also evaluated.

The rest of the paper is organized as follows. Previous work on interactive question generation is summarized in Section 2 and in Section 2 we formally define our problem and the Xbox Avatar experimental dataset. We present our algorithms for finding question candidates in Section 3. Finally, we present our experimental results in Section 4 and conclude with a discussion of future work in Section 5.

2 Related Work

With the recent evolution in online search systems, question generation systems that improve the retrieval results in response to a user query have become an important area of research.

A major step in this process is *what to ask in the conversation?*. This involves clustering either the entire web (pre-retrieval method) (Voorhees, 1985) or only the retrieved results in response to a user query (post-retrieval method) (Cutting et al., 1993; Hearst et al., 1996; Allen et al., 1993; Leouski and Croft, 1996). In this paper, we follow the post-retrieval method to cluster the retrieved results. The description of the cluster is then used in the form of a question to the user.

Scatter/Gather (Cutting et al., 1993; Hearst et al., 1996) is a cluster-based approach which divides the documents in the collection into K clusters. Based on a query, a subset of clusters are selected which are dynamically reclustered (Carpineto et al., 2009). The scatter/gather method assigns attribute terms (descriptions) to the clusters from the feature vector or centroid. These attribute terms are difficult to use and understand especially in our scenario where we want to use them as questions to the user.

Suffix Tree Clustering ensures that the attribute terms of the clusters are meaningful and usable (Zamir and Etzioni, 1999). The idea is to extract terms from the text which are complete, self contained and meaningful. Zamir and Etzioni (1999) achieved this by taking frequent terms which are not crossing sentence boundaries. The problem with Suffix Tree Clustering is that only terms are used in the similarity metric for documents. This results in a decrease in the quality of clusters – especially for languages with free constituent order where parts of speech may come in various orders in a sentence (Carpineto et al., 2009; Masłowska, 2003).

Another class of algorithms focuses both on the quality of the cluster and the quality of its attribute terms. Vivísimo and Lingo (Osinski, 2006) are algorithms of this type. Lingo follows similar steps as that of Suffix Tree Cluster. It differs at query level where it finds abstract concepts from the query and matches them with frequent terms.

Kotov and Zhai (2010) add a question/answering feature to a search engine in order to improve search results and to guide the user to the output they are looking for. They generate a question for every candidate attribute and rank the questions using various heuristics, such as the number of query words that a question candidate matches. A set of top ranked questions are shown to the user who then selects the most relevant question according to their requirement. This user action leads to a modification of the original query and to an update of the search results. They require user input to select the best question. In this paper, we propose a ranking function to automatically rank the questions in a way that minimizes the number of questions needed to find the target entity.

Ontology-based term selection methods use dictionaries, thesauri and WordNet to learn the association between query terms and candidate terms (Bhogal et al., 2007; Hersh et al., 1992; Basili et al., 2007). Each term is mapped to a concept in an ontology. A term is a good candidate for selection if it belongs to the same concept as the query term. The drawback of using ontologies is that they are not available for all languages and for all domains, and their construction is expensive and time consuming. A detailed analysis of ontology-based query expansion can be found in Bhogal et al. (2007).

In contrast to most of the previous work on methods to browse web results, our domain is a web of entities. We learn good attribute terms from the unstructured data associated with the entities using a Maximum Entropy technique. We propose a set of ranking functions that optimize against the number of necessary questions to find the desired entity. Our

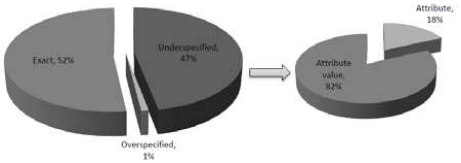


Figure 1: *Left*: Percentage of query types found in a random sample of the Avatar Dataset. *Right*: Percentage of relevant question types for underspecified queries.

method does not require any knowledge-based resource or user feedback, and uses only unstructured text.

In this section, we formally define the problem of helping a user quickly and effectively find her target result in a vertical search scenario.

2.1 Query Scope

Queries to vertical search engines can be categorized by the type of result set they generate: 1) an **exact query** is one that leads to a single result; 2) an **underspecified query** is one that leads to multiple results; and 3) an **overspecified query** is one that leads to zero results. For underspecified and overspecified queries, a follow-up user action is necessary in order to satisfy the information need.

The left pie chart in Figure 1 illustrates the percentage of query types in one dataset, introduced in Section 2.3. Although numbers vary by source and domain, the underspecified queries generally greatly outnumber overspecified queries.

Underspecified queries, which form the focus of this paper, result in multiple valid matches called a **confusion set**. When this set is large, it is difficult for a user to navigate through the matches to find her target result. There is an opportunity to help the user by building a system capable of interactively narrowing-down the confusion set.

2.2 Task Definition

We refer to the underlying data store being queried as a **Database**. We assume no structure within the database other than: 1) the set of records constituting the full search space; and 2) records belong to one or more semantic categories. For example, in a clothing database, each item of clothing is a record belonging to categories such as *pants, t-shirts, sweaters,* and *socks*. Some databases will contain other structured information such as attributes (e.g., t-shirts have a *color*, *price* and a *designer*) and relations (e.g., a particular t-shirt coordinates well with a set of pants). Although this information can be (and has been) leveraged for guiding users through a faceted search experience, we focus in this paper on the extraction of salient questions from **unstructured data**. Specifically, we aim to leverage user-generated comments and descriptions of database records as the source of information from which we will guide search users.

Problem Statement: Consider a database $\mathscr{D}$ where each record r is associated with a set of semantic categories C_r and a set of unstructured textual descriptions $S_r = \{s_{r1}, s_{r2}, ..., s_{rk}\}$.

Given a user query and a matching confusion set $R = \{r_1, r_2, ..., r_m\}$, our task is to ask a natural language question to the user that, based on the user's answer, best reduces the size of the confusion set.

For example, consider our query "*blue polo with purple stripes*" from our Bing Shopping scenario in Section 1. Suppose that there are 30 resulting matches in R consisting of 14 long-sleeved shirts and 16 short-sleeved shirts, as well as 10 shirts each from three fashion designers Ralph Lauren, Calvin Klein and Marc Jacobs. Suppose also that the user is seeking a short-sleeved polo from Ralph Lauren. Candidate questions to ask the user include: Q_1 "Are you looking for a long-sleeved shirt?"; Q_2 "Do you want a shirt by Marc Jacobs?"; and Q_3 "What fashion designer would you like?" The answer to Q_1 would result in cutting the confusion set nearly in half, whereas the answers to Q_2 and Q_3 would remove a third and two thirds of the confusion set, respectively. In this case then, Q_3 is the question that best reduces the size of the confusion set. Formulations of how to best reduce a confusion set are discussed in detail in Section 3.

Questions can be categorized as attribute questions or attribute value questions. An **attribute question** is one that seeks the value of an attribute of a semantic category. For our t-shirt category above, Q_3 is an example attribute question. It is seeking the value of the *fashion designer* attribute. An **attribute value question** is a binary question that asks if the target result has a particular attribute value. Q_1 and Q_2 are example attribute value questions, where Q_1 is asking if the desired sleeve length is *long* and Q_2 is asking if the desired fashion designer is *Marc Jacobs*. Attribute questions generally result in finer reductions in the confusion set, however they are cognitively harder to answer than their binary counterparts. Not considered in this paper are other more complex question types such as set questions (e.g., "*Is the shirt blue, red, or green?*") and compound questions (e.g., "*Is the shirt blue and short-sleeved?*").

We manually inspected the underspecified queries illustrated in Figure 1 along with their resulting confusion sets. We annotated each according to the question type (attribute or attribute value) of the question that would lead to the largest reduction in size of the confusion set. When both question types resulted in the same reduction of the confusion set, we preferred the binary attribute value type since it is easier for users to answer. The rightmost chart in Figure 1 illustrates the result of the study. In 82% of the cases, an attribute value question was deemed more appropriate than a value question. Based on this insight, we limit the scope of this paper to the automatic generation of attribute value questions.

Textual Grounding: We learn attribute value questions without access to any ontological structure in the database. In the textual descriptions S_r associated with a record r, we leverage the fact that many users will refer to the salient attributes and values of r. All unigrams, bigrams, and trigrams will be considered as candidate attributes and values, and we build statistical models to identify them.

2.3 Avatar Dataset

Very few people in the research community have access to the underlying databases powering vertical search engines such as Yelp, Bing Shopping, and Amazon. For the experiments in this paper we use a dataset that we believe is sufficiently similar to datasets used in vertical search engines on the Web, building upon publicly available data.

Category	Count	Category	Count
Trousers	54	Shoes	36
Wrist wear	16	Shirt	131
Ring	16	Nose	18
Mouth	27	Hat	34
Facial other	26	Gloves	16
Glasses	34	Hair	90
Facial hair	17	Eyes	45
Eyebrows	27	Ears	9
Earrings	34	Chin	9
Costume	27		

Table 1: Semantic categories in the Avatar Dataset.

We consider the domain of Xbox Avatars. Users of the Xbox gaming console associate themselves with an avatar that they can personalize with clothing, body features, and accessories. We refer to each item that can be personalized, such as clothing, as an **asset**. We utilize a dataset that was developed for a separate research project and will be publicly released in early 2013 as part of that project (Volkova et al., forthcoming). Below we briefly describe that project's process for creating the data.

There are a total of 666 assets in the dataset and each asset can belong to one of 19 categories. Each category contains 35 assets on average. Table 1 shows the categories and the number of assets in each category. We define $\mathcal{D}$ as this collection of assets.

The textual descriptions S are collected using Mechanical Turk. To ensure the quality of the annotation, a two-tier process (similar to Chen and Dolan (2011)) was followed. First, the annotations were manually inspected in order to select a group of trusted workers based on the quality of their annotations and their commitment to work with the project for a longer period of time. Only these trusted workers were then allowed to annotate.

For each asset, 50 descriptions were obtained from 50 different workers, where descriptions were produced in a task-independent manner (i.e., the annotators were not aware of the final use of these descriptions). Workers were asked to produce a description of the asset and its distinctive features. Sample descriptions of an asset from the category *Hat* are shown in Table 2.

Category attributes, such as the color of eyes, are grounded in the crowdsourced descriptions. Consider the category *Shirt* which has *sleeves*, *color* and *design* as general attributes. An instance (asset) of the category *shirt* contains the values of these attributes such as *long*, *brown* and *flag on chest* for the attributes *sleeves*, *color* and *design* respectively. These attributes and their values are not explicitly stored for assets. Instead, this information is grounded in the free text description of the assets.

We refer to this data set as the **Avatar Dataset**.

2.4 Summary

Our goal is to generate meaningful and well-formed attribute value questions for underspecified queries in the Avatar Dataset. In the next section, we describe our system architecture for question generation.

Descriptions of a hat asset
Green color flower design hat
Drab and yellow beanie with flowers
Green, flower print elastic clothing hat
Green toboggan with flowers and stripes
Green and yellow winter hat, daisy design on them and pompom on top

Table 2: Example descriptions for an Avatar asset from the category *Hat*.

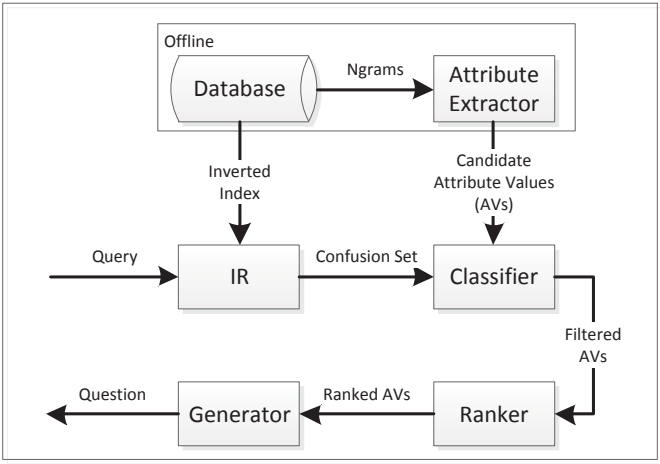


Figure 2: Question generation architecture.

3 Question Generation

Figure 2 outlines our question generation architecture. The input is an unstructured and underspecfied user query and the output is a well-formed attribute value question to the user. Offline processes are first applied to build an inverted index mapping each word in S (the textual descriptions defined in Section 2.2) to its corresponding assets, and to extract candidate attribute values from S. Given a user query, an IR system retrieves a confusion set consisting of matching assets in the database. A classifier is applied to select the appropriate candidate attribute values, which are then ranked according to how they are expected to reduce the size of the confusion set. The top ranked attribute value is formulated into a question by the generator. Below we describe each component in turn.

3.1 Attribute Extractor

As described in Section 2.2, category attributes are not explicitly modeled in the database and must instead be inferred from the textual descriptions associated with each asset. Each unigram, bigram, and trigram is considered as a potential attribute value. The Attribute

Extractor associates with each category the ngrams that appear to be most likely attribute value candidates.

If an attribute value, such as the sleeve length of a shirt, is salient to a category, then we hypothesize that strings referring to the attribute value, such as "short-sleeved" and "short sleeves", will occur more often in descriptions for assets of the category then for other assets. We therefore seek ngrams that are highly associated with each category, where association can be measured using statistics such as pointwise mutual information (PMI) and log-likelihood. In this paper, we use PMI. Given an ngram $n \in S$, we measure its association with a category c as:

$$\text{PMI}(n; c) = \log \frac{P(n, c)}{P(n)P(c)} \tag{1}$$

where $P(n, c)$ is the probability that an ngram in a description of an asset in c is n, $P(n)$ is the probability that an ngram in any description is n and $P(c)$ is the probability of any ngram occurring in a description in class c.

Ngrams with a PMI score higher than a predetermined threshold with a category are selected as candidate attribute values for that category. The resulting candidates are noisy and will be further filtered online by the Classifier component.[1] Table 3 lists examples of good and bad candidate attribute values for an asset from the *Shirt* category.

Entity	Blue half sleeved polo with stripes
Attribute values	
Answerable	Blue, half sleeved, polo, stripes
Unanswerable	polo with, with, with stripes

Table 3: Sample ngrams extracted as candidate attribute values by the Attribute Extractor for an asset from the *Shirt* category.

3.2 IR

Following (Salton, 1971), we build an inverted index mapping each ngram n in S to its corresponding asset r along with its tf-idf, defined as:

$$\text{tf-idf}(n, r) = tf(n, r) \times \log idf(n) \tag{2}$$

where $tf(n, r)$ is the frequency of n in S_r, and $idf(n)$ is the fraction of textual descriptions $s \in S$ containing n.

Let $\mathbf{r}$ be a vector of all ngrams in S_r where the value of each ngram is its tf-idf with r. Then, given a query q, we form a query vector $\mathbf{q}$ consisting of all ngrams in q, where the value of each feature is 1. Our IR component first retrieves from the inverted index all assets matching an ngram with q. For each matching asset r, we then compute a simple IR rank score as the cosine of the angle between $\mathbf{q}$ and $\mathbf{r}$:

$$\text{cosine}(\mathbf{q}, \mathbf{r}) = \frac{\sum_i q_i \cdot r_i}{\sqrt{\sum_i q_i^2 \cdot \sum_i r_i^2}} \tag{3}$$

[1] In this paper, we experimentally set the threshold to 1.

Binary features unigram, bigram, trigram, POS tag sequence of candidate, separate POS tag of every word in the candidate, candidate is a substring of the query, candidate contains the queried category
Real-valued features PMI score, log-likelihood score

Table 4: Features used in our Maximum Entropy classifier.

3.3 Classifier

The Attribute Extractor provides a shortlist of salient attribute value candidates, but we still need to further filter this list to arrive at our final list of candidates. For example, we still find spurious and rare candidates as well as non-constituent terms ("long and") in the candidates. In addition, we know that the ultimate usefulness of a salient attribute is dependent on the query: An attribute value candidate may be salient for a category, but given a specific query, it can still be useless as a refinement candidate. For example, a salient attribute value may not lead to any reduction in the confusion set or it could overlap with what is already asked for in the query, making it redundant for a refinement question.

To address these issues, we use a machine learned model that utilizes features derived from both the user's query and the Attribute Extractor provided list, and filters out attribute value candidates that are unanswerable or not relevant given the query.

The model we use is a Maximum Entropy classifier. The selection of the training data for this supervised classification approach is described in detail in Section 4. For every query in the training set, we retrieve a confusion set using the IR component (Section 3.2) and select candidates from the list provided by the Attribute Extractor that also match at least one description of the entities in the confusion set: an attribute value candidate that does not fulfil that criterion is by definition not able to serve as a disambiguator on the set. We use the Ranker module (Section 3.4) to select two attribute-value candidates for every query in the training set and annotate them with nine automatically extracted features as summarized in Table 4. The *unigram* feature indicates that the candidate term is a unigram, similarly for *bigram* and *trigram*. The feature *POS tag sequence of candidate* represents the part-of- speech tags of the words in a candidate. *POS tag of every word in the candidate* indicates all individual POS tags. Information about the relation of the candidate to the query and the category of the queried record is captured using the features *candidate is a substring of the query* and *candidate contains the queried category* respectively. We also utilize real-valued features for the PMI score and log-likelihood score of the candidates wrt the category. Finally, we manually annotate each example as a positive example (good candidate) or a negative example (bad candidate). The Maximum Entropy classifier is trained on the annotated examples. Given a list of candidate attribute values, the Classifier predicts whether the attribute values are good candidates or bad. The values which are good candidates according to the Classifier are then input to the Ranker module.

3.4 Ranker

Recall our problem definition from Section 2.2, which states that we aim to ask a question that best reduces the size of the confusion set, thus allowing a user to find her target result

with the minimum number of refinement questions. Our goal of finding the target result in an end to end system is difficult to evaluate without deployment. We instead introduce a ranking component that we reasonbaly expect to correlate with that goal of finding the right result. Our Ranker component orders the filtered attribute values from the Classifier according to how well each is expected to reduce the size of the confusion set. The top ranking attribute value will be used in formulating the final question asked to the user.

Since attribute value questions are binary, the most effective questions will be those that result in dividing the confusion set in half. This would result in an optimal interaction strategy where $\log_2 m$ questions are needed to guide a user through a confusion set of size m. We define our ranking score for an attribute value n and confusion set R, $\text{score}_R(n)$, as a real-valued function ranging from zero to one where *zero* indicates that n will cut the confusion set in half and *one* indicates that n will leave the confusion set unchanged. We seek questions that minimize this score. Formally:

$$\text{score}_R(n) = 2\left|\frac{\sum_{r\in R}\phi_R(n)f(r)}{\sum_{r\in R}f(r)} - 0.5\right|$$

where $\text{score}_R(n)\colon \mathbb{R} \to [0,1]$, $f(r)$ represents a weight function associated with each asset r in the confusion set, and $\phi_R(n)$ is the number of assets in the confusion set that hold the attribute value n (estimated by whether or not at least one textual description mentions n).

If $f(r) = 1$, $\text{score}_R(n)$ is minimized when n cuts the confusion set in half. Recall however that each asset in the confusion set has a relevance score assigned by the IR module (see Section 3.2). It is therefore reasonable to assume that items at the head of R will be more likely the target asset than items at the tail of R. We derive various definitions of the weight function $f(r)$ to capture this intuition:

$$f(r) = \begin{cases} M_1: & 1 \\ M_{rank}: & \frac{1}{\text{rank}(r)} \\ M_{ir}: & IR(r) \\ M_{dcg}: & \begin{cases} 1 \text{ for } \text{rank}(r) = 1 \\ \frac{\text{rank}(r)}{\log_2 \text{rank}(r)} \text{ otherwise} \end{cases} \end{cases}$$

M_1 considers all assets in the confusion set equally probable to be the target asset. M_{rank} weighs assets according to their rank in R and M_{ir} weighs them according to their cosine with the user query. Similarly, M_{dcg} weighs assets according to their gain discounted by rank position (similarly to that done in the Discounted Cumulative Gain (DCG) metric used primarily in IR).

The attribute value candidate with the lowest score is selected to form the final question. We use the Attribute Extractor score as tie-breaker (see Section 3.1).

In our experiments, we build our system using M_{rank}. M_1, M_{ir}, and M_{dcg} are used as evaluation metrics (see Section 4).

Template	POS
Should it be	JJ (Cat: NN)
Should they be	JJ (Cat: NNS)
Do you want	DT JJ NN
Do you want (a/an)	JJ NN NN
Are these	JJ VB NNS
Is it	RB VBN
Is it (a/an)	JJ VB NN
Does it have	NN NNS

Table 5: Question templates used to form a question

3.5 Generator

The final component in our system takes an attribute value as an input and produces a grammatical binary question from it. We use eight manually created question templates for this purpose. These question templates contain part-of-speech placeholders for the attribute value. Table 5 shows our question templates and a few examples of the part of speech sequences that can be used to complete each question template into a well-formed question. For question templates that only differ by an article (*a*, *an*), we check the first character of the question term to select the appropriate question template.

4 Experiments

4.1 Data Sets and Systems

From the Avatar Data Set described in Section 2.3, we sub-sampled a set of 160 assets for manual analysis. We refer to this set as the *Sampled Avatar Data Set*. For each asset we randomly chose one of the 50 available descriptions to serve as a set of random queries for the Sampled Avatar Data Set. Of the 160 queries, 75 were underspecified, which forms our *Underspecified Query Set*. Each query in that set is also associated with a confusion set as retrieved by the IR system from all assets in the Sampled Avatar Data Set.[2] We split the Underspecified Query Set into 50 query/asset pairs for training the maximum entropy classifier, called the *Classifier Training Set*, and 25 pairs for testing of the end-to-end system, called the *Test Set*.

We compare our system from Section 3, labeled *SYS*, against three baseline systems. For every query in the Test Set, we generate one question from each of the three baseline systems and our final system.

All three baseline systems extract attribute value candidates from the Attribute Extractor and that are relevant to the assets in the confusion set. As opposed to *SYS*, they do not use Classifier to filter the good candidate attribute values and use the output of the Attribute Extractor directly in Ranker. The systems select a question candidate based on the Ranking Function using M_{rank} as a weight function. PMI is used as a tie-breaker. The baseline systems differ with respect to the ngram size that they consider. System *B1* considers unigram attribute value candidates, *B2* considers bigram candidates and *B3* trigram candidates.

[2]The IR system was built using 49 textual descriptions per asset since one was reserved for the Underspecified Query Set.

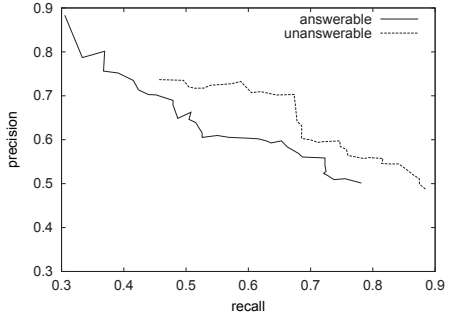


Figure 3: Precision-recall curve of the answerable and unanswerable classes.

4.2 Training of the Classifier

From the Classifier Training Set we need to derive a set of positive and negative examples for training. Positive examples are attribute values that are meaningful with respect to the given query and answerable if used in a question, whereas negative examples are either not meaningful or unanswerable. For simplicity, we refer to these examples as *answerable* and *unanswerable*, respectively.

Ideally, we would annotate all attribute value candidates for each query in the Classifier Training Set as answerable or unanswerable. In order to make the annotation task feasible, however, consider that the classifier needs to be optimized for its runtime task of filtering out attribute value pairs from the list provided by the Attribute Extractor. It is reasonable, therefore, to pick examples for annotation that are likely to be relevant in that scenario. For this purpose we first form the intersection of (i) the terms in the descriptions of the assets in the confusion set; and (ii) the list of attribute values produced by the Attribute Extractor. This produces a set of attribute value candidates just like the ones that the classifier will be exposed to at runtime. We also need to focus on finding answerable training cases, since the negative (unanswerable) cases are in the majority and hence much easier to come by. Of particular importance are cases that are selected by the Attribute Extractor and highly ranked by the Ranker, i.e. "borderline" candidates. We collect the top-10 attribute values (selected by the Attribute Extractor) for a given query and confusion set as measured by the Ranker (using M_{rank} as a weight function). From this top-10 set, we pick the top candidate and a random candidate to annotate as answerable or unanswerable. The resulting training set consists of 100 data points, 47 answerable and 53 unanswerable. We represent each training case as a feature vector as described in Section 3.3.

We evaluate the classifier based on 10-fold cross validation on the training set. The precision-recall curve of answerable (solid line) and unanswerable (dotted line) for different probability thresholds is shown in Figure 3. For our final system, we select a probability threshold greater than 0.7 for the answerable question terms.

4.3 Output Judgments

There are two properties of a system-generated question that we want to evaluate. First and most importantly, we want to know how good a final question is with respect to best dividing

	ANS	M_1	M_{rank}	M_{ir}	M_{dcg}
B1	0.6	0.6	0.63	0.62	0.74
B2	0.76	0.38	0.44	0.39	0.57
B3	0.68	0.41	0.48	0.46	0.61
SYS	**0.88**	**0.26**	**0.36**	**0.32**	**0.49**

Table 6: System effectiveness at reducing the size of the confusion set. Better systems will have a high ANS score (i.e., more questions are answerable) and low values for M_1, M_{rank}, M_{ir}, and M_{dcg}.

the confusion set. Second, we evaluate the grammaticality of the question to address the quality of our question generation component. During the latter task, we found some cases where the POS sequence of a question term does not match with any of the POS sequences allowed for question templates in our generation component. In these cases no question can be generated and we distinguish these cases from the "formulated" questions.

We designed an evaluation form which shows a query from the test set with its corresponding gold asset and four questions generated by the four systems. For every question, the judge has to select whether (i) the correct answer to the question is "yes" or "no"; or (ii) whether the question is answerable. The confusion set is then reduced based on the answer to the question by matching the attribute value ngram against the descriptions for each asset in the confusion set. We keep separate statistics for the unanswerable questions as determined by the judge's input to (ii). One of the authors served as the judge.

4.4 Results

Table 6 shows the results of all systems on our four metrics (recall that our final system uses M_{rank} in its Ranking Function, so the M_{rank} column is not a bona fide evaluation result and is only included for completeness). The "ANS" column refers to the percentage of questions that are answerable and relevant in context of the gold asset. Better systems maximize ANS and minimize M_1, M_{rank}, M_{ir}, and M_{dcg}.

Our final system shows a significant increase in the percentage of answerable question terms in comparison with the baseline systems. It also has the lowest scores for every evaluation metric, which shows that the question terms generated by our system divide the confusion set better than the baseline systems.

One author also rated every question for well-formedness. Each question was judged for grammaticality and for non-grammar errors (e.g., spelling errors), which are accounted for in a separate category "Other". Table 7 summarizes the results. Our system resulted in the most well-formed queries, with fewer mismatches with the POS templates described in Table 5 and fewer grammatical errors.

The lower percentage of well-formed questions for B2 and B3 reflects the fact that bigrams and trigrams tended to contain more rare POS sequences (such as non-constituents) that could not be accommodated by any question template.

4.5 Error Analysis

For a few test queries, our system produces meaningless questions like "shaped mustache" in response to a query "long and broad mustache with terror face look". This can happen

	Well-Formed	Errors		
		Template	Grammatical	Other
B1	0.52	**0.16**	0.24	0.16
B2	0.48	0.36	0.16	**0**
B3	0.28	0.52	0.20	**0**
SYS	**0.68**	**0.16**	**0.16**	0.04

Table 7: Question well-formedness. Error types include no matching POS template (i.e., the Generator component did not fire), grammatical errors, and other errors such as misspelling.

when the classifier rejects all question candidates as unanswerable and the system selects a question term from the list provided by the Attribute Extractor with the lowest M_{rank} score.

The answer to 30% of the questions resulted in the removal of the target asset from the confusion set. We found that there are two reasons for this, neither of which is a shortcoming of the system: either the user has made a mistake in answering the question or the description of the gold asset is not correct. We examined the latter cases and found that some textual descriptions associated with a few assets were inaccurate. For example, a pair of red pants was described as "red shorts" by a microtask worker. Similarly, some t-shirts are described as jackets. Suppose a user is searching for a pair of red pants and based on the description of the target asset and the confusion set, the system asks, "Are these shorts?" The correct response "no" will lead to the elimination of the target asset from the confusion set. Answering questions pertaining to the value of scalar facial attributes depends on a user's perception. The answer to questions "are these long lips?" or "are these bushy eyebrows?" depends on the user's notion of long and bushy. Here, again, the target asset may be wrongly excluded from the reduced confusion set. Similarly, the annotators sometimes confuse the position (left/right) of the attribute of an asset (e.g., "mole under the left/right eye").

Examining the ungrammatical questions, we found that the most common source of error is the use of a singular article with a mass noun, due to the fact that we neglected to distinguish between mass and count nouns in our question templates.

5 Conclusion

We have proposed a question generation system that produces refinement questions for underspecified queries from unstructured text in a vertical search scenario. We applied our system to an Xbox Avatar personalization dataset and found that compared to three baselines, our system offers the best reductions in confusion set size and the highest percentage of well-formed natural language questions.

There are many opportunities for future research in this area and on the Avatar Dataset. Some examples include automatic detection of inconsistent asset descriptions, refinements in attribute value extraction, and improved question generation including the generation of more complex questions.

Acknowledgments

The authors thank Bill Dolan, Chris Brockett, Yun-Cheng Ju, and Svitlana Volkova for sharing their Avatar Dataset and providing vision and valuable guidance.

References

Allen, R. B., Obry, P., and Littman, M. (1993). An interface for navigating clustered document sets returned by queries. In *Proceedings of the conference on Organizational computing systems*, COCS '93, New York, NY, USA.

Basili, R., Cao, D. D., Giannone, C., and Marocco, P. (2007). Data-driven dialogue for interactive question answering. In *AI*IA*.

Bhogal, J., Macfarlane, A., and Smith, P. (2007). A review of ontology based query expansion. *Information Processing and Management*, 43.

Carpineto, C., Osinski, S., Romano, G., and Weiss, D. (2009). A survey of web clustering engines. *ACM Computing Surveys*, 41(3).

Cutting, D. R., Karger, D. R., and Pedersen, J. O. (1993). Constant interaction-time scatter/gather browsing of very large document collections. In *Proceedings of the 16th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '93, New York, NY, USA.

Hearst, M. A., Pedersen, J. O., and Alto, P. (1996). Reexamining the cluster hypothesis : Scatter / gather on retrieval results. *Computing Systems*.

Hersh, W. R., Hickam, D. H., and Leone, T. (1992). Words, concepts, or both: optimal indexing units for automated information retrieval. In *Proceedings of the 16th Annual Symposium on Computer Applications in Medical Care*, Oregon Health Sciences University, Portland.

Kotov, A. and Zhai, C. (2010). Towards natural question guided search. In *Proceedings of the 19th international conference on World wide web*, WWW '10, New York, NY, USA. ACM.

Leouski, A. V. and Croft, W. B. (1996). An evaluation of techniques for clustering search results. Technical report, Department of Computer Science, University of Massachusetts.

Masłowska, I. (2003). Phrase-based hierarchical clustering of web search results. In *Proceedings of the 25th European conference on IR research*.

Osinski, S. (2006). Improving quality of search results clustering with approximate matrix factorisations. In *Proceedings of the 28th European conference on IR research*.

Salton, G. (1971). *The SMART Retrieval System–Experiments in Automatic Document Processing*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA.

Voorhees, E. M. (1985). The cluster hypothesis revisited. In *Proceedings of the 8th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '85, New York, NY, USA.

Zamir, O. and Etzioni, O. (1999). Grouper: a dynamic clustering interface to web search results. In *Proceedings of the 18th international conference on World Wide Web*, WWW '99, New York, NY, USA.

Joint English Spelling Error Correction and POS Tagging for Language Learners Writing

Keisuke Sakaguchi, Tomoya Mizumoto,
Mamoru Komachi, Yuji Matsumoto
Graduate School of Information Science
Nara Institute of Science and Technology
8916-5, Takayama, Ikoma, Nara 630-0192, Japan
{keisuke-sa, tomoya-m, komachi, matsu}@is.naist.jp

Abstract

We propose an approach to correcting spelling errors and assigning part-of-speech (POS) tags simultaneously for sentences written by learners of English as a second language (ESL). In ESL writing, there are several types of errors such as preposition, determiner, verb, noun, and spelling errors. Spelling errors often interfere with POS tagging and syntactic parsing, which makes other error detection and correction tasks very difficult. In studies of grammatical error detection and correction in ESL writing, spelling correction has been regarded as a preprocessing step in a pipeline. However, several types of spelling errors in ESL are difficult to correct in the preprocessing, for example, homophones (e.g. **hear/here*), confusion (**quiet/quite*), split (**now a day/nowadays*), merge (**swimingpool/swimming pool*), inflection (**please/pleased*) and derivation (**badly/bad*), where the incorrect word is actually in the vocabulary and grammatical information is needed to disambiguate.

In order to correct these spelling errors, and also typical typographical errors (**begginning/beginning*), we propose a joint analysis of POS tagging and spelling error correction with a CRF (Conditional Random Field)-based model. We present an approach that achieves significantly better accuracies for both POS tagging and spelling correction, compared to existing approaches using either individual or pipeline analysis. We also show that the joint model can deal with novel types of misspelling in ESL writing.

Keywords: Part-of-Speech Tagging, Spelling Error Correction.

Proceedings of COLING 2012: Technical Papers, pages 2357–2374,
COLING 2012, Mumbai, December 2012.

1 Introduction

Automated grammatical error detection and correction have been focused on natural language processing (NLP) over the past dozen years or so. Researchers have mainly studied English grammatical error detection and correction of areas such as determiners, prepositions and verbs (Izumi et al., 2003; Han et al., 2006; Felice and Pulman, 2008; Lee and Seneff, 2008; Gamon, 2010; Dahlmeier and Ng, 2011; Rozovskaya and Roth, 2011; Tajiri et al., 2012). In previous work on grammatical error detection and correction, spelling errors are usually corrected in a preprocessing step in a pipeline. These studies generally deal with **typographical** errors (e.g. **begginning/beginning*). In ESL writing, however, there exist many other types of spelling errors, which often occur in combination with, for example, **homophone** (**there/their*), **confusion** (**form/from*), **split** (**Now a day/Nowadays*), **merge** (**swimingpool/swimming pool*), **inflection** (**please/pleased*), and **derivation** (**badly/bad*) errors. Unlike typographical errors, these spelling errors are difficult to detect because the words to be corrected are possible words in English.

Previous studies in spelling correction for ESL writing depend mainly on edit distance between the words before and after correction. Some previous works for correcting misspelled words in native speaker misspellings focus on homophone, confusion, split, and merge errors (Golding and Roth, 1999; Bao et al., 2011), but no research has been done on inflection and derivation errors.

One of the biggest problems in grammatical error detection and correction studies is that ESL writing contains spelling errors, and they are often obstacles to POS tagging and syntactic parsing. For example, POS tagging fails for the following sentence[1]:

Input:
... it is **verey/very *convent/convenient* for the group.
without spelling error correction:
... it/PRP, is/VBZ, verey/PRP, convent/NN ...
with spelling error correction:
... it/PRP, is/VBZ, very/RB, convenient/JJ ...

Conversely, spelling correction requires POS information in some cases. For instance, the sentence below shows that the misspelled word **analysys/analyses* is corrected according to its POS (NNS), while it is difficult to select the best candidate based only on edit distance (analysis/NN or analyses/NNS).

Input:
... research and some **analysys/analyses*.
when assigning POS tags:
... and/CC, some/DT, analysys/NNS ...
candidates and their POS:
['analysis/NN', 'analyses/NNS']

In order to detect and correct errors in ESL writing, spelling correction is essential, because sentences with misspelled words cannot be parsed properly. However, the conventional pipeline for grammatical error detection and correction has a limitation due to the different types of spelling errors and the unavailability of contextual information, which results in failures in the subsequent POS tagging and syntactic parsing (Figure 1(1)).

In this work, we propose a joint model for spelling correction and POS tagging (Figure 1(2)). The model is based on morphological analysis, where each node in a lattice has both POS and

[1] We use Penn treebank-style part-of-speech tags.

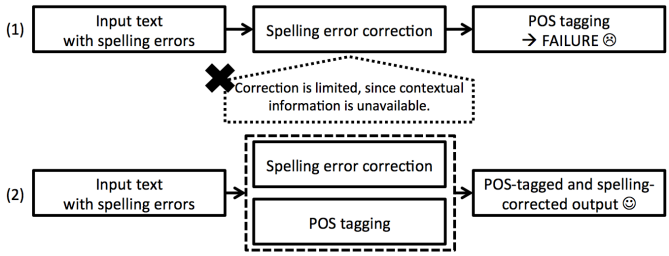


Figure 1: A limitation of pipeline analysis (1), and our proposed joint model (2).

spelling information as features. Because of these features, our method can deal with not only typographical errors but also homophones, confusion, split, merge, inflection and derivation errors. Also, higher accuracy with spelling correction improves POS tagging. We evaluated the joint model with two different ESL learners' error-annotated corpora, with the results showing 2.1% and 3.8% improvement in F-values of POS tagging for the corpora, and 5.0% in F-value of spelling errors. The results significantly outperform baseline and pipeline.

There are three main contributions described in this paper:

1. This is the first joint model for assigning POS tags and correcting misspelled words simultaneously.

2. Our work shows that the joint model improves the accuracy of both POS tagging and spelling correction for ESL writing compared to conventional pipeline methods.

3. This is the first model which is able to correct a wide range of misspelled words, including misspellings due to inflection and derivation errors.

In the following, we first present previous research done on grammatical error correction, spelling correction, and joint analysis (Section 2), and then describe our proposed method in detail (Section 3). The experimental setting and the results are presented in Section 4, and error analysis is given in Section 5. Finally, we conclude in Section 6.

2 Related works

In spelling error correction, the main concern is how to extract confusion pairs that consist of words before and after correction. A number of studies depend on such edit distance between written and corrected words as *Levenshtein Distance* (LD), *Longest Common Subsequence* (LCS) string matching, and pronunciation similarities (Kukich, 1992; Brill and Moore, 2000; Islam and Inkpen, 2009; Bao et al., 2011; Toutanova and Moore, 2002). In order to cover more misspelled words, many spelling errors were collected from web search queries and their results (Chen et al., 2007; Gao et al., 2010), click through logs (Sun et al., 2010), and users' keystroke logs (Baba and Suzuki, 2012). Note that previous studies for spelling correction described above focus on errors made by native speakers rather than second language learners, who show a wider range of misspellings with, for example, split, merge, inflection and derivation errors.

In most grammatical error detection and correction research, spelling error correction is performed before such linguistic analysis as POS tagging and syntactic parsing. Spelling correction as preprocessing generally uses existing spelling checkers such as GNU Aspell[2] and Jazzy[3], which depend on edit distance between words before and after correction. Then, candidate words are often re-ranked or filtered using a language model. In fact, in the Helping Our Own (HOO) 2012 (Dale et al., 2012), which is a shared task on preposition and determiner error correction, highly-ranked teams employ the strategy of spelling correction as preprocessing based on edit distance.

Some recent studies deal with spelling correction at the same time as whole grammatical error correction. For example, (Brockett et al., 2006) presents a method to correct whole sentences containing various errors, applying a statistical machine translation (SMT) technique where input sentences are translated into correct English. Although this approach can deal with any type of spelling errors, it suffers from a poverty of error-annotated resources and cannot correct misspelled words that have never appeared in a corpus. Similarly, (Park and Levy, 2011) propose a noisy channel model to correct errors, although they depend on a bigram language model and do not use syntactic information. A discriminative approach for whole grammatical error correction is also proposed in a recent study (Dahlmeier and Ng, 2012) where spelling errors are corrected simultaneously. In terms of spelling error types, however, typographical errors using GNU Aspell are dealt with, but not other misspelling types such as split and merge errors. Our proposed model uses POS features in order to correct spelling. As result, a wider range of spelling errors such as inflection and derivation errors can be corrected. Inflection and derivation errors are usually regarded as grammatical errors, not spelling errors. However, we include inflection and derivation error correction in our task, given the difficulty of determining whether they are grammatical or spelling errors, as will be explained in Section 4.1.

Joint learning and joint analysis have received much attention in recent studies for linguistic analysis. For example, the CoNLL-2008 Shared Task (Surdeanu et al., 2008) shows promising results in joint syntactic and semantic dependency parsing. There are also models that deal with joint morphological segmentation and syntactic parsing in Hebrew (Goldberg and Tsarfaty, 2008), joint word segmentation and POS tagging in Chinese (Zhang and Clark, 2010), and joint word segmentation, POS tagging and dependency parsing in Chinese (Hatori et al., 2012). These studies demonstrate that joint models outperform conventional pipelined systems. Our work applies for the first time a joint analysis to spelling correction and POS tagging for ESL writing in which input sentences contains multiple errors, whereas previous joint models deal only with canonical texts.

3 Joint analysis of POS tagging and spelling correction

In this section, we describe our proposed joint analysis of spelling error correction and POS tagging for ESL writing. Our method is based on Japanese morphological analysis (Kudo et al., 2004), which disambiguates word boundaries and assigns POS tags using re-defined Conditional Random Fields (CRFs) (Lafferty et al., 1999), while the original CRFs deal with sequential labeling for sentences with word boundaries fixed. We use the re-defined CRFs rather than the original CRFs because disambiguating word boundaries is necessary for split and merge error correction. In terms of decoding, our model has a similar approach to the decoder proposed by (Dahlmeier and Ng, 2012), though the decoder by Dahlmeier and Ng uses beam search. In (Kudo et al., 2004), they define CRFs as the conditional probability of an output path $\mathbf{y} = \left(\langle w_1, t_1 \rangle, ..., \langle w_{\#\mathbf{y}}, t_{\#\mathbf{y}} \rangle \right)$, given

[2] http://aspell.net/
[3] http://jazzy.sourceforge.net/

an input sentence $\mathbf{x}$ with words w and labels t:

$$P(\mathbf{y}|\mathbf{x}) = \frac{1}{Z_{\mathbf{x}}} \exp\left(\sum_{i=1}^{\#\mathbf{y}} \sum_{k} \lambda_k f_k\big(\langle w_{i-1}, t_{i-1} \rangle, \langle w_i, t_i \rangle \big) \right)$$

where $\#\mathbf{y}$ is the number of tokens according to the output sequence, and $Z_{\mathbf{x}}$ is a normalization factor for all candidate paths $\mathcal{Y}(\mathbf{x})$,

$$Z_{\mathbf{x}} = \sum_{\mathbf{y}' \in \mathcal{Y}(\mathbf{x})} \exp\left(\sum_{i=1}^{\#\mathbf{y}'} \sum_{k} \lambda_k f_k\big(\langle w'_{i-1}, t'_{i-1} \rangle, \langle w'_i, t'_i \rangle \big) \right)$$

Here, $f_k\big(\langle w_{i-1}, t_{i-1} \rangle, \langle w_i, t_i \rangle \big)$ is a feature function of the i-th token $\langle w_i, t_i \rangle$ and its previous token $\langle w_{i-1}, t_{i-1} \rangle$. λ_k is the weight for the feature function f_k. When decoding, the most probable path $\hat{\mathbf{y}}$ for an input sentence $\mathbf{x}$ is

$$\hat{\mathbf{y}} = \operatorname*{argmax}_{\mathbf{y} \in \mathcal{Y}(\mathbf{x})} P(\mathbf{y}|\mathbf{x})$$

which can be found with the Viterbi algorithm.

The lexicon consists of basic information: surface form, its base form, and its POS tag. In order to deal with misspelled words, we extend the format of the lexicon appending correctness of spelling and correct form in conjunction with the basic information. With the extended format, we prepare a misspelling dictionary in addition to the existing English dictionary. Here are examples of lexical entries in both dictionaries:

Examples of correct lexicon:
writing,-40,VB,write,VBG,CORR,*
English,152,NN,English,NNP,CORR,*
Examples of lexicon of spelling errors:
absoletuly,-18,RB,absolutely,RB,INCO,absolutely
difficultly,36,JJ,difficult,JJ,INCO,difficult

where each entry consists of a surface form, followed by cost of the word, POS group[4], base form, POS, CORR (correct) / INCO (incorrect) spelling error flag, and correct spelling form. If the flag is CORR, the correct spelling form is written as '*'. In the above examples for the lexicon of spelling errors, **absoletuly/absolutely* is a typographical error and **difficultly/difficult* is a derivation error. The unigram costs in the correct lexicon and POS bigram costs are calculated as a result of learnt weights in the CRFs, and the detail of weights learning of the CRFs is found in Kudo et al.(2004). The cost in the lexicon of spelling errors is obtained based on the corresponding correct form. In other words, the model is able to decode unseen spelling errors, if correct candidates for the misspelled word exist in the correct lexicon. The way to construct a lexicon of spelling errors is described in detail in Section 4. With the additional lexicon, where the cost for each entry is determined, we can decode sentences including spelling errors, with simultaneous spelling correction and POS tagging. Algorithm 1 shows a brief overview of our proposed model for decoding. Figure 2 shows examples of the decoding process, where **beggining/beginning*, **Auguest/August*, and **swimingpool/swimming pool* are misspelled. Without a misspelling dictionary, we fail to decode spelling error words and to assign POS tags (as shown in dotted lines in Figure 2). Because we prepare a misspelling dictionary as explained above, we can decode **begginning* as *beginning*,

[4]POS groups are a coarse version of Penn Treebank POS tags. For example, JJ, JJR and JJS are merged into JJ.

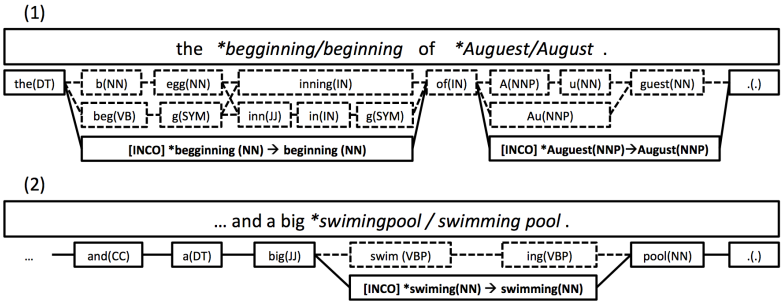


Figure 2: Samples of decoding process with proposed method. "[INCO]" is a misspelling flag.

Algorithm 1 Proposed joint POS tagging and spelling correction algorithm

Input: *Lexicon, Sentence* // *Sentence* ignores blanks between words.
Output: *Best path for the input sentence*
  Lattice $= \phi$
  $i = 0$ // i is letter index of a *Sentence*.
  repeat
    for each node ending with Sentence[i] **do**
      right_nodes = Lexicon.commonPrefixSearch(Sentence[i+1:])
      for for each right_node in right_nodes **do**
        Append right_node with unigram cost into *Lattice*
        Append the edge between node and right_node with POS bigram cost into *Lattice*
      end for
    end for
    i++
  until The end of the input sentence
  Best_Path = **Decode_Viterbi**(*Lattice*)
  return Best_Path

**Auguest* as *August* in Figure 2(1) (shown in solid lines). Furthermore, since the re-defined CRFs deal with word boundary ambiguity, this model is suitable for split and merge spelling error detection and correction as shown in Figure 2(2). In Figure 2(2), where **swimingpool* is a merge error, the misspelled word is split into **swiming/swimming* and *pool*, and corrected from **swiming* to *swimming*.

4 Experiment

4.1 Data

For our experiments, we use two different ESL learners' corpora: the Cambridge Learners Corpus First Certificate in English (CLC FCE) dataset (Yannakoudakis et al., 2011) and the Konan-JIEM learner corpus (KJ corpus) (Nagata et al., 2011). Table 1 shows the statistics of the two corpora. The CLC FCE dataset, which is one of the largest and most commonly used ESL learners' corpora, consists of 1,244 files, and each file consists of two essays with gold-standard error annotation.

	CLC FCE dataset	KJ corpus
# Essays	2,488	233
# Sentences	28,033	3,199
# Tokens	423,850	25,537
1st language	16 languages	Japanese
Error Tagged	Yes	Yes* (Spelling errors are not tagged.)
POS Tagged	No	Yes

Table 1: Statistical overview of the datasets: CLC FCE dataset and KJ corpus.

CLC FCE dataset		KJ corpus	
Error Types	%	Error Types	%
Verb	20.8	Noun	27.6
Punctuation	14.2	Verb	23.9
Spelling	10.7	Article	18.4
Preposition	10.5	Preposition	13.0
Determiner	9.5	Adjective	4.1
Noun	9.3	Adverb	3.4

Table 2: The top 6 error types in CLC FCE dataset and KJ corpus.

The KJ corpus consists of more than 200 essays written by Japanese ESL learners. This is the only dataset where POS tags are assigned for ESL writing. Table 2 shows the proportion of error types for the two datasets. Note that the KJ corpus does not contain error tags for spelling errors and other ungrammatical errors such as punctuation errors.

In terms of spelling errors in the CLC FCE dataset, there are 'S'(spelling) and 'SX'(spelling confusion) error tags. The number of 'S' and 'SX' are 4,922 and 789 respectively. Under the definition of spelling error types in our work, *homophone*, *confusion*, *split*, and *merge* errors are included in 'S' and 'SX' error annotations. There are also 760 'I' (inflection) and 1,913 'D' (derivation) error tags that contain spelling errors such as **usefull/useful* and **suppost/supposed*, in addition to clear examples of inflection/derivation errors (e.g. **badly/bad*). In total, there are 8,349 spelling errors in the CLC FCE dataset, which accounts for 1.9% of the spelling errors in the whole corpus. The distribution of spelling error types is shown in Table 3. A confusion pair is excluded when the original word length is less than 3 letters or when the word is a pronoun, in order to avoid highly frequent words being corrected. We also exclude a confusion pair when the pair derives from semantic confusion (e.g. **dead/killed* and **although/however*).

4.2 Methodology

For training and decoding, we use the MeCab[5] toolkit, a CRF-based POS and morphological analyzer. Table 4 shows the feature template for MeCab (i.e. CRF) training. As mentioned in Section 3, we also use the POS bigrams as the cost of a sequential edge.

The CLC FCE dataset is used for training, development and test sets, where files are randomly divided into 1,000 for training, 100 for development and 100 for test sets. For statistical analysis,

[5]MeCab 0.98 `http://mecab.googlecode.com/svn/trunk/mecab/doc/index.html`

Spelling Error Types	Numbers	%
Typographical	4,859	58.2
Homophone or Confusion	789	9.5
Split	17	0.2
Merge	11	0.1
Inflection	760	9.1
Derivation	1,913	22.9
Total	8,349	

Table 3: A distribution of spelling error types in the CLC FCE dataset.

we take five different samples from the CLC FCE dataset. We use the development set for deciding a hyper-parameter c during MeCab training. We use the KJ corpus only as a test set for POS tagging, because it does not have a gold standard for spelling errors. For evaluating the KJ corpus, we use the same training and development sets of the CLC FCE dataset explained above.

Since the CLC FCE dataset does not contain POS tags, we need to assign POS tags for the corrected sentences in the CLC FCE corpus. We use a MeCab trained on Penn Treebank[6] with the NAIST English dictionary (NAIST edic)[7] (hereafter referred to as MeCab-PTB). The accuracy of MeCab-PTB is 0.974 of precision, 0.980 of recall, and 0.977 of F-value as a result of a preliminary experiment[8]. We assign POS tags by MeCab-PTB for the CLC FCE training set in which all sentences are corrected, and the output is used as the POS-tagged CLC FCE training set (CLC-POS-Train). Then, we train again MeCab-PTB on both Penn Treebank and CLC-POS-Train (referred to as MeCab-CLC).

In order to analyse spelling errors, we extract the pairs of misspelled and corrected words in the CLC FCE training set, so as to develop a lexicon of spelling errors (hereafter LexTrain) as shown in Section 3. The cost for each misspelled entry is extracted from cost-learnt NAIST edic in MeCab-CLC. For example, when the misspelled and corrected word is **bok/book*, the word '*book*' is found in learnt NAIST edic as *book/NN* and *book/VB*. Since the costs for these two candidates are determined, we can construct a lexicon of spelling errors with the flag "INCO". For each of the five training sets, 4,656 entries on average are extracted for the lexicon of spelling errors, including all spelling error types.

For the CLC FCE test set, since we cannot add the gold-standard pairs of misspelled word and its correction directly into the lexicon, we obtain candidates for misspelled words in a test set using GNU Aspell[9]. If the pair of misspelled word and its candidate does not exist in LexTrain, we add the pair into a new lexical dictionary (LexTest), where the cost of learning, POS group, and POS are extracted from learnt NAIST edic in MeCab-CLC. As is the case with LexTrain, all possible entries are added into LexTest for the words that have several POS tags (e.g. NN and VB for *book*). If a candidate word does not exist in learnt NAIST edic, we do not add its pair because no

[6]The Penn Treebank Project Release 2 `http://www.cis.upenn.edu/treebank/`

[7]NAIST-edic-0.2.0 `http://sites.google.com/site/masayua/p/naist-edic`

[8]We use the sections 0-18 of the Penn Treebank for training and sections 22-24 for evaluation. However, it is difficult to make a fair comparison of the result with other PTB POS taggers, since we use *.pos files for training and test sets instead of the *.mrg files that are generally used. We use *.pos files because they have more data.

[9]Because the CLC FCE contains some words, such as proper nouns, that GNU Aspell does not recognize, we add all words in correct sentences of the CLC FCE training set into the GNU Aspell dictionary.

Feature description (Unigram)	Feature description (Bigram)
WORD[i]	WORD[i-1] + WORD[i]
WORD[i] + POS[i]	WORD[i-1] + WORD[i] + POS[i-1]
WORD[i] + POS_group[i]	WORD[i-1] + WORD[i] + POS[i]

Table 4: Feature template of *i*-th token used for training CRF.

information about the cost for the candidate is available. We develop MeCab-CLC+Lex by adding the indices of LexTrain and LexTest into MeCab-CLC.

In our experiment, we analyse test set sentences, where all but spelling errors are corrected beforehand. We compare three conditions for POS tagging: POS-BASELINE, POS-PIPELINE, and POS-JOINT. For POS-BASELINE and POS-PIPELINE, we use MeCab-CLC to analyse test set sentences. In the case of POS-PIPELINE, unknown words in the test set are replaced by the best candidate suggested by GNU Aspell and re-ranked by a 5-gram language model built on the Google 1T Web corpus (Brants and Franz, 2006) with IRSTLM toolkit[10]. In POS-JOINT, we use MeCab-CLC+Lex to analyse the test set.

For spelling correction, we compare two conditions: SP-BASELINE and SP-JOINT. We use GNU Aspell as SP-BASELINE, and the output from POS-JOINT is used for SP-JOINT. With respect to gold standard POS and spelling correction, we analyse the error-free test set with MeCab-PTB.

4.3 Evaluation

We evaluated the performance of POS tagging and spelling correction by computing precision, recall, and F-value. In POS tagging, for each sentence, we count the number of words in the gold standard (*REF-POS*) as N_{REF} and the number of words in system output (*SYS-POS*) as N_{SYS}. In addition, we count the the number of words when the word tokenization and POS tagging match exactly between gold standard and system output (*CORR-POS*) as N_{CORR}. For example, when an input sentence is "*Are you *studing a lot?*" and its reference and output are as follows,

REF-POS: {Are/VBP, you/PRP, studying/VBG, a/DT, lot/NN, ?/? }
SYS-POS: {Are/VBP, you/PRP, stud/JJ, ing/NN, a/DT, lot/NN, ?/?}
CORR-POS: {Are/VBP, you/PRP, a/DT, lot/NN, ?/?}

then N_{REF} is 6, N_{SYS} is 7, and N_{CORR} is 5.

With respect to spelling correction, along with the POS tagging, we count N_{REF}, N_{SYS}, and N_{CORR}, looking at the spelling of tokenized words. N_{REF} is the number of gold-standard spelling correction pairs in (*REF-SP*), N_{SYS} is the number of corrected pairs in the system output (*SYS-SP*), and N_{CORR} is the number of pairs in the system output that correctly identifies the gold standard (*CORR-SP*). For instance, when an input is "*There aren't *convinent *appliaces in their houses yet.*" and the output is "*There aren't **convenient places** in **there** houses yet.*", the result is as follows:

REF-SP: {*appliaces/appliances, *convinent/convenient}
SYS-SP: {*appliaces/places, *convinent/convenient, *their/there}
CORR-SP: {*convinent/convenient}

[10] irstlm 5.70 `http://sourceforge.net/projects/irstlm/files/irstlm/`

		Precision	Recall	F-value
CLC FCE	POS-BASELINE	0.950	0.971	0.960
	POS-PIPELINE	0.961 (+1.1%)	0.976 (+0.5%)	0.968 (+0.8%)
	POS-JOINT	**0.982 (+3.2%)***†	**0.979 (+0.8%)**	**0.981 (+2.1%)***†
KJ corpus	POS-BASELINE	0.794	0.857	0.824
	POS-PIPELINE	0.827 (+3.3%)*	0.857 (± 0.0%)	0.842 (+1.7%)*
	POS-JOINT	**0.853 (+5.9%)***†	**0.871 (+1.4%)***†	**0.862 (+3.8%)***†

Table 5: Experimental result on POS tagging. Statistically significant improvements over the baseline are marked with an asterisk (*), and those over the pipeline are marked with a dagger (†), where $p < 0.05$.

		Precision	Recall	F-value
CLC FCE	SP-BASELINE	**0.519**	0.427	0.468
	SP-JOINT	0.445 (-7.3%)*	**0.622 (+19.5%)***	**0.519 (+5.0%)***

Table 6: Experimental result on spelling error correction. Statistically significant improvements over the baseline are marked with an asterisk (*), where $p < 0.05$.

and therefore, in this case, N_{REF} is 2, N_{SYS} is 3, and N_{CORR} is 1.

Precision, Recall, and F-value are computed by N_{REF}, N_{SYS}, and N_{CORR} as the following equations.

$$Precision = \frac{N_{CORR}}{N_{SYS}}, \quad Recall = \frac{N_{CORR}}{N_{REF}}, \quad F\text{-}value = \frac{2 \times Precision \times Recall}{Precision + Recall}$$

4.4 Result and Analysis

The experimental results of POS tagging is shown in Table 5. From the table, we make the following observations.

First, the joint model and the pipeline perform better than the baseline both in the CLC FCE dataset and the KJ corpus. For the two corpora, the joint model achieves 2.1% and 3.8% improvements and the pipeline achieves 0.8% and 1.7% in F-value, although only POS-JOINT shows statistical significance. Second, the result of the KJ corpus is lower than that of the CLC FCE dataset. This may be because there is a slight difference in segmentation and POS format in the KJ corpus. For example, some words are assigned multiple POS tags such as *everyday/DT-NN* and *logout/VBN-RP*. Furthermore, in the KJ corpus, there are a lot of Japanese words written in Roman letters (e.g. *Onigiri* (*rice ball* in English), *himawari* (*sunflower*)), which make it difficult to segment words and assign POS tags in this corpus. Third, the result shows that our joint analysis performs better in POS tagging than the pipeline in all three metrics for the two ESL learners' corpora. This is because our proposed model assigns POS tags and corrects spelling errors simultaneously, and the joint model can correct not only typographical spelling errors but also homophone, split, merge, inflection, derivation, and confusion errors. Finally, the overall results in the CLC FCE dataset show relatively high values for POS tagging. This may be because the topics in the CLC FCE dataset are limited and there are categorical overlaps between training and test sets.

In terms of spelling error correction, Table 6 presents our experimental results. Overall, the joint model performs better in recall (+19.5%) and F-value (+5.0%), whereas the precision decreases

from 0.519 to 0.445. The result of higher recall and less precision is not surprising, since the joint model can deal with all types of spelling errors whereas only typographical errors are corrected in the baseline.

5 Discussion

In this section, we look at our experimental results in detail and discuss the contribution of our work.

First, looking at the cases when POS tagging and spelling error correction are successfully analysed, we find that the joint model (POS-JOINT) works well for all 7 types of spelling errors we defined. Figure 3 shows successful examples of the 7 error types. For instance, (1) in Figure 3 shows that the word **surprice* is misspelled and split into two words *sur* and *price* in the baseline (POS-BASELINE), whereas the joint model corrects the spelling error and assigns a POS tag successfully. Of course, these typographical errors can be corrected using conventional ways such as edit distance, and in fact these errors are also corrected in the pipeline (POS-PIPELINE), where misspelled words are corrected using edit distance before POS tagging.

The rest of the examples, (2) to (7), in Figure 3 are harder to correct if we depend only on edit distance. However, as pointed out above, the joint model can correct these different kinds of spelling errors. In (2) in Figure 3, the homophone error **hear/here* is corrected in the joint analysis, since the model compares the path costs between the POS sequences of "**... am(VBP)-hear(VB)-to(TO) ...**" and "**... am(VBP)-here(RB)-to(TO) ...**", while the baseline and pipeline cannot figure out the homophone spelling error. The confusion and split errors such as the examples in Figures 3(3) and 3(4) are corrected successfully with the joint model, as is the case of homophone errors. When it comes to merge errors as shown in Figure 3(5), a misspelled word **swimingpool* should be rewritten from **swiming* to *swimming* and also split into *swimming pool*. The joint analysis corrects the error successfully, while the baseline fails to split and the pipeline fails to correct the spelling error. Previous studies, as mentioned in Section 2, deal with the spelling error types shown in examples (1) to (5) in Figure 3, but our work widens the scope of spelling error types to *inflection* and *derivation* errors as shown in the examples in (6) and (7) in Figure 3, since ESL writing contains a number of inflection and derivation errors, as shown in Table 3. In addition, hyphenated words (e.g. **fourty-five/forty-five*) are also corrected by the joint model.

Second, we find several errors, where POS tagging and spelling correction fail. In many error cases, incorrect POS tagging is due to a failure in spelling error correction. In other words, when misspelled words are corrected successfully, the result of POS tagging is also correct. Therefore, we analyse errors in cases of failed spelling correction.

With regard to false positives, when our model could not correct spelling errors in the experiment, we found two main patterns. First, the joint model (SP-JOINT) suggests different words for typographical errors, while the baseline (SP-BASELINE) also tends to fail to correct spelling errors. For example, Figures 4(1) and 4(2) show the failures in typographical error correction. In (1), the misspelled word **beginers* is corrected to *beginner* instead of *beginners*. In the same manner, **concer* in 4(2) is changed to *cancer*. For this pattern, both the baseline and the joint model are able to detect typographical spelling errors, although they fail to suggest correct words. These errors are difficult to correct, because we need information about the broader context or semantics information that sometimes goes beyond the sentence level. Second, our joint model changed correct words into different words. The examples seen in Figures 4(3) and 4(4) show that the proposed model rewrites correct words into different words. In Figure 4(3), the correct word *fell* is rewritten

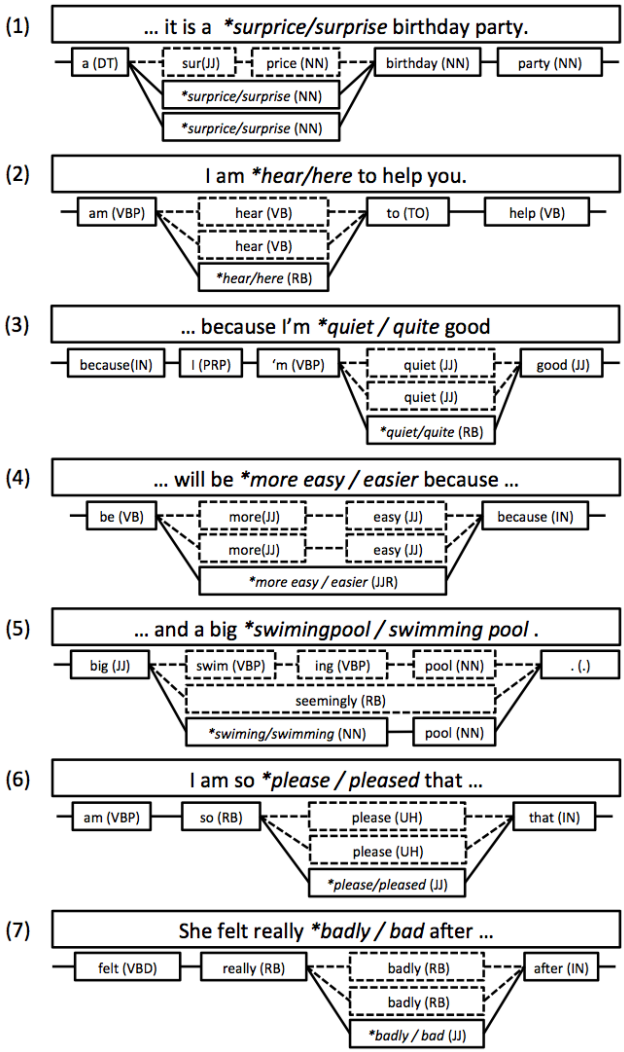


Figure 3: Examples of true positives for POS tagging and spelling correction. Branched nodes represent the output of POS-BASELINE, POS-PIPELINE and POS-JOINT models respectively. Paths and nodes are dotted when they are incorrect. (1) is an example of *typographical error*, (2) is *homophone error*, (3) is *confusion error*, (4) is *split error*, (5) is *merge error*, (6) is *inflection error*, and (7) is *derivation error*.

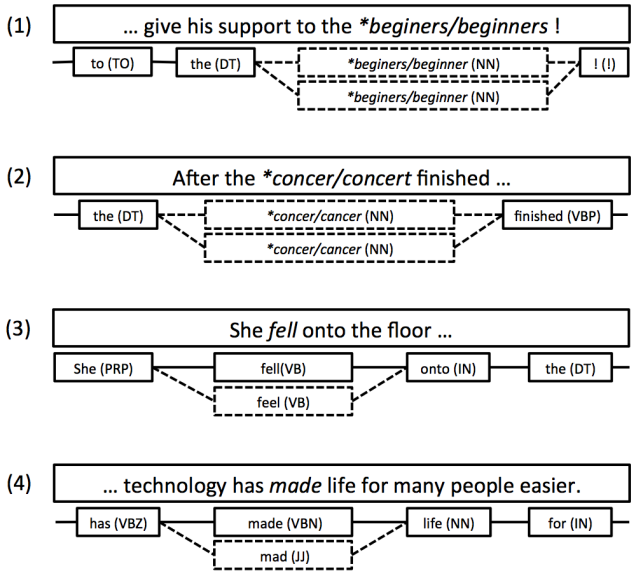


Figure 4: Examples of false positives for spelling correction. Branched nodes represent the output of SP-BASELINE and SP-JOINT models respectively. Edges and nodes are dotted when they are incorrect.

as *feel* and *made* is changed to *mad*. These false positives may be caused by insufficient feature templates and/or data sparseness (overfitting), and we need to deal with them in further research. Of course, both in (3) and (4), this type of wrong corrections does not occur in the baseline, because baseline concerns only typographical errors and does nothing for other types of spelling errors. Since the joint model can detect and correct a wider range of spelling errors, as shown in Figure 3, it causes more false positives, resulting in a lower precision than the baseline. We also find some false positives where the corrected words are also acceptable but regarded as false positives due to the gold standard. Examples of these are British spellings such as *color/colour*, and some adverbs (e.g. *first/firstly*). If we can deal with these cases, the precision will increase.

As shown in Figure 5, we find several examples of false negatives where the system cannot detect spelling errors. In the false negatives, most errors belong to confusion or derivation types, whereas some errors are also found in split and inflection types, indicating that when words before correction are existing words they are hard to detect. For example, Figure 5(1) shows that a misspelled *main* is not detected as an error by the joint model. The error in Figure 5(2) "**After words/Afterwards*" is not corrected, since this error contains a combination of split and typographical errors. With regard to inflection and derivation errors, as Figures 5(3) and 5(4) show, some errors are hard to detect, because the POS sequence before correction is also acceptable to some extent. In order to reduce false negatives, and also false positives, more contextual information will be needed.

Finally, we find that there are some annotation errors in the CLC FCE dataset. For instance, **ab-*

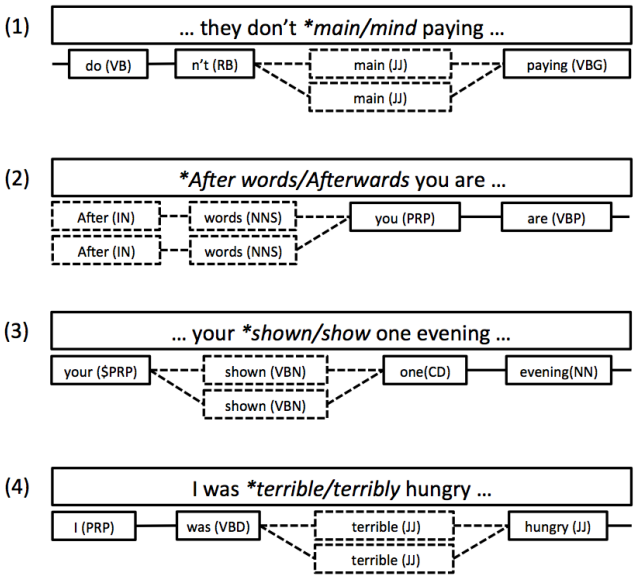


Figure 5: Examples of false negatives for spelling correction. Branched nodes represent the output of SP-BASELINE and SP-JOINT models respectively. Paths and nodes are dotted when they are incorrect.

solutly is corrected to **abolutely* instead of *absolutely*, and **dissapointing* is corrected to **diappointing* instead of *disappointing*. This may force precision downward, though perhaps not to such a great extent.

6 Conclusion

We have presented a joint model of POS tagging and spelling error correction for ESL writing. The model is a CRF-based morphological analysis with word boundary disambiguation. Because the model deals with word boundary ambiguities, it can detect and correctly split and merge errors. In addition, we add misspelled words and their correct/candidate forms into the lexicon, so that the model can deal with not only typographical errors but also a wider range of spelling errors such as homophone, confusion, split, merge, inflection, and derivation errors that often appear in ESL learners' corpora. Our model shows statistically significant improvements in POS tagging and spelling correction, achieving 2.1% and 3.8% of F-value improvements for POS tagging and 5.0% of F-value improvement for spelling error correction compared to the baseline. We have also showed that the joint model improves F-values more than the pipeline model, which is statistically significant.

Acknowledgments

We would like to thank anonymous reviewers for their valuable and very helpful comments.

References

Baba, Y. and Suzuki, H. (2012). How Are Spelling Errors Generated and Corrected? A Study of Corrected and Uncorrected Spelling Errors Using Keystroke Logs. *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics*, pages 373–377.

Bao, Z., Kimelfeld, B., and Li, Y. (2011). A Graph Approach to Spelling Correction in Domain-Centric Search. *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics*, pages 905–914.

Brants, T. and Franz, A. (2006). Web 1T 5-gram Corpus version 1.1. *Technical report, Google Research*.

Brill, E. and Moore, R. C. (2000). An Improved Error Model for Noisy Channel Spelling Correction. *Proceedings of the 38th Annual Meeting on Association for Computational Linguistics*, pages 286–293.

Brockett, C., Dolan, W. B., and Gamon, M. (2006). Correcting ESL Errors Using Phrasal SMT Techniques. *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 249–256.

Chen, Q., Li, M., and Zhou, M. (2007). Improving Query Spelling Correction Using Web Search Results. *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 181–189.

Dahlmeier, D. and Ng, H. T. (2011). Grammatical Error Correction with Alternating Structure Optimization. *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics*, pages 915–923.

Dahlmeier, D. and Ng, H. T. (2012). A Beam-Search Decoder for Grammatical Error Correction. *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 568–578.

Dale, R., Anisimoff, I., and Narroway, G. (2012). HOO 2012 : A Report on the Preposition and Determiner Error Correction Shared Task. *The 7th Workshop on the Innovative Use of NLP for Building Educational Applications*, pages 54–62.

Felice, R. D. and Pulman, S. G. (2008). A Classifier-based Approach to Preposition and Determiner Error Correction in L2 English. *Proceedings of the 22nd International Conference on Computational Linguistics (Coling 2008)*, pages 169–176.

Gamon, M. (2010). Using Mostly Native Data to Correct Errors in Learners' Writing: a Meta-Classifier Approach. *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 163–171.

Gao, J., Li, X., Micol, D., Quirk, C., and Sun, X. (2010). A Large Scale Ranker-based System for Search Query Spelling Correction. *Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010)*, pages 358–366.

Goldberg, Y. and Tsarfaty, R. (2008). A Single Generative Model for Joint Morphological Segmentation and Syntactic Parsing. *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics*, pages 371–379.

Golding, A. R. and Roth, D. (1999). A Winnow-Based Approach to Context-Sensitive Spelling Correction. *Machine Learning*, 34:107–130.

Han, N.-R., Chodorow, M., and Leacock, C. (2006). Detecting Errors in English Article Usage by Non-Native Speakers. *Natural Language Engineering*, 12(02):115–129.

Hatori, J., Matsuzaki, T., Miyao, Y., and Tsujii, J. (2012). Incremental Joint Approach to Word Segmentation, POS Tagging, and Dependency Parsing in Chinese. *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics*, pages 1045–1053.

Islam, A. and Inkpen, D. (2009). Real-Word Spelling Correction using Google Web 1T 3-grams. *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 1241–1249.

Izumi, E., Uchimoto, K., Saiga, T., Supnithi, T., and Isahara, H. (2003). Automatic Error Detection in the Japanese Learners' English Spoken Data. *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics*, pages 145–148.

Kudo, T., Yamamoto, K., and Matsumoto, Y. (2004). Applying Conditional Random Fields to Japanese Morphological Analysis. *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, pages 230–237.

Kukich, K. (1992). Techniques for Automatically Correcting Words in Text. *ACM Computing Surveys*, 24(4):377–439.

Lafferty, J., McCallum, A., and Pereira, F. (1999). Conditional Random Fields : Probabilistic Models for Segmenting and Labeling Sequence Data. *Proceedings of the 18th International Conference on Machine Learning 2001 (ICML 2001)*, pages 282–289.

Lee, J. and Seneff, S. (2008). Correcting Misuse of Verb Forms. *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics*, pages 174–182.

Nagata, R., Whittaker, E., and Sheinman, V. (2011). Creating a Manually Error-tagged and Shallow-parsed Learner Corpus. *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics*, pages 1210–1219.

Park, Y. A. and Levy, R. (2011). Automated Whole Sentence Grammar Correction Using a Noisy Channel Model. *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics*, pages 934–944.

Rozovskaya, A. and Roth, D. (2011). Algorithm Selection and Model Adaptation for ESL Correction Tasks. *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics*, pages 924–933.

Sun, X., Gao, J., Micol, D., and Quirk, C. (2010). Learning Phrase-Based Spelling Error Models from Clickthrough Data. *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 266–274.

Surdeanu, M., Johansson, R., Meyers, A., Màrquez, L., and Nivre, J. (2008). The CoNLL-2008 Shared Task on Joint Parsing of Syntactic and Semantic Dependencies. *Proceedings of the 12th Conference on Computational Natural Language Learning*, pages 159–177.

Tajiri, T., Komachi, M., and Matsumoto, Y. (2012). Tense and Aspect Error Correction for ESL Learners Using Global Context. *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 198–202.

Toutanova, K. and Moore, R. C. (2002). Pronunciation Modeling for Improved Spelling Correction. *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 144–151.

Yannakoudakis, H., Briscoe, T., and Medlock, B. (2011). A New Dataset and Method for Automatically Grading ESOL Texts. *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics*, pages 180–189.

Zhang, Y. and Clark, S. (2010). A Fast Decoder for Joint Word Segmentation and POS -Tagging Using a Single Discriminative Model. *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 843–852.

Automatic Detection of Psychological Distress Indicators and Severity Assessment from Online Forum Posts

Shirin Saleem[1], Rohit Prasad[1], Shiv Vitaladevuni[1], Maciej Pacula[1], Michael Crystal[1], Brian Marx[2,3], Denise Sloan[2,3], Jennifer Vasterling[2,3] and Theodore Speroff[4,5]

(1) Raytheon BBN Technologies, 10 Moulton St, Cambridge, MA, U.S.A.
(2) National Center for PTSD at VA Boston Healthcare Sytem, Boston, MA, U.S.A.
(3) Boston University School of Medicine, Boston, MA, U.S.A.
(4) VA Tennessee Valley Healthcare System, Nashville, TN, U.S.A.
(5) Vanderbilt University School of Medicine, Nashville, TN, U.S.A.

{ssaleem,rprasad,svitalad,mpacula,mcrystal}@bbn.com
{brian.marx,denise.sloan,jennifer.vasterling}@va.gov
ted.speroff@vanderbilt.edu

ABSTRACT

Psychological disorders are frequently under-diagnosed and consequently have an irreversible impact on individuals and society. The stigma associated with such disorders makes face-to-face discussions with family members and clinicians difficult for many individuals. In contrast, people openly relate experiences on Internet forums. This paper describes a novel system that analyses forum posts to: (1) detect distress indicators that directly map to the Diagnostic and Statistical Manual of Mental Disorders (DSM) IV constructs, and (2) assess the severity of distress for prioritizing individuals who should seek clinical help (i.e. triage). For distress indicator detection, we use support vector machines (SVMs) trained on a suite of innovative intra- and inter-message features. We show significant improvements in multi-label classification accuracy using human-generated rationales in support of annotated distress labels. For triage assessment, we demonstrate the effectiveness of Markov Logic Networks (MLNs) in dealing with noisy distress label detections and encoding expert rules.

KEYWORDS: Psychological Distress, Web forums, Text classification, Annotator rationales, Support Vector Machines, Probabilistic Logic, Markov Logic Networks.

Proceedings of COLING 2012: Technical Papers, pages 2375–2388,
COLING 2012, Mumbai, December 2012.

1 Introduction

Psychological health disorders pose a growing threat to individuals, their family members and to society. Disorders such as Depression, Post-Traumatic Stress Disorder (PTSD), and mild Traumatic Brain Injury (mTBI), are often under-diagnosed and under-treated (Kessler et. al, 1999). Failure to intervene early and effectively impacts individuals and their family members adversely and results in profound long-term costs to society.

The standard approach to diagnosing psychological health disorders is through a series of clinically administered diagnostic interviews and tests (Weathers et. al, 2001). However, assessment of patients using these tests is expensive and time-consuming. Furthermore, the stigma associated with mental illnesses motivates inaccurate self-reporting by affected individuals and their family members, thus making the tests unreliable.

In recent years, there has been a tremendous growth in social interactions on the Internet via social networking sites and online discussion forums. In contrast to clinical tests, the Internet is an ideal, anonymous medium for distressed individuals to relate their experiences, seek knowledge, and reach out for help. Web-forum discussions of symptoms, thoughts and experiences are open, descriptive, and honest, making them an ideal source for observing communications of individuals for assessing psychological status.

In this paper, we present a multi-stage text classification system for assessing psychological status of individuals based on their text postings on online web forums. Specifically, our system combines state-of-the-art NLP and machine learning techniques to: (1) extract fine-grained psychological distress indicators/labels derived from Diagnostic and Statistical Manual of Mental Disorders (DSM) IV (American Psychiatric Association, 2000), and (2) assesses the severity of distress that can be used to triage individuals who should seek clinical help.

The same factors that make web-forum data interesting for observing psychological distress also make automated analysis extremely challenging. For instance, the language used in such forums is highly informal, with ill-formed, grammatically incorrect sentences, misspellings, and special character sequences such as emoticons. Vague references to emotional states, description of present vs. past traumatic experiences, and relating one's own versus other's experience all pose novel challenges to natural language processing (NLP). Additionally, any approach for psychological health analysis of text interactions must incorporate domain knowledge from expert psychologists and clinicians. Together these challenges make this domain a fascinating research area with the potential for research advances to revolutionize psychological healthcare.

1.1 Previous Work

Existing applications for automatic detection of psychological disorders have been limited to structured questionnaires and formal clinical records (Brown, et. al. 2006). In contrast, our work is focused on noisy, informal text messages from Web-forums. Text classification research on such data has primarily focused on identifying social roles in scientific forums (Wang, et. al, 2011) and sentiment analysis (Abbasi et. al, 2008). To the best of our knowledge, the work presented in this paper for assessing psychological status from web-forum text is first of its kind.

Several rule-based approaches have been explored for detecting PTSD and mTBI from clinical narratives (Elkin et. al, 2010) (Trusko et. al, 2010). However, these approaches rely on annotating individual words as positive, negative, or neutral indicators of the condition. Such annotation is

laborious, lacks consistency, and requires deep subject matter expertise. Instead, our approach uses statistical models that do not require such laborious annotation and encode domain knowledge by learning weights for the domain rules from data.

1.2 Novel Contributions

We present several novel techniques within a multi-stage text classification framework for assessing psychological status from informal text posted on Web-forums. First, we describe a suite of features and classifiers trained on expert-annotated text to detect distress indicators. The training data itself is a first of its kind, where each message has been annotated by psychologists using a codebook of 136 distress labels that directly map to DSM-IV constructs. Since messages are often tagged with multiple distress indicators, the detection task is a multi-label classification problem with a large set of labels. Additionally, a fraction of our data is annotated with rationales that support distress labels. We show that these rationales can be effectively used to improve multi-label classification accuracy. Specifically, we observe a relative improvement of 14.6% over using plain text features. Another key contribution of this work is the use of probabilistic logic, namely Markov Logic Networks (MLNs) (Richardson and Domingos, 2006) to incorporate domain-specific rules, and handle the inherent noise in the data. We show that MLNs improve the triage classification accuracy, and provide a robust approach for inferring triage codes from noisy distress label detections as well as potentially contradictory domain rules.

2 Corpus for Experimentation

Our corpus consists of threads downloaded from an online forum for veterans with post-combat psychological issues. The forum fosters anonymous discussions between returning military personnel with PTSD or suspected of PTSD, and their caregivers. Note that we do not identify any individuals from their posted text nor do we trace any distress signals to a specific poster.

In consultation with psychologists, a codebook of 136 psychological distress labels spanning PTSD, mTBI, and depression symptoms was developed. Codes/labels were mostly derived from the DSM-IV guidelines (American Psychiatric Association, 2000). The labels were organized into five broad categories: Stress Exposure (e.g., Combat Exposure, Traumatic Loss, Captivity), Affect (e.g., Anger/Rage/Frustration/Contempt, Fear, Worthlessness), Behaviour (e.g., Social Isolation, Sleep problems, Excessive Drug Use), Cognition (e.g., Intrusive Thoughts and Memories, Homicide Ideation, Posttraumatic Amnesia), and Domains of Impairment (e.g., Legal Problems, Financial Problems, Occupational Impairment). In the annotation process, each message is first tagged to indicate if a message is relevant to assessing the author's psychological state. Each relevant message is then annotated with one or more labels from the codebook characterizing the psychological state of the author in accordance with the message content. Additionally, for a subset of messages, we highlighted contextual rationales to support the distress labels annotations. Figure 1 shows a snapshot of the distress labels and their hierarchy.

Expert psychologists next annotated each author in a thread with a triage code that indicates treatment acuity or the priority assigned to a referral for additional treatment. We used three triage codes in our annotation – TR1 indicating current or imminent danger to self or others; TR2 indicating behavioural disturbances, distress, functional impairment and/or suicidal/homicidal ideation without any imminent danger to self or others; and TR3 where there is no evidence of current behavioural disturbance, distress or functional impairment. For each of these triage codes,

the treatment acuity varies from emergency intervention or urgent care evaluation for TR1 to non-urgent treatment referral for TR2 to no recommendation for treatment for TR3. Since online forums are moderated and expunged of sensitive content, we rarely observed any occurrences of TR1 in the forum posts. Our focus in this paper is hence restricted to distinguishing between codes TR2 and TR3. However, our approach is extensible to the detection of TR1 if appropriate training data were available.

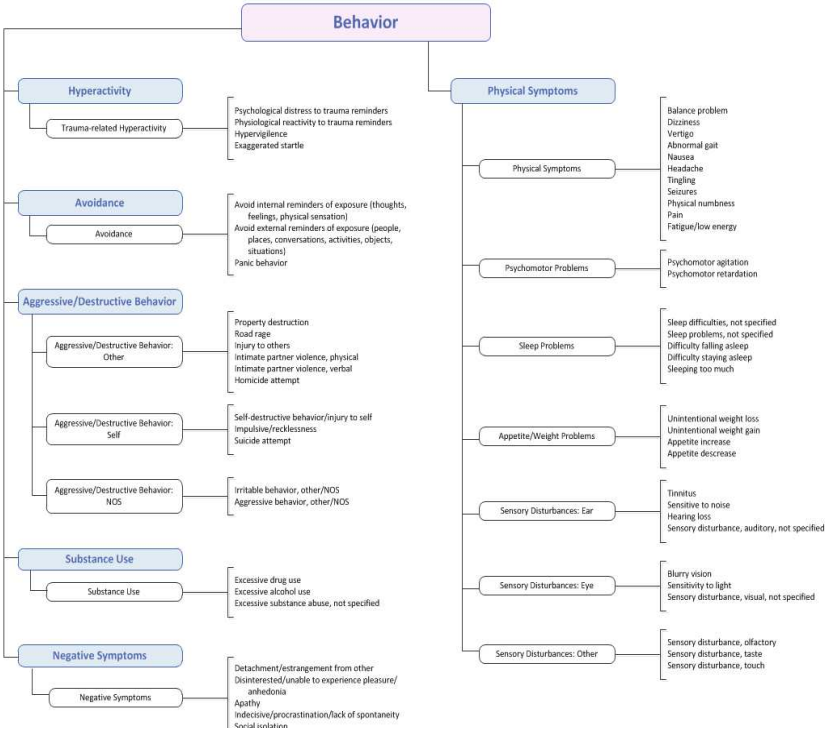


Figure 1: Snapshot of codebook of distress labels and their hierarchy.

3 Approach Overview

Figure 2 gives an overview schematic of our approach. We use a trainable multi-stage text-classification system to detect distress indicators from text interactions on Web forums and severity of distress of an author for prioritizing need for clinical care. Our system analyses the text posted by an author to first determine if it is relevant for psychological distress. If relevant, the text is further processed using multi-label classification to estimate fine-grained

psychological distress indicators. Next, information from the text and the detected distress labels is combined using domain-specific rules to estimate priority for intervention. In what follows, we describe the details for fine-grained distress detection and severity assessment.

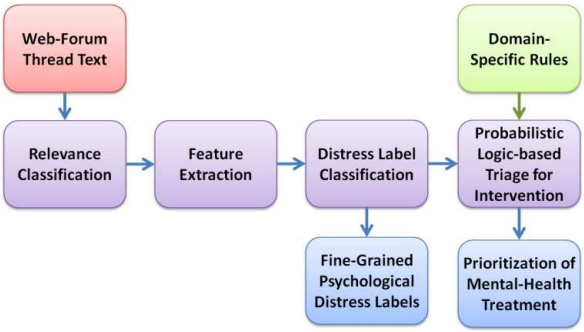


Figure 2: Schematic of Approach to Estimate Psychological Distress Labels and Prioritization of Mental-Health Intervention from Web-Forum Text.

4 Multi-label Distress Classification

4.1 Classifier

Algorithms for multi-label classification, the task of assigning one or more labels to an instance, can be grouped into two main categories: (1) problem transformation methods, and (2) algorithm adaptation methods (Tsoumakas et al. 2011). Problem transformation methods transform the multi-label classification problem into many single-label classification problems. Algorithm adaptation methods extend specific learning algorithms in order to handle multi-label data directly. Given the large size of our label set (118 observed labels out of 136 total), we could not find a memory-efficient way to use many of the algorithm adaptation methods. Instead, we focused on problem transformation methods using binary one-versus-all Support Vector Machines (SVMs) that detect the presence or absence of each of the fine-grained distress labels.

4.2 Features

Most systems for text classification represent documents as a bag-of-words. While this approach works well for most tasks with adequate training data, it does not capture any semantic correlations or higher order information between words. In our experiments, we explored a variety of features that look beyond the identity of the words in the message. These include message-level features computed based on the content of individual messages as well as thread-level features that exploit the structure of the discussion thread and look at other messages in the thread. In all cases, the features are binary, integer, or real valued and contain no Personally Identifiably Information (PII).

A1: Unigrams – We extracted unigrams from the forum messages by first removing stop words. Next, we apply Porter stemming to remove the common morphological and inflectional endings in English. Emoticons such as smileys were retained and used as features.

A2: Pronoun Count – Pronouns are typically discarded in most text classification applications in the pre-processing stage under the assumption that they occur too frequently to bear any information. However, in (Campbell and Pennebaker, 2003) it was shown that changes in the way people use pronouns when writing about traumatic experiences is a powerful predictor of changes in physician visits or an indicator of their general health. We hence included the normalized pronoun count as a feature.

A3: Punctuation Count – Normalized count of punctuations in the message calculated as the percentage of tokens/words in the message that are punctuations.

A4: Average Sentence Length - Average number of words in the message sentences, where sentence segmentation was determined based on punctuations and line breaks.

A5: Sentiment Words - Sentiment bearing words are correlated with specific distress labels (especially in the Affect category of distress labels). Identifying and grouping such words in a message could positively influence the classification performance of these labels. We extracted 125 binary features indicating the presence or absence of sentiment bearing words in the message. These words were selected from two sources: 68 lexicons form the Linguistic Inquiry and Word Count (LIWC) (Pennebaker et. al, 2007), and 57 lexicons from the General Inquirer (GI) system (Stone, 1966). The LIWC includes categories corresponding to affective and emotional processes (e.g.: positive/negative emotions), Cognitive Processes (e.g.: causation) and Social Processes (e.g.: friends) among others. The GI System includes valence categories (positive, negative) and motivation related words.

A6: Lead Author Post -Binary feature indicating whether the message was posted by the author who started the thread.

A7: First Responder Post -Binary feature indicating whether the message was posted by the author who first responded to the lead message of the thread.

A8: Thread Similarity - Real-valued feature that measures the average cosine similarity of the words in the message to the other messages in the thread.

A9: First Message Similarity - Real-valued feature that measures the cosine similarity of the words in the message to the words in the first message posted in the thread.

A10: Domain Phrases Derived from Rationales – (Zaidan et. al, 2008) showed improved performance in a sentiment classification task using annotator rationales within a contrastive learning framework of an SVM. Here, we use the rationales by extracting label-specific textual features from them. For every label, we first find the most frequent n-grams (n <= 5) in the highlighted rationales. We then filtered n-grams that had a high overlap ratio with other labels and also those that consisted solely of words in a pre-defined stop word list. The resulting n-grams were then used as binary features for classification. Examples of such phrases for the label Suicidal Ideation include: “thought about jumping”, “me suicidal”, “end their life”, “feel like killing myself”.

5 Psychological Triage Models for Severity Assessment

Our goal is to find authors who might require treatment or medical evaluation based on any behavioural disturbances, distress, functional impairments and/or suicidal or homicidal ideation. We explored two approaches to address this problem. The first approach uses an SVM trained on the words and predicted distress labels for the messages posted by the author. Our second approach uses Markov Logic Networks (MLNs) (Richardson and Domingos, 2006) to encode domain knowledge using probabilistic first order rules with associated weights.

In our system, the MLN computes the probability of a triage code using: (1) the distribution of words in the messages posted by an author, (2) the predicted distress labels, and (3) domain-specific rules that encode dependencies between the text, distress labels and the triage. The domain-specific rules were derived from existing diagnostic criteria as follows:

1. Rules derived from Primary Care-PTSD (PC-PTSD) screening test (Prins et al. 2003) used routinely in the VA to screen for PTSD. It comprises of 4 questions which map to 10 distress labels from the codebook.
2. Rules derived from DSM-IV guidelines for PTSD. These comprise of 4 criteria consisting of questions that map to distress labels in the codebook. For example, a criterion encoded as a rule in the MLN is the presence of one or more of the trauma exposure labels and one or more of the fear/helpless labels.

hasSymptom(Helplessness, p) OR hasSymptom(Fear, p) OR hasSymptom(Horror,p) => triageCode(+t, p)
hasSymptom(Intimate family impairment, p) OR hasSymptom(Extended family impairment, p) OR hasSymptom(Friendship impairment, p) OR hasSymptom(Social impairment, p) OR hasSymptom(Occupational impairment, p) OR hasSymptom(Educational impairment, p) OR hasSymptom(Self-care impairment, p) OR hasSymptom(Financial problems, p) OR hasSymptom(Legal problems, p) => triageCode(+t, p)
hasSymptom(Sleep problems, p) OR hasSymptom(Difficulty falling asleep, p) OR hasSymptom(Anger, p) OR hasSymptom(Road rage, p) OR hasSymptom(Property destruction, p) OR hasSymptom(Concentration problem, p) OR hasSymptom(Hypervigilence, p) OR hasSymptom(Exaggerated startle, p) => triageCode(+t, p)
hasSymptom(Intrusive thoughts, p) OR hasSymptom(Nightmares, p) OR hasSymptom(Reliving event, p) OR hasSymptom(Psychological distress to trauma reminders, p) OR hasSymptom(Physiological reactivity to trauma reminders, p) => triageCode(+t, p)
hasSymptom(Nightmares, p) OR hasSymptom(Reliving event, p) OR hasSymptom(Intrusive thoughts and memories of events, p) => criterion1(True, p)

Table 1: Examples of domain-specific rules derived from DSM-IV guidelines and PC-PTSD screening tests. Here, p is variable ranging over authors of messages; and t ranges over triage codes.

MLNs have two key advantages for our application. First, the use of statistical inference provides robustness to noise in the text and label predictions, and potential contradictions in the domain-specific rules. Second, the relative weights for the domain-specific rules can be automatically learned from the training data.

We employed Alchemy, an implementation of learning and inference algorithms for MLNs, (Richardson and Domingos, 2006) for our experiments. To learn the weights of the domain-specific rules, we used discriminative training, which maximizes the conditional likelihood of target labels (in our case the triage codes) given the observed variables (in our case the message words and distress labels). Alchemy uses an approach referred to as pre-conditioner scaled conjugate gradient for discriminative weight learning (Lowd and Domingos, 2007). The inference is performed using MaxWalkSAT; see (Richardson and Domingos, 2006) for details. Table 1 shows examples of domain-specific rules incorporated in the MLN based on DSM-IV guidelines and PC-PTSD screening test.

6 Experimental Results

6.1 Inter-annotator Agreement

We performed an inter-annotator agreement study for both distress label classification and triage annotation. Annotation for distress labels was performed by four Subject Matter Experts (SMEs). We measured inter-annotator agreement among multiple annotators using the Fleiss Kappa statistic (Fleiss, 1971). In order to compute the overall Kappa for the distress labels, we first computed the Fleiss Kappa for each label, and then performed a weighted combination of these scores. We observed a Kappa of 0.68 for the "Relevant" tag and 0.59 for the "Distress Labels" on a set of 9 threads comprising 126 messages that were annotated by all four SMEs. In general, a Kappa of 0.41-0.60 suggests moderate agreement, and 0.61 to 0.80 suggests good agreement (Landis and Koch, 1977). We found that the inter-annotator agreement, i.e. the Kappa values, for the individual distress labels spanned a wide range. Some of the distress labels had very good agreement, e.g., Sleep problems, and Alcohol abuse, possibly because the messages contained extensive descriptions of the distress conditions. The labels that were in poor agreement were typically those that required inference and world knowledge, e.g., Despair and Worthlessness. We will further investigate this inter-annotator agreement disparity as part of future work. Annotation for the triage classification was performed by six SMEs. We again measure the Fleiss Kappa statistic for triage codes assigned to 43 authors across 10 threads. We found this value to be 0.71, indicating good agreement.

6.2 Multi-label Distress Classification

We chose a set of 512 threads, comprising of 5000 relevant and irrelevant messages, for our multi-label distress classification experiments. We held out 90 threads for testing, and used the remaining for the training set. We collected rationales for 650 messages in training. The SVM parameters were tuned based on 10-fold cross validation on the training set where threads were randomly distributed across 10 different subsets. Performance is reported on the held-out test set. Table 2 shows the data statistics of the experimental corpus.

Category		Train	Test
Threads		422	90
Authors		1166	260
Relevant	Messages	1868	440
	Total Words	397K	92K
	Unique Labels	118	97
	Average Number of Labels per message	2.8	2.9

Table 2: Corpus setup for Multi-Label Distress Classification

As described in Figure 2, we approached the problem of automatically detecting psychological distress indicators in forum posts in two stages. We first applied a classifier to filter out messages that have no bearing on the detection of psychological distress. Irrelevant messages include cases such as when authors choose to post very short messages that do not have any information bearing content, like a simple "Thank you", and when the topic of discussion digresses to sub-topics or tangential topics. In order to identify relevant versus irrelevant messages, we trained an SVM on the annotated forum messages, and used it to automatically recognize relevant messages in the test set. We then applied multi-label classifiers to predict one or more distress labels described by the author on the relevant messages. In this paper, we focus on this second stage of text classification, and report closed-set results on messages that we know are relevant.

Classification performance is measured by computing the mean of the Area Under the Curve (AUC) for all labels. The AUC for each label is computed on a Receiver Operating Characteristic (ROC) curve with the false acceptance rate (FAR) bounded at 10%, and normalized such that the maximum possible AUC is 1.We also report the overall AUC number for the entire ROC curve, i.e. FAR of 100%. The labels detected for all messages posted by the same author within a thread were pooled for evaluation. For our experiments with SVMs, we used the Weka machine learning software (Hall et. al, 2009) with the Radial Basis Function (RBF) kernel. We performed grid-search to find the best regularization (C) and gamma (g) parameters on the cross-validation set. For the baseline experiment with SVMs, each message was treated as a bag of words with normalized (TF-IDF) frequencies. Next, the remaining features described in section 4.1 were incrementally added to the baseline feature set of the SVM classifier. Table 3 shows the performance of the SVM with the unigram TF-IDF features as well as the improvements from adding the other features. For a random classifier, the mean-AUC bounded up to False Acceptance Rate of 10% is 0.05, and the overall AUC is 0.5. No significant change in performance is seen with the incremental addition of the message level features A2-A5 and thread level features A6-A9. We retained these features since their addition did not explicitly hurt performance. Overall, the mean-AUC improves by 14.6% relative using the full set of features in section 4.1 over just the unigram words (Table 3). We see a large gain from the addition of the domain phrase features derived from rationales (A10).

We found that our approach of using the rationales by extracting label specific domain phrase features out-performed the contrastive approach in (Zaiden et. al, 2008). The latter gave a

bounded mean-AUC of 22.3, whereas our feature-based approach yielded 23.5 when added to the unigram feature set.

Feature Set	Mean AUC Bounded for 0-10% False Accept Rate	AUC (Overall)
A1	0.213	0.6757
A1, A2, A3, A4, A5	0.211	0.6699
A1, A2, A3, A4, A5, A6, A7, A8, A9	0.212	0.6699
A1, A2, A3, A4, A5, A6, A7, A8, A9, A10	0.244	0.6874

Table 3: Multi-label distress classification results with different feature sets

It is to be noted that the dataset has a high class imbalance. The most frequently occurring label – Anger/Rage/Frustration/Contempt has 698 training examples whereas half of the labels have less than 20 examples in training. Hence, a large number of labels perform poorly merely due to the lack of sufficient training data. In Figure 3 we also show the AUCs for all the labels. Approximately half the labels have an AUC < 0.2. The maximum value of individual AUC was found to be 0.884 for Excessive Substance Use. The top 5 labels with maximum AUC are Excessive Substance Use, Panic behavior, Nightmares or Unpleasant Dreams, Concentration Problems and Child Maltreatment. In all of these labels, there is extensive description of the distress condition in the messages. In contrast, there are many labels that are implied in the text, and are inconsistently inferred even amongst human annotators. We demonstrated this in the inter-annotator agreement study where we found only moderate agreement between annotators in the coding of these distress labels.

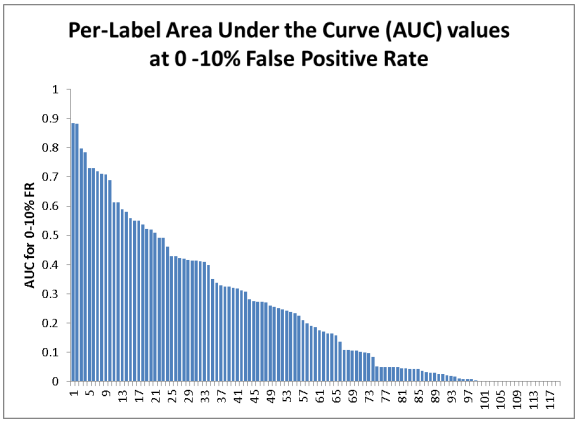


Figure 3: Per-label AUC values for false positive rate capped at 10%. The AUC is normalized such that the maximum possible value is 1.0.

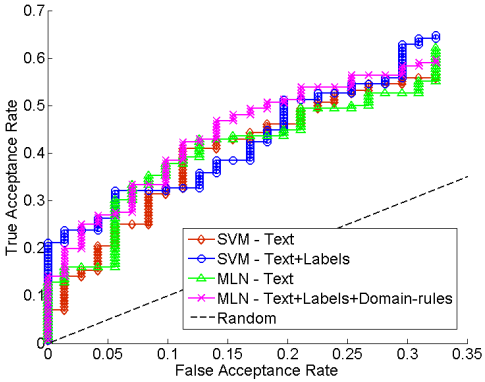


Figure 4: ROC curves for triage classification for SVM and MLN for Text and Text + Distress Labels. MLN with Text and Distress Labels combined using domain specific rules gives best results.

6.3 Triage Assessment

A subset of authors in the downloaded forum threads were tagged with triage codes, specifically 907 authors out of 1426. We used 680 of these for training the triage models, and 227 for evaluation. We compare the triage classification performance for SVM and MLN using ROC curves, i.e. rate of correct acceptance of TR2 versus false acceptance of TR3. The performance was measured using area under the curve (AUC). We capped the ROC curves to false acceptance rates less than 33% based on the fact that high false acceptance rates make the triage impractical for our application. The AUC is normalized such that the maximum possible AUC is 1. The AUC of a random/chance classifier is 0.165. Table 4 presents the AUC values for SVM and MLN for different types of inputs. As can be observed, MLNs provide statistically significant gains over SVMs by using domain-specific rules for combining information from text as well as the distress label detections. Figure 4 shows the ROC curves for the triage classification.

Method	**Area Under the Curve (AUC) with Bounded False Accept Rate of 33%**
SVM - Text	0.4090
SVM – Text + Distress Labels	0.4354
MLN – Text	0.4148
MLN – Text + Distress Labels + DSM-IV and PC-PTSD Rules	**0.4515**

Table 4: Triage classification performance AUC for ROC curves capped at false acceptance rate less than 33%. The AUC is normalized such that maximum possible value is 1.0

7. Conclusions and Future Work

In this paper, we introduced a powerful system that automatically detects psychological distress indicators from text in online forum posts, and demonstrated it in a novel domain of unconstrained web-forums. We presented multi-label classification for 136 labels of fine-grained psychological distress conditions on extremely challenging unstructured text data, and a novel approach based on probabilistic logic to employ domain-specific rules for combining information from text features and the distress label detections. We also showed that incorporating rationales from domain experts for the label annotations helps improve the multi-labeling performance, and presented a novel feature to exploit the rationale annotations.

In the future, we intend to investigate methods that exploit label dependencies. We will also investigate contextual features for classification that exploit information from previous messages within a thread. Finally, we plan to validate the system on text data from subjects diagnosed with PTSD and compare the outcomes on a control group that does not suffer from PTSD.

Acknowledgments

This paper is based upon work supported by the DARPA DCAPS Program. The views expressed here are those of the author(s) and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

References

A. Abbasi, H. Chen, and A. Salem. 2008. Sentiment Analysis in Multiple Languages: Feature Selection for Opinion Classification in Web Forums. ACM Transactions on Information Systems, 26(3), no. 12.

American Psychiatric Association. 2000. Diagnostic and statistical manual of mental disorders (4th ed., text rev.). Washington, D

S. H. Brown, et al. 2006. eQuality: Electronic Quality Assessment from Narrative Clinical Reports. Mayo Clinic Proceedings, vol. 81, pp. 1472-1481.

R. S. Campbell and J. W. Pennebaker. 2003. The secret life of pronouns: Flexibility in writing style and physical health. Psychological Science, 14, 60-65.

P. L. Elkin, et al. 2010. The Health Archetype Language (HAL-42): Interface considerations. International Journal of Medical Informatics, vol. 79, pp. 71-75.

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

M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann and I. H. Witten. 2009. The WEKA Data Mining Software: An Update. SIGKDD Explorations, Volume 11, Issue 1

R. C. Kessler, et al. 1999. Past-year use of outpatient services for psychiatric problems in the National Comorbidity Survey. American Journal of Psychiatry, 156(1), 115-123.

J. R. Landis and G.G. Koch. 1977. The measurement of observer agreement for categorical data. . Biometrics33 (1): 159–174.

D. Lowd, and P. Domingos, 2007, E*ffi*cient weight learning for Markov logic networks. In PKDD-2007, 200–211.

J. W. Pennebaker, R. J. Booth, M. E. Francis. 2007. Linguistic Inquiry and Word Count (LIWC2007): A text analysis program. Austin, TX: LIWC (www.liwc.net).

A. Prins, P. Ouimette, R. Kimerling, R. P. Cameron, D. S. Hugelshofer, J. Shaw-Hegwer, A. Thrailkill, F. D. Gusman, J. I. Sheikh. 2003. The primary care PTSD screen (PC-PTSD): development and operating characteristics. Primary Care Psychiatry, 9, 9-14

M. Richardson and P. Domingos. 2006. Markov logic networks. Machine Learning 62, 1-2, pp. 107-136.

P. J. Stone. 1966. The General Inquirer: A Computer Approach to Content Analysis. The MIT Press

B. Trusko, et al. 2010. Are Post Traumatic Stress Disorder Mental Health Terms Found in SNOMED-CT Medical Terminology? Journal of Traumatic Stress, vol. 23, pp. 794-801.

G. Tsoumakas, I. Katakis, I. Vlahavas. 2011. Random k-Labelsets for Multi-Label Classification. IEEE Transactions on Knowledge and Data Engineering, IEEE, 23(7), pp. 1079-1089

V. N. Vapnik. 1995. The Nature of Statistical Learning Theory. Springer, New York.

L. Wang, M. Lui, S. N. Kim, J. Nivre and T. Baldwin. 2011. Predicting Thread Discourse Structure over Technical Web Forums. Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing (EMNLP 2011), Edinburgh

F. W. Weathers, T. M. Keane, and J. R. T. Davidson. 2001. Clinician-Administered PTSD Scale: A review of the first ten years of research. Depression and Anxiety, Vol 13(3), 132-156.

O. F. Zaidan, J. Eisner and C. Piatko. 2008. Machine Learning with Annotator Rationales to Reduce Annotation Cost. Proceedings of the NIPS 2008 Workshop on Cost Sensitive Learning

Ant Colony Algorithm for the Unsupervised Word Sense Disambiguation of Texts: Comparison and Evaluation

Didier SCHWAB, Jérôme GOULIAN,
Andon TCHECHMEDJIEV, Hervé BLANCHON
LIG-Laboratory of Informatics of Grenoble
GETALP-Study Group for Machine Translation and Automated Processing of Languages and Speech
Univ. Grenoble-Alpes, France
{Didier.Schwab,Jerome.Goulian,Andon.Tchechmedjiev,Herve.Blanchon}@imag.fr

ABSTRACT
Brute-force word sense disambiguation (WSD) algorithms based on semantic relatedness are really time consuming. We study how to perform WSD faster and better on the span of a text. Several stochastic algorithms can be used to perform Global WSD. We focus here on an Ant Colony Algorithm and compare it to two other methods (Genetic and Simulated Annealing Algorithms) in order to evaluate them on the Semeval 2007 Task 7. A comparison of the algorithms shows that the Ant Colony Algorithm is faster than the two others, and yields better results. Furthermore, the Ant Colony Algorithm coupled with a majority vote strategy reaches the level of the first sense baseline and among other systems evaluated on the same task rivals the lower performing supervised algorithms.

TITLE AND ABSTRACT IN FRENCH

Algorithme à colonie de fourmis pour la désambiguïsation lexicale non supervisée de textes : comparaison et évaluation

Les algorithmes exhaustifs de désambiguïsation lexicale ont une complexité exponentielle et le contexte qu'il est calculatoirement possible d'utiliser s'en trouve réduit. Il ne s'agit donc pas d'une solution viable. Nous étudions comment réaliser de la désambiguïsation lexicale plus rapidement et plus efficacement à l'échelle du texte. Nous nous intéressons ainsi à l'adaptation d'un algorithme à colonies de fourmis et nous le confrontons à d'autres méthodes issues de l'état de l'art, un algorithme génétique et un recuit simulé en les évaluant sur la tâche 7 de Semeval 2007. Une comparaison des algorithmes montre que l'algorithme à colonies de fourmis est plus rapide que les deux autres et obtiens de meilleurs résultats. De plus, cet algorithme, couplé avec un vote majoritaire atteint le niveau de la référence premier sens et rivalise avec les moins bons algorithmes supervisés sur cette tâche.

KEYWORDS: Unsupervised Word Sense Disambiguation, Semantic Relatedness, Ant Colony Algorithms, Stochastic optimization algorithms.

KEYWORDS IN FRENCH: Désambiguïsation lexicale non-supervisée, proximité sémantique, algorithmes à colonies de fourmis, algorithmes stochastiques d'optimisation

Proceedings of COLING 2012: Technical Papers, pages 2389–2404,
COLING 2012, Mumbai, December 2012.

1 Introduction

Word Sense Disambiguation (WSD) is a core problem in Natural Language Processing (NLP), as it may improve many of its applications, such as multilingual information extraction, automatic summarization, or machine translation. More specifically, the aim of WSD is to find the appropriate sense(s) of each word of a text among a pre-definied sense inventory. For example, in "*The mouse is eating cheese.*", for the word ‹*mouse*›, the WSD algorithm should choose the sense that corresponds to the animal rather than the computer device. There exist many methods to perform WSD, among which, one can distinguish between supervised methods and unsupervised methods. The former are based on machine learning techniques that use a set of (manually) labelled training data, whereas the latter do not.

This article focusses on an unsupervised knowledge-based approach for WSD, derived from the method proposed by (Lesk, 1986). This approach uses a similarity measure that corresponds to the number of overlapping words between the definitions of two word senses. With this metric, one can select, for a given word of a text, the sense that yields the highest relatedness to a certain number of its neighbouring words (with a fixed window size). Works such as (Pedersen et al., 2005) use a brute-force (BF) global algorithm that evaluates the relatedness between each word sense and all the senses of the other words within the considered context window. The execution time is exponential in the size of the input, thus reducing the maximum possible width of the window. The problem can become intractable even on the span of short sentences: a linguistically motivated context, such as a paragraph for instance, can not be handled. Thus, such approaches can not be used for applications where real time is a necessary constraint (image retrieval, machine translation, augmented reality).

In order to overcome this problem and to perform WSD faster, we are interested in other methods. In this paper, we focus on three methods that globally propagate a local algorithm based on semantic relatedness to the span of a whole text. We consider two unsupervised algorithms from the state of the art, a Genetic Algorithm (GA) (Gelbukh et al., 2003) and a Simulated Annealing (SA) algorithm (Cowie et al., 1992), as well as an adaptation of an Ant Colony Algorithm (ACA) (Schwab et al., 2011). Our aim is to provide an empirical comparison of the ACA with the two other unsupervised algorithms, using the Semeval-2007, Task-7, Coarse grained corpus (Navigli et al., 2007) (both in terms of quality and execution time). Furthermore, we also evaluate the results after applying a majority vote strategy.

After a brief review of the state-of-the-art of WSD, the algorithms are described. Subsequently, their implementations are discussed, as well as the estimation of the *best* parameters and the evaluation of the tested algorithms. Finally, an analysis of the results is presented as well as a comparison to other systems on Semeval 2007 Task 7. Then, we conclude and propose some perspectives for future work.

2 Brief State of the art of Word Sense Disambiguation

In simple terms, WSD consists in choosing the best sense among all possible senses for all words in a text. There exist many methods to perform WSD. The reader can refer to (Ide and Véronis, 1998) for works before 1998 and (Agirre and Edmonds, 2006) or (Navigli, 2009) for a complete state of the art.

Supervised WSD algorithms are based on the use of a large set of hand-labelled training data to build a classifier which can determine what are the right sense(s) for a given word in a given context. Most classical supervised learning algorithms have been applied to WSD, and

even though they tend to yield better results (on English) over unsupervised approaches, their main disadvantage is that hand-labelled examples are rare and expensive resources (knowledge acquisition bottleneck) that must be created for each sense inventory, each language and even each specialized domain. We share the opinion of (Navigli and Lapata, 2010) that unsupervised methods would be better in order to overcome these obstacles in the short term.

Among unsupervised WSD methods some use raw corpora to build word vectors or co-occurrence graphs while others use external knowledge sources (dictionaries, thesauri, lexical databases, ...). The latter are based on the use of semantic similarity metrics that assign a score representing how *related* or *close* two word senses are. Many such measures exist and can be classified in three main categories: taxonomic distance in a lexical graph; taxonomic distance weighted by information content; feature-based similarity. More recent efforts go towards hybrid measures that combine two or more of the above. Readers may consult (Pedersen et al., 2005), (Cramer et al., 2010), (Tchechmedjiev, 2012) or (Navigli, 2009) for a more complete overview.

Commonly, similarity measures use WordNet (Fellbaum, 1998), a lexical database for English widely used in the context of WSD. WordNet is organised around the notion of "synonym sets" (*synsets*), which represent a word, its class (noun, verb, ...) and its connections to all semantically related words (synonyms, antonyms, hyponyms,...), as well as a textual definition for each corresponding synset. The current version, WordNet 3.0, contains over 155000 words for 117000 synsets.

3 Local and global WSD Algorithms

3.1 Our local algorithm : A variant of the Lesk Algorithm

Our local algorithm is a variant of the Lesk Algorithm (Lesk, 1986). Proposed more than 25 years ago, it is simple, only requires a dictionary and no training. The score given to a sense pair is the number of common words (space separated strings) in the definition of the senses, without taking into account neither the word order in the definitions (bag-of-words approach), nor any syntactic or morphological information. Variants of this algorithm are still today among the best on English-language texts (Ponzetto and Navigli, 2010).

Our local algorithm exploits the links provided by WordNet: it considers not only the definition of a sense but also the definitions of the linked senses (using all the semantic relations from WordNet) following (Banerjee and Pedersen, 2002), henceforth referred as $ExtLesk$[1]. Contrarily to Banerjee, however, we do not consider the sum of squared sub-string overlaps, but merely a bag-of-words overlap that allows us to generate a dictionary from WordNet, where each word contained in any of the word sense definitions is indexed by a unique integer and where each resulting definition is sorted. Thus we are able to lower the computational complexity from $O(mn)$ to $O(m), m > n$, where m and n are the respective length of two definitions. For example for the definition: "Some kind of evergreen tree", if we say that *Some* is indexed by 123, *kind* by 14, *evergreen* by 34, and tree by 90, then the indexed representation is $\{14, 34, 90, 123\}$.

3.2 Global algorithms

A global algorithm is a method that allows to propagate a local measure to a whole text in order to assign a sense label to each word. The simplest approach is the exhaustive evaluation of

[1] All dictionaries and Java implementations of all algorithms of this article can be found on our WSD page `http://getalp.imag.fr/xwiki/bin/view/WSD/`

sense combinations (BF), used for example in (Banerjee and Pedersen, 2002), that assigns a score to each word sense combination in a given context (window or whole text) and selects the one with the highest score. The main issue with this approach is that it leads to a combinatorial explosion in the length of the context window or text: $\prod_{i=1}^{|T|}(|s(w_i)|)$, where $s(w_i)$ is the set of possible senses of word i of a text T. For this reason it is very difficult to use the BF approach in real-life scenarios as well as on analysis windows of more than a few words.

Several approximation methods can be used in order to overcome the combinatorial explosion problem. On the one hand, *complete approaches*, try to reduce dimensionality using pruning techniques and sense selection heuristics. Some examples include: (Hirst and St-Onge, 1998), based on lexical chains that restrict the possible sense combinations by imposing constraints on the succession of relations in a taxonomy (e.g. WordNet); or (Gelbukh et al., 2005) that review general pruning techniques for Lesk-based algorithms; or yet (Brody and Lapata, 2008).

On the other hand, *incomplete approaches* generally use stochastic sampling techniques to reach a local maximum by exploring as little as necessary of the search space. Our present work focuses on such approaches. Furthermore, we can distinguish two possible variants:

- local neighbourhood-based approaches (new configurations are created from existing configurations) among which are some approaches from artificial intelligence such as genetic algorithms or optimization methods such as simulated annealing ;
- constructive approaches (new configurations are generated by iteratively adding new elements of solutions to the configuration under construction), among which are for example ant colony algorithms.

3.3 Context of our work

The aim of this paper is to compare our Ant Colony Algorithm (incomplete and constructive approach) to other incomplete approaches. We choose to first confront our algorithm to two classical neighbourhood-based approaches that have been used in the context of unsupervised WSD: genetic algorithms (Gelbukh et al., 2003) and simulated annealing (Cowie et al., 1992).

The underlying assumption to our work is that the span of the analysis context should be the whole text (similarly to (Cowie et al., 1992), (Gelbukh et al., 2003) and more recently (Navigli and Lapata, 2010)), rather than a smaller context window (like many other methods do for computational reasons). Indeed, in our opinion, using a context window smaller than that of the whole text raises two main issues: no guarantee on the consistency between two selected senses; contradictory sense assignments outside of the window range.

For example in the following sentence, considering a window of 6 words: "*The two planes were parallel to each other. The pilot had parked them meticulously.*", *plane* may be disambiguated wrongly due to *pilot* being outside the window of *plane*. Furthermore it can be detrimental to the semantic unity in the disambiguation, given that as (Gale et al., 1992) or (Hirst and St-Onge, 1998) pointed out, two words used several times in the same context tend to have the same sense. Therefore, some algorithms that are similar to our Ant Colony Algorithm but that use a context window have not been studied here (notably the adaptation (Mihalcea et al., 2004) of *PageRank* (Brin and Page, 1998) to WSD).

Moreover, we are not interested in comparing these incomplete algorithm, which cannot pragmatically be used in a real-life context, to the optimal disambiguation (Brute Force). Even with a reduced windows of the context and weeks of execution time we were only able to

achieve a 77% coverage of the corpus with BF, as detailed in (Schwab et al., 2011).

4 Global stochastic algorithms for Word Sense Disambiguation

The aim of these algorithms is to assign to each ambiguous word w_i in a text of m words the most appropriate of its senses $w_{i,j}$ given the context. The definition of a sense j of word i is noted $d(w_{i,j})$. The search-space corresponds to all the possible sense combinations for the text being processed. Therefore, a configuration C of the problem can be represented as an array of integers such that $j = C[i]$ is the selected sense j of w_i.

4.1 Problem configuration and global score

The algorithms require some *fitness* measure to evaluate how good a configuration is. With this in mind, the score of the selected sense of a word can be expressed as the sum of the local scores between that sense and the selected senses of all the other words of the text. Hence, in order to obtain a *fitness* value (*global score*) for the whole configuration, it is possible to simply sum the scores for all selected senses of the words of the text: $Score(C) = \sum_{i=1}^{m} \sum_{j=i}^{m} ExtLesk(w_{i,C[i]}, w_{j,C[j]})$. The complexity of this algorithm is hence $O(m^2)$, where m is the number of words in the text.

4.2 Genetic algorithm for Word Sense Disambiguation

The Genetic Algorithm (GA) based on (Gelbukh et al., 2003) can be divided into five phases: initialisation, selection, crossover, mutation and evaluation. During each iteration, the algorithm goes through each phase but the initialisation.

The initialisation phase consists in the generation of a random initial population of λ individuals (λ configurations of the problem).

During the selection phase, the score of each individual of the current population is computed. A crossover ratio (*CR*) is used to determine which individuals of the current population are to be selected for crossover. The probability of an individual being selected is *CR* weighted by the ratio of the score of the current individual over that of the best individual. Individuals who are not selected for crossover are merely cloned (copied) into the new population. Additionally the best individual is systematically kept. After each iteration, the size of each subsequent population is a constant λ.

During the crossover phase individuals are sorted according to their global score. If the number of individuals is odd, the individual with the lowest score is unselected and cloned into the new population as it cannot serve for crossover. The crossover operator is then applied on the individuals two by two in decreasing order of their score: the resulting configurations are swapped around two random pivots (everything but what is between the pivots is swapped).

During the mutation phase, each individual has a probability of mutating (parameter *MR*, for Mutation Rate). A mutation corresponds to *MN* random changes in the configuration. Thus, after the mutation phase, we obtain a modified configuration $C_{c'}$.

The evaluation phase corresponds to the test of the termination criteria: convergence of the score of the best individual. In other words if the score of the best individual remains the same for a number of generations (*STH*), the algorithm terminates.

4.3 Simulated annealing for Word Sense Disambiguation

The simulated annealing approach as described in (Cowie et al., 1992) is based on the physical phenomenon of metal cooling.

Simulated annealing works with the same configuration representation as the genetic algorithm, however it uses a single randomly initialised configuration. The algorithm is organised in cycles and in iterations, each cycle being composed of *IN* iterations. The other parameters are the initial temperature T_0 and the cooling rate $ClR \in [0;1]$. At each iteration a random change is made to the current configuration C_c, which results in a new configuration $C_c^{'}$. Given that $\Delta E = Score(C_c) - Score(C_c^{'})$, $C_c^{'}$, the probability $P(A)$ of acceptance (the probability of replacing $C_{c'}$) of configuration $C_c^{'}$ is :

$$P(A) = \begin{cases} 1 & \text{if } \Delta E < 0 \\ e^{\frac{-\Delta E}{T}} & \text{otherwise} \end{cases}$$

The reason why configurations with lower scores have a chance to be accepted, is to prevent the algorithm from converging on a local maximum. Lower score configurations allow to explore other parts of the search-space which may contain the global maximum.

At the end of each cycle, if the current configuration is the same as the configuration at the end of the previous cycle, the algorithm terminates. Otherwise, the temperature is lowered to $T \cdot ClR$. In other words, the more cycles it takes for the algorithm to converge, the lower the probability to accept lower score configurations: this guarantees an eventual convergence. The configuration with the highest score is saved after each iteration and will be taken as a result regardless of the convergence configuration.

5 Global Ant Colony Algorithm

5.1 Ant Colony Algorithm

Ant colony algorithms (ACA) come from biology and from observations of ant social behavior. Indeed, these insects have the ability to collectively find the shortest path between their nest and a source of food (energy). It has been demonstrated that cooperation inside an ant colony is self-organised and emerges from interactions between individuals. These interactions are often very simple and allow the colony to solve complex problems. This phenomenon is called swarm intelligence (Bonabeau and Théraulaz, 2000) and is increasingly popular in computer science where centralised control systems are often successfully replaced by other types of control based on interactions between simple elements.

Artificial ants have first been used for solving the Traveling Salesman Problem (Dorigo and Gambardella, 1997). In these algorithms, the environment is usually represented by a graph, in which virtual ants exploit pheromone trails deposited by others, or pseudo-randomly explore the graph.

These algorithms are a good alternative for the resolution of problems modeled with graphs. They allow a fast and efficient exploration close to other search methods. Their main advantage is their high adaptivity to changing environments. Readers can refer to (Dorigo and Stützle, 2004), (Monmarche et al., 2009) or (Guinand and Lafourcade, 2010) for a state of the art.

5.2 Ant colony Algorithm for Word Sense Disambiguation

5.2.1 Principle

The environment of the ant colony algorithm is a graph that can be linguistic, a morphological lattice (Rouquet et al., 2010), morpho-syntactic (Schwab and Lafourcade, 2007), or simply organised following the structure of the text (Guinand and Lafourcade, 2010).

Depending on the environment chosen, the results of the algorithm differ. We are currently investigating this aspect, but as the focus of our article is to make a comparison between ACA and the two other methods presented earlier, we will use a simple graph following the structure of the text (see Fig. 1) that uses no external linguistic information (no morpho-syntactic links within a sentence for example).

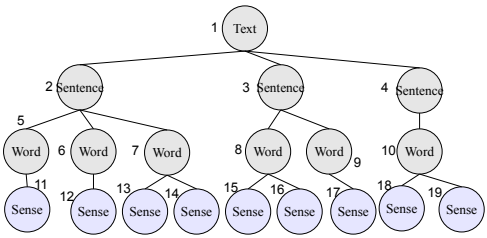


Figure 1: The environment for our experiment: text, sentences and words correspond to common nodes (1-10) and word senses to nests (11-19).

In this graph, we distinguish two types of nodes: nests and plain nodes. Following (Schwab, 2005) or (Guinand and Lafourcade, 2010), each possible word sense is associated to a nest. Nests produce ants that move in the graph in order to find energy and to bring it back to their mother nest: the more energy is deposited by ants, the more ants can be produced by the nest in turn. Ants carry an odour (array) that contains the words of the definition of the sense of its mother nest. From the point of view of an ant, a node can be: (1) *its mother nest*, where it was born; (2) *an enemy nest* that corresponds to another sense of the same word; (3) *a potential friend nest*: any other nest; (4) a *plain node*: any node that is not a nest. Furthermore, to each plain node is also associated an odour vector of a fixed length that is initially empty. For example, in Fig. 1, for an ant born in nest 19: nest 18 is an enemy (as their are linked to the same word node, 10), its potential friend nodes are from 11 to 17 and common nodes are from 1 to 10.

Ant movements depends on the scores given by the local algorithm (cf. Section 3.1), of the presence of energy, of the passage of other ants (when passing on an edge ants leave a pheromone trail that evaporates over time) and of the nodes' odour vectors (ants deposit a part of their odour on the nodes they go through). When an ant arrives on a nest of another term (that corresponds to a sense thereof), it can either continue its exploration or depending on the score between this nest and its mother nest, decide to build a bridge between them and to follow it home. Bridges behave like normal edges except that if at any given time the concentration of pheromone reaches 0, the bridge collapses.

Depending on the lexical information present and the structure of the graph, ants will favour

following bridges between more closely related senses. Thus, the more closely related the senses of the nests are, the more a bridge between them will contribute to their mutual reinforcement and to the sharing of resources between them (thus forming *meta-nests*); while the bridges between more distant senses will tend to fade away. We are thus able to build (constructive approach) interpretative paths[2] through emergent behaviour and to suppress the need to use a complete graph that includes all the links between the senses from the start (as is usually the case with classical graph-based optimisation approaches).

5.2.2 Implementation details

In this section we first present the notations used (Table 1) as well as the parameters of the Ant Colony Algorithm and their typical value ranges (Table 2), followed by a detailed description of the different steps of the algorithm.

Notation	Description
F_A	Nest that corresponds to sense A
f_A	Ant born in nest F_A
$V(X)$	Odour vector associated with X (ant or node)
$E(X)$	Energy on/carried by X (ant or node)
$Eval_f(N)$	Evaluation of a node N by an ant f
$Eval_f(A)$	Evaluation of an edge A (quantity of pheromone) by an ant f
$\varphi_{(t/c)}(A)$	Quantity of pheromone on edge A at given moment t or cycle c

Table 1: Main notations for the Ant Colony Algorithm

Notation	Description	Value
E_a	Energy taken by an ant when it arrives on a node	1-30
E_{max}	Maximum quantity of energy an ant can carry	1-60
δ	Evaporation rate of the pheromone between two cycles	0.0-1.0
E_0	Initial quantity of energy on each node	5-60
ω	Ant life-span	1-30 (cycles)
L_V	Odour vector length	20-200
δ_V	Percentage of the odour vector components (words) deposited by an ant when it arrives on a node	0-100%
c_{ac}	Number of cycles of the simulation	1-500

Table 2: Parameters of the Ant Colony Algorithm and their typical value-ranges

5.2.3 Simulation

The execution of the algorithm is a potentially infinite succession of cycles. After each cycle, the state of the environment can be observed and used to generate a solution. A cycle is composed of the following actions: (1) eliminate dead ants and bridges with no pheromone; (2) for each nest, potentially produce an ant; (3) for each ant: determine its mode (energy search or return); make it move; potentially create an interpretative bridge; (4) update the environment (energy levels of nodes, pheromone and odour vectors).

Ant production, death and energy model Initially, we assign a fixed quantity of energy E_0 to each node of the environment. At the beginning of each cycle, each nest node N has an

[2]Possible interpretation of the text.

opportunity to produce an ant A using 1 unit of energy, with a probability $P(N_A)$. In accordance with (Schwab and Lafourcade, 2007) or (Guinand and Lafourcade, 2010), we define it as the following sigmoid function (often used with artificial neural networks (Lafourcade, 2011)): $P(N_A) = \frac{arctan(E(N))}{\pi} + 0.5$.

When created, an ant has a lifespan of ω cycles (see Table 2). When the life of an ant reaches zero, the ant is deleted at the beginning of the next cycle and the energy it carried is deposited on the node where it died. By thus doing, we ensure that the global energy equilibrium of the system is preserved, which plays a fundamental role in the convergence (monopolization of resources by certain nests) to a solution.

Ant movements The ants' movements are random, but influenced by the environment. When an ant is on a node, it assigns a transition probability to the edges leading to all neighbouring nodes. The probability to cross through an edge A_j in order to reach a node N_i is $P(N_i,A_j) = \frac{Eval_f(N_i,A_j)}{\sum_{k=1,l=1}^{k=n,l=m} Eval_f(N_k,A_l)}$, where $Eval_f(N,A) = Eval_f(N) + Eval_f(A)$ is the evaluation function of a node N when coming from an edge A.

A newborn ant seeks food. It is attracted by the nodes which carry the most energy ($Eval_f(N) = \frac{E(N)}{\sum_0^m E(N_i)}$), but avoids to go through edges with a lot of pheromone, $Eval_f(A) = 1 - \varphi_t(A)$ in order to favour a greater exploration of the search space. The ant collects as much energy as possible until it decides to bring it back home (return mode) with the probability $P(return) = \frac{E(f)}{E_{\max}}$ [3]. Then, it moves while following (statistically) the edges that contain the most pheromone, $Eval_f(A) = \varphi_t(A)$ and leading to nodes with an odour close to their own, $Eval_f(N) = \frac{ExtLesk(V(N),V(f_A))}{\sum_{i=1}^{i=k} ExtLesk(V(N_i),V(f_A))}$.

Creation, deletion and types of bridges When an ant arrives on a node adjacent to a potential friend nest (i.e. that corresponds to a sense of a word), it has to decide between taking any of the possible paths or to go on that nest node. As such, we are dealing with a particular case of the ant path selection algorithm presented above in Section 5.2.3, with $Eval_f(A) = 0$ (The pheromone on the edge is ignored). The only difference is that if the ant chooses to go on the potential friend nest, a bridge between that nest and the ant's home nest is built and the ant follows it to go home. Bridges behave like regular edges, except that if the concentration of pheromone on them reaches 0, they collapse and are removed.

Pheromone Model Ants have two types of behaviours: they are either looking to gather energy or to return to their mother nest. When they move in the graph, they leave pheromone trails on the edges they pass through. The pheromone density on an edge influences the movements of the ants: they prefer to avoid edges with a lot of pheromone when they are seeking energy and to follow them when they want to bring the energy back to their mother nest.

When passing on an edge A, ants leave a trail by depositing a quantity of pheromone $\theta \in \mathbb{R}^+$ such that $\varphi_{t+1}(A) = \varphi_t(A) + \theta$.

Furthermore, at each cycle, there is a slight (linear) evaporation of pheromones (penalizing little frequented paths). Thus, $\varphi_{t+1}(A) = \varphi_t(A) \times (1-\delta)$, where δ is the pheromone evaporation rate.

[3] Consequently, when the ant reaches its carrying capacity, the probability to switch to return mode is 1.

Odour The odour of a nest is the numerical sense vector (as introduced in Section 3.1) and corresponds to the definition of the sense associated to the nest. All ants born in the same nest have the same odour vector. When an ant arrives on a common node N, it deposits some of the components of its odour vector (following a uniform distribution), which will be added to or will replace existing component of the node's vector $V(N)$. The odour of nest nodes on the other hand is never modified.

This mechanism allows ants to find their way back to their mother nest. Indeed, the closer a node is to a given nest, the more ants from that nest will have passed through and deposited odour components. Thus, the odour of that node will reflect its nest neighbourhood and allow ants to find their way by computing the score between their odour (that of their mother nest) and the surrounding nodes and by choosing to go on the node yielding the highest score. This process leaves some room for error (such as an ant arriving on a nest other than its own), which is beneficial as it leads ants to build more bridges (see Section 5.2.3).

5.3 Global Evaluation

At the end of each cycle, we build the current problem configuration from the graph: for each word, we take the sense corresponding to the nest with the highest quantity of energy. Subsequently, we compute the global score of the configuration (see Section 4.1). Over the execution of the algorithm we keep the configuration with the highest absolute score, which will be used at the end to generate the solution.

5.4 Main parameters

Here we present a short characterisation of the influence of the parameters on the emergent phenomena in the system:

- The maximum amount of energy an ant can carry, E_{max}, influences how much an ant explores the environment. Ants cannot go back through an edge they just crossed and have to make circuits to come back to their nest (if the ant does not die before that). The size of the circuits depend on the moment the ants switch to return mode, hence on E_{max}.
- The evaporation rate of the pheromone between cycles (δ) is one of the memories of the system. The higher the rate is, the least the trails from previous ants are given importance and the faster interpretative paths have to be confirmed (passed on) by new ants in order not to be forgotten by the system.
- The initial amount of energy per node (E_0) and the ant life-span (ω) influence the number of ants that can be produced and therefore the probability of reinforcing less likely paths.
- The odour vector length (L_v) and the proportion of odour components deposited by an ant on a plain node (δ_V) are two dependent parameters that influence the global system memory. The higher the length of the vector, the longer the memory of the passage of an ant is kept. On the other hand, the proportion of odour components deposited has the inverse effect.

Given the lack of an analytical way of determining the optimal parameters of both the Ant Colony Algorithm and the other algorithms presented, they have to be estimated experimentally, which is detailed in Section 6.

6 Empirical Evaluation

In this section we first describe the evaluation task we used to evaluate the three systems, followed by the methodology we used for the estimation of the parameters, and then the experimental protocol for the empirical quantitative comparison of the algorithms and subsequently, the interpretation of the results. Finally, we briefly compare the number of evaluations of the semantic similarity score function ($ExtLesk$) and discuss the positioning of our system relatively to the other systems that are evaluated with Semeval 2007 Task 7 (participating systems and more recent advances).

6.1 Evaluation Campaign Task

We evaluated the algorithms with the SemEval 2007 coarse-grained English all-words task 7 corpus (Navigli et al., 2007). Composed of 5 texts from various domains (journalism, book review, travel, computer science, biography), the task consists in annotating 2269 words with one of their possible senses from WordNet, with an average degree of polysemy of 6.19. The evaluation of the output of the algorithm is done considering coarse grained senses distinction i.e. close senses are counted as equivalent (e.g. snow/precipitation and snow/cover).

A Perl script that evaluates the quality of the solutions is provided with the task files and allows us to compute the **P**recision, **R**ecall, and F_1 score[4], which are the standard measures for evaluating WSD algorithms (Navigli, 2009).

6.2 Parameter estimation

The algorithms we are interested in have a certain number of parameters that need tuning in order to obtain the best possible score on the evaluation corpus. There are three approaches:

- Make an educated guess about the value ranges based on *a priori* knowledge about the dynamics of the algorithm;
- Test manually (or semi-manually) several combinations of parameters that *appear* promising and determine the influence of making small adjustments to the values ;
- Use a learning algorithm on a subset of the evaluation corpus, for example with SA or GA to find the parameters that lead to the best score.

Both GA and SA, as presented in (Gelbukh et al., 2003) and (Cowie et al., 1992) respectively, use the Lesk semantic similarity measure as a score metric. We reimplemented them with the $ExtLesk$ measure and used the *optimal* parameters provided. However, the similarity values are higher than with the standard Lesk algorithm and we had to adapt the parameters to reflect that difference. We made one parameter vary at a time over 10 executions, in order to maximise the F_1 measure.

For our ACA, given that an execution of the algorithm is very fast[5], it was possible to use a simplified simulated annealing approach for the automated estimation of the parameters. For each parameter combination we ran the algorithm 50 times and considered the means coupled with the p-values from a one-way ANOVA. We still needed to use our *a priori* knowledge to set likely parameter intervals and discreet steps for each of them[6].

The best parameters we found are:

[4] F_1 is the harmonic mean of P and R. When 100% of the corpus is annotated, $P = R = F_1$.

[5] Depending on the parameters the execution takes between 15s to 40s for the first text of the corpus.

[6] Supervised approach to parameter tuning which does not affect the unsupervised nature of the algorithm itself.

- For GA: $\lambda = 500$, $CR = 0.9$, $MR = 0.15$, $MN = 80$, $CR = 0.9$;
- For SA: $T_0 = 1000$, $ClR = 0.9$, $IN = 1000$;
- For ACA: $\omega = 25$, $E_a = 16$, $E_{max} = 56$, $E_0 = 30$, $\delta_v = 0.9$, $\delta = 0.9$, $L_V = 100$

6.3 Experimental Protocol

The objective of the experiments is to compare the three algorithms according to different criteria. First, the F_1 score obtained on the Task 7 of Semeval 2007 and then the execution time. Furthermore, given that they use the same similarity measure and that it is the computational bottleneck, we also measure the average number of similarity computations between word senses.

Since the algorithms are stochastic, we need to have a representation of the distribution of solutions as accurate as possible in order to make statistically significant comparisons and thus we ran the algorithms 100 times each.

The first step in the evaluation of the significance of the results is the choice of an appropriate statistical tool. In this case we are using a one-way ANOVA analysis (Miller, 1997), coupled with a Tuckey' HSD post-hoc pairwise analysis (Hsu, 1996). These two techniques rely on three principal assumption: independence of the groups, normal distribution of the samples, within group homegeneity of variances.

After the direct comparison of the results we apply a majority voting strategy for each word among ten consecutive executions so as to obtains 100 overlapping vote results. The same evaluation methodology is applied. In both cases, the baseline for our comparison is the first-sense(FS) baseline. Let us now check for the ANOVA assumptions and analyse the results.

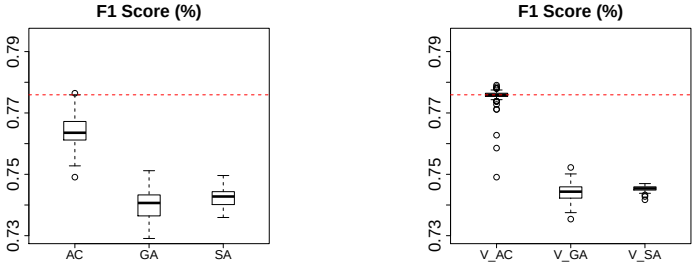
6.4 Quantitative results

In order to check the normality assumption for ANOVA, we computed the correlation between the theoretical (normal distribution) and the empirical quantiles. For all metrics and algorithms there was always a correlation above 0.99. Furthermore we used Levene's variance homogeneity test and found a minimum significance level of 10^{-6} between all algorithms and metrics.

Algorithm	F_1 (%)	σ_{F_1}	$Time(s)$	σ_{Time}	Sim. Eval.	$\sigma(S.Ev.)$
F.S. Baseline	77.59	N/A	N/A	N/A	N/A	N/A
G.A.	73.98(74.53)†	0.0052	4,537.6†	963.2	137,158,739†	13,784.43
S.A.	74.23(75.18)†	0.0028	1,436.6†	167.3	4,405,304†	50,805.27
A.C.A.	76.41(77.50)†	0.0048	65.46†	0.199	1,559,049†	17,482.45

Table 3: Comparison of the F_1 scores (after vote between brackets), execution times and similarity measure evaluations of the algorithms ($\dagger \longleftrightarrow p < 0.01$) over texts 2 through 5.

Table 3 presents the results, for the three algorithms with the F_1 scores, execution time and number of evaluations of the similarity measure along with their respective standard deviations. For all three metrics and between all three algorithms, the difference in the means was systematically significant with p<0.01. Since the first text was used to train the ACA parameters, in this sections the F_1 scores presented are calculated for the 4 next texts in order to remove any bias.



(a) Normal Execution (b) Majority Vote

Figure 2: Boxplots of the F_1 compared to the first sense baseline (dashed line)

Figures 2a and 2b respectively present boxplots of the distributions of F_1 scores for all three algorithms with and without vote compared to the first sense baseline.

In terms of F_1 score, SA and GA obtain similar results, even though SA is slightly better and shows a lower variability in the score distribution. As for ACA, the scores are on average +1.61% better than SA and +1.76% better than GA, with a variability similar to that of GA. All three algorithms are below the FS baseline, even though the maximum of the ACA distribution is close. After applying a majority vote on the answer files 10 by 10, for SA and GA there is a slight improvement of the scores ($p<0.01$), respectively +0.17% and +0.46% and for ACA, there is a larger improvement ($p<0.01$) of +1.17% (despite a certain number of lower-bound outliers). ACA tends to converge on solutions close to the FS baseline. After the vote, the distribution is practically centred around the latter as far as the score is concerned.

In terms of execution times there are huge differences between the algorithms, the slowest being GA, which on average runs over 1.5h (±16 min). SA is much faster and takes on average about 24m (± 2.8m), but still remains much slower that ACA, which converges on average in 65s(± 190ms). As one would expect, the number of evaluations of the similarity measure is directly correlated with execution times of the algorithms (cor_p=0.9969).

6.5 Comparison to other Task 7 systems

For the comparison to other systems, we restricted ourselves to those that disambiguate the whole text and not only a subset, in order to make a fair comparison. Furthermore, given that the results are over the 5 texts, we will consider the ACA results for the 5 texts as well contrarily to the previous section. Vis-a-vis the original participants, our algorithms is ahead of all other unsupervised systems, and beats the weakest supervised system by getting very close to the first sense baseline. However, most supervised systems are still ahead. If we add more recent results from the experiments of (Navigli and Pozetto, 2012) using Babelnet, a multilingual database which adds multilingual and monolingual links to WordNet, the Degree algorithm reaches a score almost as good as ACA (only WordNet) -0.63%, followed by Pagerank (Mihalcea et al., 2004) (adapted to Babelnet) -5.43%. When a voting strategy is used, ACA is ahead with 1.75% compared to Degree. However, it is important to note that when looking a the scores per part of speech, Degree exhibits notably higher results for nouns (85% versus 76.35%), while ACA performs much better for adverbs adjectives and verbs (83.98%, 82.44%, 74.16%).

System	A	P	R	F_1
UoR-SSI†	100	83.21	83.21	83.21
NUS-PT†	100	82.5	82.5	82.5
NUS-ML†	100	81.58	81.58	81.58
LCC-WSD†	100	81.45	81.45	81.45
GPLSI†	100	79.55	79.55	79.55
ACA Maj. Vote (Wn)	100	78.76	78.76	78.76
UPV-WSD†	100	78.63	78.63	78.63
ACA (Wn)	100	77.64	77.64	77.64
Degree (Babelnet)	100	77.01	77.01	77.01
Page Rank (Babelnet)	100	72.60	72.60	72.60
TKB-UO	100	70.21	70.21	70.21
RACAI-SYNWSD	100	65.71	65.7	65.7
FS Baseline	100	78.89	78.89	78.89

Table 4: Comparison with 100% coverage systems evaluated on Semeval2007-Task7 († ⟷ supervised system)

Conclusions and perspectives

In this paper we have presented three stochastic algorithms for knowledge-based unsupervised Word Sense Disambiguation: a genetic algorithm, a simulated annealing algorithm, and an ant colony algorithm; two from the state of the art, and our own ant colony algorithm. All three algorithms belong to the family of incomplete approaches, which allows us to consider the whole text as the disambiguation context and thus to go further towards ensuring the global coherence of the disambiguation of the text. We have estimated the *best* parameter values and then evaluated and compared the three algorithms empirically in terms of the F_1 score on Semeval 2007 Task 7, the execution time as well as the number of evaluation of the similarity measure. We found that the ACA is notably better both in terms of score and execution time. Furthermore, the number of evaluations of the similarity measure are directly correlated with the computation time. Then, we applied a majority vote strategy, which led to only small improvements for SA and GA and more substantial improvements for ACA. The vote strategy allowed ACA to reach the level of the first sense baseline, to beat the state-of-the-art unsupervised systems and the lowest performing supervised systems.

However, some open-ended questions remain. The three methods rely on parameters that are not (*a priori*) linguistically grounded and that have an influence both on the results and the computation time. The estimation of the parameters, whether manual or through an automated learning algorithm prevent these algorithms from being entirely unsupervised. However, the degree of supervision remains far below supervised approaches that use training corpora approximately 1000 times larger. Our work is currently focussed on the study of these parameters for ACA. Their values seem to depend mostly on the structure of the text and on its consequences on the environment graph of ACA, as we have outlined in the paper. Moreover, we are also interested in determining the degree to which the similarity measure and the lexical resources it uses influences the parameters. We are currently adapting our $ExtLesk$ to use Babelnet in order to investigate the matter further as well as to enable us to perform Multilingual Word Sense Disambiguation.

References

Agirre, E. and Edmonds, P. (2006). *Word Sense Disambiguation: Algorithms and Applications (Text, Speech and LT)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA.

Banerjee, S. and Pedersen, T. (2002). An adapted lesk algorithm for word sense disambiguation using wordnet. In *CICLing 2002*, Mexico City.

Bonabeau, É. and Théraulaz, G. (2000). L'intelligence en essaim. *Pour la science*, (271):66–73.

Brin, S. and Page, L. (1998). The anatomy of a large-scale hypertextual web search engine. In *Proceedings of the seventh international conference on World Wide Web 7*, WWW7, pages 107–117, Amsterdam, The Netherlands, The Netherlands. Elsevier Science Publishers B. V.

Brody, S. and Lapata, M. (2008). Good neighbors make good senses: Exploiting distributional similarity for unsupervised WSD. In *Proceedings of the 22nd International Conference on Computational Linguistics (Coling 2008)*, pages 65–72, Manchester, UK.

Cowie, J., Guthrie, J., and Guthrie, L. (1992). Lexical disambiguation using simulated annealing. In *COLING 1992*, volume 1, pages 359–365, Nantes, France.

Cramer, I., Wandmacher, T., and Waltinger, U. (2010). *WordNet: An electronic lexical database*, chapter Modeling, Learning and Processing of Text Technological Data Structures. Springer.

Dorigo and Stützle (2004). *Ant Colony Optimization*. MIT-Press.

Dorigo, M. and Gambardella, L. (1997). Ant colony system: A cooperative learning approach to the traveling salesman problem. *IEEE Transactions on Evolutionary Computation*, 1:53–66.

Fellbaum, C. (1998). *WordNet: An Electronic Lexical Database (Language, Speech, and Communication)*. The MIT Press.

Gale, W., Church, K., and Yarowsky, D. (1992). One sense per discourse. In *Fifth DARPA Speech and Natural Language Workshop*, pages 233–237, Harriman, New-York, États-Unis.

Gelbukh, A., Sidorov, G., and Han, S. (2005). On some optimization heuristics for lesk-like wsd algorithms. In *10th International Conference on Applications of Natural Languages to Information Systems, NLDB-2005*, pages 402–405, Alicante, Spain.

Gelbukh, A., Sidorov, G., and Han, S. Y. (2003). Evolutionary approach to natural language wsd through global coherence optimization. *WSEAS Transactions on Communications*, 2(1):11–19.

Guinand, F. and Lafourcade, M. (2010). *Artificial Ants. From Collective Intelligence to Real-life Optimization and Beyond*, chapter 20 - Artificial ants for NLP, pages 455–492. Lavoisier.

Hirst, G. and St-Onge, D. D. (1998). Lexical chains as representations of context for the detection and correction of malapropisms. *WordNet: An electronic Lexical Database. C. Fellbaum. Ed. MIT Press. Cambridge. MA*, pages 305–332. Ed. MIT Press.

Hsu, J. (1996). *Multiple Comparisons: Theory and Methods*. Chapman and Hall.

Ide, N. and Véronis, J. (1998). Wsd: the state of the art. *Computational Linguistics*, 28(1):1–41.

Lafourcade, M. (2011). Lexique et analyse sémantique de textes – structures, acquisitions, calculs, et jeux de mots. HDR de l'Université Montpellier II.

Lesk, M. (1986). Automatic sense disambiguation using mrd: how to tell a pine cone from an ice cream cone. In *Proceedings of SIGDOC '86*, pages 24–26, New York, NY, USA. ACM.

Mihalcea, R., Tarau, P., and Figa, E. (2004). Pagerank on semantic networks, with application to word sense disambiguation. In *Proceedings of the 20th international conference on Computational Linguistics*, COLING '04, Stroudsburg, PA, USA. Association for Computational Linguistics.

Miller, R. G. (1997). *Beyond ANOVA: Basics of Applied Statistics (Texts in Statistical Science Series)*. Chapman & Hall/CRC.

Monmarche, N., Guinand, F., and Siarry, P., editors (2009). *Fourmis Artificielles et Traitement de la Langue Naturelle*. Lavoisier, Prague, Czech Republic.

Navigli, R. (2009). Wsd: a survey. *ACM Computing Surveys*, 41(2):1–69.

Navigli, R. and Lapata, M. (2010). An experimental study of graph connectivity for unsupervised word sense disambiguation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 32:678–692.

Navigli, R., Litkowski, K. C., and Hargraves, O. (2007). Semeval-2007 task 07: Coarse-grained english all-words task. In *SemEval-2007*, pages 30–35, Prague, Czech Republic.

Navigli, R. and Pozetto, S. P. (2012). Babelnet: The automatic construction, evaluation and application of a wide-coverage multilingual semantic network. http://dx.doi.org/10.1016/j.artint.2012.07.004.

Pedersen, T., Banerjee, S., and Patwardhan, S. (2005). Maximizing Semantic Relatedness to Perform WSD. Research report, University of Minnesota Supercomputing Institute.

Ponzetto, S. P. and Navigli, R. (2010). Knowledge-rich word sense disambiguation rivaling supervised systems. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 1522–1531.

Rouquet, D., Falaise, A., Schwab, D., Boitet, C., Bellynck, V., Nguyen, H.-T., Mangeot, M., and Guilbaud, J.-P. (2010). Rapport final de synthèse, passage à l'échelle et implémentation : Extraction de contenu sémantique dans des masses de données textuelles multilingues. Technical report, Agence Nationale de la Recherche.

Schwab, D. (2005). *Approche hybride pour la modélisation, la détection et l'exploitation des fonctions lexicales en vue de l'analyse sémantique de texte*. PhD thesis, Université Montpellier 2.

Schwab, D., Goulian, J., and Guillaume, N. (2011). Désambiguïsation lexicale par propagation de mesures sémantiques locales par algorithmes à colonies de fourmissation lexicale par propagation de mesures sémantiques locales par algorithmes à colonies de fourmis. In *Traitement Automatique des Langues Naturelles (TALN)*, Montpellier, France.

Schwab, D. and Lafourcade, M. (2007). Lexical functions for ants based semantic analysis. In *ICAI'07- The 2007 International Conference on Artificial Intelligence*, Las Vegas, Nevada, USA.

Tchechmedjiev, A. (2012). état de l'art: Mesures de similarité sémantique et algorithmes globaux pour la désambiguïsation lexicale a base de connaissances. In *RECITAL 2012*, Grenoble. ATALA.

Learnability-based Syntactic Annotation Design

Roy Schwartz *Omri Abend* *Ari Rappoport*
Institute of Computer Science, Hebrew University of Jerusalem
{roys02|omria01|arir}@cs.huji.ac.il

Abstract

There is often more than one way to represent syntactic structures, even within a given formalism. Selecting one representation over another may affect parsing performance. Therefore, selecting between alternative syntactic representations (henceforth, *syntactic selection*) is an essential step in designing an annotation scheme. We present a methodology for syntactic selection and apply it to six central dependency structures. Our methodology compares pairs of annotation schemes that differ in the annotation of a single structure. It selects the more *learnable* scheme, namely the one that can be better learned using statistical parsers. We find that in three of the structures, one annotation is unequivocally better than the alternatives. Our results are consistent over various settings involving five parsers and two definitions of learnability. Furthermore, we show that the learnability gains incurred by our selections are both considerable (error reductions of up to 19.8%) and additive. The contribution of this work is in demonstrating that syntactic selection has a substantial and predictable effect on parsing performance, and showing that this effect can be effectively used in designing syntactic annotation schemes.

Keywords: Syntactic annotation design, Learnability, Parsing.

Proceedings of COLING 2012: Technical Papers, pages 2405–2422,
COLING 2012, Mumbai, December 2012.

1 Introduction

The formal manner in which syntactic relations are represented is at the core of the study of grammar. Numerous representations have been proposed over the years for expressing similar syntactic relations. This diversity of representations is expressed in a variety of syntactic annotation schemes currently in use in NLP. Examples include, for constituency annotation, schemes by (Marcus et al. 1993; Sampson 1995; Nelson et al. 2002, *inter alia)* and for dependency annotation, schemes by (Collins 1999; Rambow et al. 2002; Yamada and Matsumoto 2003; Johansson and Nugues 2007, *inter alia)*. Variation within the same formalism is expressed in structures that have several alternative annotations (henceforth *Varying Syntactic Structure* or *VSS*).

In this work we focus on dependency structures, where some of the most basic structures are VSS's. One example is prepositional phrases, which consist of a preposition followed by a noun phrase (e.g., "about everyone"). While some schemes select the preposition to head the NP (Collins 1999), others select the NP as the head of the preposition (Johansson and Nugues 2007) (see Figure 1). Other prominent VSS's include coordination structures and verb group constructions (see Section 3). In fact, more than 40% of the tokens in the Penn Treebank (Marcus et al. 1993) participate in at least one VSS (Schwartz et al. 2011).

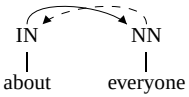


Figure 1: An example of a prepositional phrase – a Varying Syntactic Structure (VSS). Both annotation alternatives for this structure are plausible: either setting the preposition ("about" – solid line) as head, or the noun phrase ("everyone" – dashed line).

Despite the similar content represented by the alternative annotations to VSS's, selecting one over the other (*syntactic selection*) may have significant empirical implications. Previous work showed that syntactic selection can affect the parsing performance of a specific parser (see Section 7). In this work, we are the first to show that in some VSS's, syntactic selections improves parsing performance consistently across different parsers. As our findings are not parser-specific, they can be used to guide future syntactic annotation design.

The empirical implications of syntactic selection stem from the inter-relations between the VSS's and their surrounding structures. Figure 2 presents two alternative annotations for the sentence "he is sure about everyone". The alternatives differ in whether the preposition ("about") or the NP ("everyone") is selected to head the PP ("about everyone"). The two annotations can be deterministically derived from one another and express a similar syntactic relation, namely in both cases the PP is the complement of the adjective "sure". However, selecting one of the alternatives (the preposition) and not the other (the NP) results in a more learnable scheme.

Concretely, in dependency grammar, an adjective's complement is encoded by setting the head of the complement (either "about" or "everyone") as a dependent of the adjective ("sure"). It is plausible that a parser which is strongly guided by POS tags would not select an adjective ("sure") as the head of a noun ("everyone") as it is unlikely for adjectives to head nouns. This would result in a parsing error as in Figure 2(b). On the other hand, a similar parser would more likely select an adjective ("sure") to head a preposition ("about"), resulting in a correct parse as in Figure 2(a). Indeed, the MST parser (McDonald et al. 2005) exhibits such behavior.

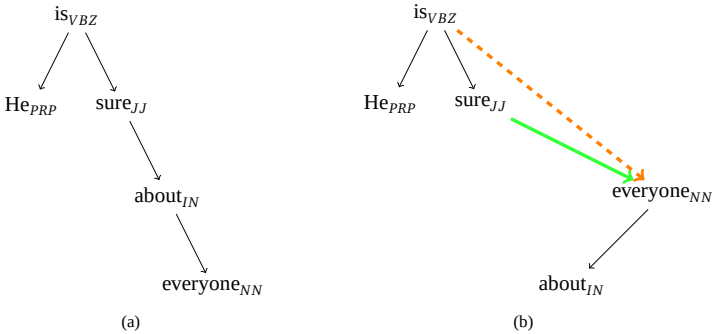


Figure 2: Exploring the effect of VSS annotation on neighboring structures. The sentence "He is sure about everyone", annotated when prepositions head PPs (Figure 2(a)) and when NPs head PPs (Figure 2(b)). Thin solid black lines mark gold+parser edges, thick green solid edges mark **gold edges**, and thick orange dashed lines mark **parser edges**.

The implications of syntactic selection underscore the importance of taking empirical considerations into account when designing an annotation scheme. In this work, we propose *learnability* as an empirical criterion for syntactic selection. The notion that more learnable schemes are preferable is motivated both practically and theoretically. Practically, more learnable schemes result in more accurate parsers. Theoretically, learnability has been a major consideration in the design of phrase structure grammar (Chomsky 2006), and can also be seen as a measure of simplicity, a fundamental principle in many other scientific fields (see Section 7).

We present a learnability-based methodology for syntactic selection and apply it to six central VSS's. We compare alternative annotations for each VSS, by examining pairs of schemes that differ only in their annotation of this VSS. For each pair, we pick the scheme that can be more easily learned using statistical parsers. We select an annotation for this VSS if we find that the schemes that use this annotation are consistently picked.

We experiment with five parsers of various types and using two different learnability measures. We obtain highly consistent results. Our experiments show that for three of the VSS's there is one alternative that is more learnable over all settings. That is, training any of the five parsers on an annotation scheme that uses the more learnable alternative results in higher parsing performance. The differences are substantial in magnitude, yielding error reductions that range between 2.4%-19.8%. Moreover, this gain is additive – using all three of the more learnable alternatives results in an even more accurate parser.

To further establish learnability as a coherent empirical criterion for syntactic selection, we show that our results are consistent with a parser-independent measure based on information theoretic notions.

The contribution of this paper is twofold. First, we present the first study focusing on syntactic selection and showing that it has a substantial and predictable effect on parsing performance. Second, we show that this effect can be used for designing syntactic annotation schemes. Specifically, our findings indicate that future dependency schemes should use (a) prepositions as heads of PPs (b) conjuncts as heads of coordination structures and (c) nouns (and not their determiners) as heads of NPs.

Section 2 describes our methodology. Section 3 discusses Varying Syntactic Structures (VSS). Experimental setup and results are described in Sections 4, 5. Section 6 discusses our methodology and presents further experiments that provide a wider context for understanding our findings. Section 7 surveys related work.

2 Methodology

We present a learnability-based methodology for selecting between alternative annotations for VSS's.

2.1 Notation

In the following we give a formal definition of an annotation scheme. We then turn to describe the different settings in which our methodology conducts experiments.

Our methodology experiments with a set S of VSS's. For each $s \in S$, we examine a set of alternative annotations. For clarity of presentation we assume each VSS has exactly two alternatives and denote them α_s, β_s. Let k denote the size of S ($k = 6$ in our experiments).

An annotation scheme is defined as a selection of an annotation for each of the structures in the language. It therefore includes a (fixed) annotation for non-VSS's, as well as a selected annotation for each of the VSS's. We can thus represent a scheme $\mathscr{A}$ as a k-tuple that selects one of the alternative annotations for each of the VSS's in S (all in all, 2^k schemes). Table 1 shows an example of two annotation schemes that differ in the annotation of exactly one structure (s_4).

Structure	s_1	s_2	s_3	s_4	s_5	s_6
$\mathscr{A}_1$	α	β	α	$\boldsymbol{\alpha}$	α	β
$\mathscr{A}_2$	α	β	α	$\boldsymbol{\beta}$	α	β

Table 1: Applying our methodology to VSS's ($s_1, \ldots, s_6$): $\mathscr{A}_1, \mathscr{A}_2$ are annotation schemes that are identical in their annotation of all the VSS's but s_4 (bold red **column**). α, β are short for $\alpha_{s_i}, \beta_{s_i}$ respectively.

To obtain robust results, our methodology repeats each experiment in different settings, each determined by a parser and a learnability measure. We use P (L) to refer to the set of parsers (learnability measures). We use $|P| = 5, |L| = 2$, all in all $5 \times 2 = 10$ settings.

2.2 Learnability Measures

We propose two straightforward definitions of learnability. They are both defined with respect to a parser p and an annotation scheme $\mathscr{A}$ (as defined above). Both measures assume a fixed corpus partitioned into a training set and a test set.

The first measure is "Accuracy-Learnability". To compute it, we train p on the training set annotated according to $\mathscr{A}$, parse the test set, and evaluate it against the annotation determined by $\mathscr{A}$. We use attachment score for evaluation, which is the standard measure for dependency parsing evaluation. An annotation for which p receives a higher attachment score is considered more learnable.

The second measure is "Rate-Learnability" that measures the rate in which the different annotation schemes can be learned to a given accuracy. We define a target attachment score β. We train p on a corpus annotated with $\mathscr{A}$ several times, using an increasingly larger number of samples. We then

evaluate the trained parser on our test data (annotated with $\mathscr{A}$) and create a learning curve of p and $\mathscr{A}$. An annotation for which p reaches β using less training samples is considered more learnable.

2.3 Learnability-based Methodology

We turn to describing a methodology for selecting learnable annotations for VSS's. The methodology runs a set of experiments, each using a parser p, a learnability measure l and a scheme $\mathscr{A}$. In each experiment, we compute the learnability of $\mathscr{A}$ with respect to p and l.

For every $s \in S$ and alternative annotations α_s, β_s, there are $2^k/2$ pairs of schemes that differ only in their annotation of s, one using α_s, and the other using β_s (see Table 1 for an example). Given a parser p and a learnability measure l, we compute the learnability of each pair of schemes and pick the more learnable scheme (see Table 2). We count the number of pairs in which the scheme using α_s is picked and the number of pairs in which the scheme using β_s is picked. We thus receive two figures that sum up to $2^k/2$ (32 in our experiments).

Annotation	s_1	s_2	s_3	s_4	s_5	s_6	score
$\mathscr{A}_1$	α	α	α	α	α	α	0.91
$\mathscr{A}_2$	α	α	α	α	α	β	**0.92**
$\mathscr{A}_3$	α	α	α	α	β	α	**0.94**
$\mathscr{A}_4$	α	α	α	α	β	β	0.935
⋮							
$\mathscr{A}_{2^k-1}$	β	β	β	β	β	α	0.892
$\mathscr{A}_{2^k}$	β	β	β	β	β	β	**0.896**

Table 2: Applying our methodology for selecting a syntactic annotation for VSS s_6, under parser p and learnability measure l: each row in the table is an experiment with annotation scheme $\mathscr{A}_i$. The experiment compares the learnability (last column) of pairs of annotation schemes that differ only in their annotation of s_6 (where the annotations for $s_1, \ldots, s_5$ are fixed). For each pair of annotation schemes, the more learnable annotation for s_6 is in boldface (blue for $\boldsymbol{\alpha}$, red for $\boldsymbol{\beta}$).

We then define a significance value $1 \geq r \gg 0.5$. If one annotation (say α_s) is more learnable than the other (with respect to p, l) in a relative portion r of these pairs, we say that p is r-*biased* towards α_s with respect to l.

If for some $s \in S$, it holds that for every $p \in P, l \in L$, p is r-biased ($r \gg 0.5$) with respect to l towards the same annotation (say, α_s), we say there is a *unanimous* r-*bias* towards α_s. Consequently, α_s is the *empirically preferred annotation* of s.

3 Varying Dependency Structures

Varying syntactic structures are prevalent in many syntactic formalisms (see Section 7). In this section we focus on dependency structures.

Dependency structures receive varying annotation when the identity of the structure's head is debatable. This stems from the multiple, occasionally conflicting, criteria for defining a head. A few of the more generally acknowledged criteria for defining H to be the head of D in constituent C are (Kübler et al. 2009):

1. H determines the syntactic category of C and can often replace C.
2. H determines the semantic category of C; D gives semantic specification.
3. The form of D depends on H.

These definitions can often be applied to determine the identity of the head. For example, according to (1,2) a noun is the head of its modifying adjective (e.g., "cat" in "big cat") and a verb is the head of its adverb (e.g., "eat" in "eat quickly").

In VSS's, these criteria are either inapplicable or conflicting. For example, in a sequence of proper nouns (e.g., "John Smith"), neither criterion is applicable. In a verb group construction (e.g., "can eat"), the main verb should be the head according to (2). On the other hand, the preceding modal restricts the main verb to be in infinitive form, and thus should be the head according to (3) (e.g., "he can eat" vs. "he eat**s**").

Such structures have led to the creation of several dependency schemes, each taking a different approach to annotating them (Collins 1999; Rambow et al. 2002; Yamada and Matsumoto 2003; Johansson and Nugues 2007, *inter alia)*. We turn to describing the VSS's that we experiment with and the alternatives annotations we consider for them. All of these annotations are in use in NLP. They are also plausible from a theoretical standpoint. Figure 3 shows a diagram for each of the structures, along with their possible annotations.

Coordination Structures are composed of two words, separated by a conjunction (e.g., "John and Mary"). It is not clear which token should be the head of this structure, if any (Nilsson et al. 2006). We consider two alternative annotations: (a) setting the conjunction as head, and both conjuncts as its dependents and (b) setting either of the conjuncts as head, selected according to the specific structure type (e.g., noun phrase, verb phrase).

Infinitive Verbs are verb phrases that contain the sequence "to" + infinitive verb (e.g., "to eat"). In (Yamada and Matsumoto 2003) the verb is the head, while in (Collins 1999) the "to" token is the head. We consider both annotations.

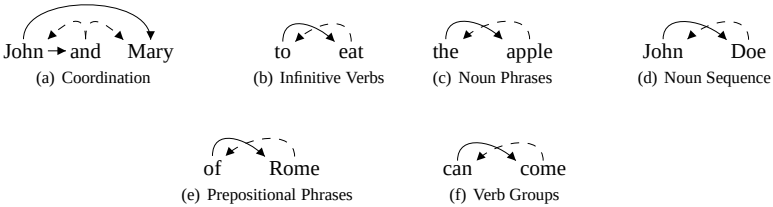
Noun Phrases that contain a determiner and a noun (e.g., "the apple" or "a dog"). Either the determiner (Bosco and Lombardo 2004) or the noun (Collins 1999) may serve as the head. We consider both annotations.

Noun Sequences are noun phrases that contain sequences of more than one noun (e.g., "John Doe"). Various alternative annotations for this structure include (Collins 1999), which takes the last noun as head, and BIO's scheme which is somewhat more complex (Dredze et al. 2007). We consider either the rightmost or the leftmost noun as head, and mark all other nouns as its dependents.

Prepositional Phrases consist of a preposition and a noun phrase (e.g., "in a bag" or "of Rome"). Complement clauses that contain a subordinating conjunction (e.g., "after you go") are also included[1]. Either the preposition/subordinating conjunction (Collins 1999) or the NP/clause (Johansson and Nugues 2007) can be the head. We consider both alternatives.

Verb Groups are composed of a verb and a modal verb (e.g., "can come"). Some schemes select the modal as head (Collins 1999), others select the verb (Rambow et al. 2002). We consider both

[1]For brevity, we use the term Prepositional Phrases to refer to both structures.



(a) Coordination (b) Infinitive Verbs (c) Noun Phrases (d) Noun Sequence

(e) Prepositional Phrases (f) Verb Groups

Figure 3: The VSS's with which we experiment. The possible annotations for each structure are marked using solid and dashed lines.

alternatives[2].

4 Experimental Setup

4.1 The Parsers

In this work we experiment with five parsers of different types. We briefly describe them.

Dependency Model with Valence (DMV) (Klein and Manning 2004) is a generative parser that defines a probabilistic grammar for unlabeled dependency structures. This parser is widely used in the field of *unsupervised* dependency parsing, where the great majority of recent works are in fact elaborations of this model (e.g., (Cohen and Smith 2009; Headden III et al. 2009)). In our experiments we use a *supervised* version of this parser, by training it using maximum likelihood estimation (MLE). This approach was used in various previous works as an upper bound for the unsupervised model (Blunsom and Cohn 2010; Spitkovsky et al. 2011). Decoding is performed using the Viterbi algorithm[3].

MST Parser (McDonald et al. 2005)[4] formulates dependency parsing as a search for a maximum spanning tree (MST). It uses online training and extends the Margin Infused Relaxed Algorithm (MIRA) (Crammer and Singer 2003) to learning with structured outputs.

Clear Parser (Choi and Nicolov 2009)[5] is a fast transition-based parser that uses the robust risk minimization technique (Zhang et al. 2002). *k*-best ranking is used to prune the next state in decoding.

S_u Parser (Nivre 2009)[6] is a transition-based parser and an extension of the MALT parser (Nivre et al. 2006). The parser starts by constructing arcs between adjacent words and then swaps the order of input words in order to learn more complex structures. It uses the *stackeager* algorithm, and is trained using various linear classifiers (including SVM).

NonDir Parser (Goldberg and Elhadad 2010)[7] is a non-directional, easy-first parser, which is greedy and deterministic. It first attempts to induce a non-directional version of the easiest arcs in

[2]Some definitions of verb groups also include auxiliaries. We choose to exclude them from our definition since we use the PTB POS set, which distinguishes modals, but not auxiliaries, from other verbs.

[3]http://www.cs.columbia.edu/~scohen/parser.html

[4]http://www.seas.upenn.edu/~strctlrn/MSTParser/MSTParser.html

[5]http://code.google.com/p/clearparser/

[6]http://maltparser.org/

[7]http://www.cs.bgu.ac.il/~yoavg/software/easyfirst/

a dependency structure, and continues by iteratively selecting the best pair of neighbors to connect, until a complete dependency tree is created.

These parsers span the major approaches to statistical dependency parsing. The two main approaches are (Kübler et al. 2009) (a) *transition-based* methods that use state machines to map sentences to dependency graphs, attempting to reach the optimal state; and (b) *graph-based* methods, which try to find the best scoring dependency graph in some graph space. *Clear Parser* and S_u *Parser* are examples of (a), while *MST Parser* and *DMV* are examples of (b). NonDir takes a somewhat different parsing approach.

4.2 Technical Details

Following standard practice in English, used in the great majority of recent works, all the corpora are generated by converting constituency annotation to dependency using a set of head percolation rules[8]. Using these rules is also suitable here since they can easily be manipulated to create the different corpora required for applying our methodology.

Parsers are trained on sections 2–21 of the Penn TreeBank (PTB) WSJ corpus (Marcus et al. 1993), and are tested on section 23. We use the default feature set for each of the parsers. Evaluation is done using unlabeled attachment score, a common evaluation measure for dependency parsing.

For the Rate-Learnability measure, we select a different β value for each parser, due to their different performance levels; β is set to be the attachment score of the least learnable annotation for that parser, as determined by our experiments with the Accuracy-Learnability measure. This is the highest value of β that all schemes would reach at some point along their learning curve.

5 Results

Table 3 shows our results. In three out of the six structures, a strong unanimous bias is found. A unanimous 0.9-bias is found towards (a) selecting the preposition as head of prepositional phrases, and (b) selecting either of the conjuncts as head of coordination structures. A unanimous 0.7-bias is found towards the noun in noun phrases. For these structures, one annotation is clearly more learnable than the other, independently of the selected annotations for the other structures. This gives an empirical motivation for using these annotations.

In two of the remaining structures (verb groups and noun sequences), we find a trend towards one of the annotations; in five of the settings a 0.7-bias is found towards one alternative (modal and leftmost noun, respectively). In the other five settings no strong bias is found towards either alternative. In these structures, it might be the case that certain modeling assumptions incorporated into the parsers affect whether one alternative is preferred or not. This calls for a more detailed investigation, which we defer to future work.

Finally, no considerable bias is found in the infinitive verb structures, as a 0.7-bias towards any alternative is found in only one setting. Thus, our experiments suggest no preference towards either alternative in this case.

[8] We use a slightly modified version of the *pennconvertor*, tailored for our experimental setup (`http://nlp.cs.lth.se/software/treebank_converter/`) (Johansson and Nugues 2007).

Structure	*Setting / Annotation*	*DMV*		*MST*		*Clear*		S_u		*N.D.*		
		A.L.	*R.L.*	*A.L.*	*R.L.*	*A.L.*	*R.L.*	*A.L.*	*R.L.*	*A.L.*	*R.L.*	
Coord.	**CONJ**	32	30.5	32	32	32	32	32	32	32	32	◄
	CC	0	1.5	0	0	0	0	0	0	0	0	
Inf. Verbs	TO	16	17	19	17	21	17.5	25	19	18.5	13	
	VB	16	15	13	15	11	14.5	7	13	13.5	19	
NP	**NN**	24	24	32	24	32	23	32	24.5	30	23.5	◄
	DT	8	8	0	8	0	9	0	7.5	2	8.5	
N. Seq.	LEFT	25.5	24	29	21.5	32	31.5	21.5	18	11.5	12	
	RIGHT	6.5	8	3	10.5	0	0.5	10.5	14	20.5	20	
PP	**IN**	32	28.5	32	32	32	32	32	32	32	32	◄
	NP	0	3.5	0	0	0	0	0	0	0	0	
Verb Gr.	MD	32	23	28	20	23.5	20	15	17	24	17	
	VB	0	9	4	12	8.5	12	17	15	8	15	

Table 3: Exploring the learnability of the different annotation schemes. Each row pair corresponds to a pair of annotations for a given VSS, and each column pair corresponds to a parser, under Accuracy-Learnability (*A.L.*) and Rate-Learnability (*R.L.*) (see Section 2). For a given VSS, learnability measure and parser, we show the number of times one annotation is more learnable than the alternative. There are 32 experiments with each such combination, each has a single winner, resulting in a pair of numbers that sums up to 32. Gray cells mark settings in which the annotation is substantially more learnable than the alternative (dark/light gray correspond to $r = 0.9/0.7$ respectively). Rows marked with an arrow (◄) mark annotations that are *unanimously* biased. The annotations (see Section 3): Coordinations – headed by one of the conjuncts (CONJ) or by the conjunction (CC) ; Infinitive Verbs – headed by "to" (TO) or by the Verb (VB) ; Noun Phrases – headed by the noun (Noun) or by the determiner (DT) ; Noun Sequences – headed by the left/rightmost noun (LEFT/RIGHT); Prepositional Phrases – headed by the preposition (IN) or by the noun phrase (NP) ; Verb Groups – headed by the modal (MD) or by the Verb (VB). The Parsers (see Section 4.1): *DMV* (Klein and Manning 2004) ; *MST* (McDonald et al. 2005) ; *Clear* (Choi and Nicolov 2009) ; S_u (Nivre 2009) ; *N.D.* – NonDir (Goldberg and Elhadad 2010).

5.1 Analysis

The empirically preferred annotations cannot be reduced to any simple, intuitive rule. For example, they do not match simple distinctions such as the one between closed and open classes: some of the more learnable annotations select closed class tags as heads (e.g., the preposition in prepositional phrases), while others select open class tags (e.g., the noun in noun phrases). Similarly, it is also not necessarily the rightmost or the leftmost word in the structure that is preferred.

Our results indicate that the biases are substantial. Table 4 shows that the difference between the accuracies of the most learnable annotation and the least learnable annotation for each parser under the Accuracy-Learnability measure. The accuracies range between 2.5-8.3%, which correspond to 22.2-35.3% error reduction. Table 4 also shows the the average performance gain from selecting each of the three empirically preferred annotations. These gains are substantial and yield error reductions that range between 3.7-19.8%, 2.4-4.8% and 7.4-15.3% for Coordinations, NPs and PPs respectively. Moreover, the gains are additive. That is, selecting all three of the empirically preferred annotations results in a gain similar to the sum of the average gains in the individual structures.

	Struct.	*DMV*	*MST*	*Clear*	S_u	*N.D.*	Err. Red.
Avg. Per. Diff.	Coord.	1.3%	1.2%	2.1%	1.6%	0.9%	**3.7-19.8%**
	NP	1.6%	0.2%	0.3%	0.5%	0.2%	**2.4-4.8%**
	PP	2.6%	1.6%	1.1%	1.0%	0.9%	**7.4-15.3%**
Best – Worst		8.3%	3.4%	4.2%	3.4%	2.5%	**22.2-35.3%**
Avg. Per.		66.2%	90.1%	90.2%	89.2%	90.4%	—

Table 4: The average performance gain incurred by selecting the empirically preferred annotations for the VSS's for which a unanimous bias is found. The last column is the error reduction range. The last row shows the mean attachment score of each parser when averaging over all schemes. The row before shows the difference between the lowest scoring and the highest scoring scheme for each parser. Annotation abbreviations (see Section 3): Coord. – Coordinations, NP – Noun Phrases, PP – Prepositional Phrases. Parser names are taken from Table 3.

Another natural question to ask is whether there is a single scheme that receives the highest score in all settings. We find that in fact this is the case. Figure 4 shows this scheme. The obtained scheme does not exactly match any of the commonly used annotation schemes, although it closely resembles that of (Collins 1999), differing only in the annotation of noun sequences. We note that since we addressed a particular set of VSS's, the winning scheme presented here is optimal only with respect to this selection.

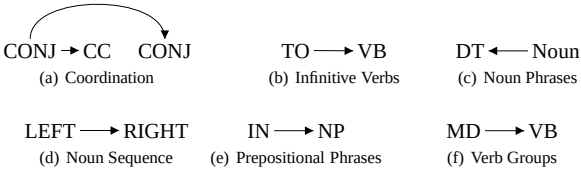


Figure 4: The scheme that receives the highest score under all settings. Annotation abbreviations are taken from Table 3.

Correlation between Settings. We aim to show that our results are independent of the setting, and can therefore be seen as reflecting underlying phenomena. The parsers and the specific learnability measures can thus be seen as proxies by which these phenomena are observed.

In each setting (i.e., parser + learnability measure), we sort the different schemes according to their learnability (a total of 2^k values per ordering). The ten different settings (5 parsers $\times$ 2 learnability measures) yield ten relative orderings. To assess their similarity we compute the Kendall rank correlation coefficient (Kendall 1938)[9] between each pair of relative orderings ($\binom{10}{2} = 45$ pairs). The coefficient receives values in $[-1, 1]$, where 1 indicates equality, 0 indicates no correlation, and -1 indicates anticorrelation. We also compute a significance p-value, which is the probability for obtaining a given correlation at random (Abdi 2007).

Results show that the relative orderings obtained in the different settings are very much in concordance. The obtained Kendall τ correlation coefficients range between $(0.46, 0.88)$. Interestingly, when excluding DMV, results are even more significant (correlation in $(0.64, 0.88)$). This corresponds to p-values smaller than 10^{-7} and smaller than 10^{-13} if we exclude DMV.

[9]This is a commonly used measure in NLP (Lapata 2006; Brody and Kantor 2011).

Relation between Learnability Measures. In order to explore the relations between the two learnability measures, we focused on pairs of orderings that use the same parser, but different learnability measures ($|P| = 5$ pairs). The Kendall τ values in this case range between (0.75, 0.82), which corresponds to p-values $< 10^{-18}$.

Despite the high correlation between the measures, the biases discovered under the Accuracy-Learnability measure are stronger than the ones discovered under the Rate-Learnability measure. This demonstrates the somewhat different perspectives obtained by using different definitions of learnability.

6 Discussion

6.1 Syntactic Selection in a Wider Context

This paper presents a methodology for syntactic selection using learnability. The use of learnability is justified both for theoretical (see Section 7) and practical reasons, as it has direct implications on parsing technology. Namely, it is advantageous to train parsers on schemes that are inherently more learnable.

In the following we define a different, simplified empirical measure for syntactic selection and show that it correlates with learnability. The proposed measure is conceptually simpler than learnability and can therefore be used to partially explain the learnability results. However, it will be argued that learnability has several advantages over it as an empirical measure for syntactic selection.

We define the *predictability* of a scheme as minus the entropy of a parent given its child (unlike learnability, *predictability* is not defined with respect to a parser). We represent a word (x) as its POS tag, and a parent ($Pa(x)$) as the conjunction of its POS tag and the direction of the parent relative to the child (left or right). Concretely (P denotes the set of POS tags):

$$predictability = -H(Pa(x)|x) =$$

$$\sum_{x \in P} \sum_{\substack{Pa(x) \in \\ P \times \{L,R\}}} Pr(Pa(x)|x) \cdot \log Pr(Pa(x)|x)$$

Intuitively, *predictability* measures how easy it is to predict a word's head. On the face of it, higher *predictability* is likely to imply higher learnability. For example, if predictability is very high (i.e., the entropy is very low), words generally determine their parents, which facilitates learning. The opposite case, when predictability is very low, occurs when given a word, any other word is equally probable to serve as its parent. It is likely that such a scheme would be hard to learn.

We repeated the experiments described in Section 4, this time using predictability instead of learnability[10]. Results show that in the three structures where a unanimous learnability bias is found, a strong predictability bias is also found. Predictability yields similar results to learnability in the infinitive verb structure as well, both showing no strong bias. However, in the two other structures results diverge. In noun sequences, predictability shows a strong bias towards the left noun, while learnability showed a weaker trend (with no unanimous bias). In verb groups, a strong predictability bias is found in the opposite direction to the non-unanimous one found with learnability.

In addition, we derive a relative ordering of the different schemes (see Section 5.1). We compare this ordering to each of the orderings obtained in the learnability experiments. The obtained Kendall τ values range between (0.38, 0.66), which corresponds to p-values $< 10^{-4}$.

[10] Note that this time there is only one setting.

Predictability is a simple measure to understand and compute. The fact that it correlates with learnability can provide a partial explanation to the learnability results. However, it has several disadvantages compared to learnability. First, learnability relates directly to parsing technology, as parsers trained on more learnable schemes are likely to obtain higher results. Second, learnability is better motivated theoretically – it has been used extensively as a deciding factor in both linguistics and cognitive science (see Section 7). Third, predictability only quantifies a specific aspect of an annotation scheme (namely the POS tag and direction of the parent relative to its child), while parsers tend to take into account many other factors. These factors are captured by learnability.

Looking at our results, we observe that while correlations between predictability and learnability orderings are relatively high (mean Kendall τ value 0.51), they are generally lower than the correlations between the different settings of our learnability experiments (mean Kendall τ value 0.67). We conclude that predictability does give partial explanation to our results, but that further research is required in order to fully comprehend why exactly are some schemes more learnable than others.

6.2 The Methodology

Our methodology is designed for deciding between several alternatives, each having equal a-priori plausibility. It is therefore applicable for deciding between alternative annotations in VSS's.

Although we compare performance against different test sets, we find the comparison meaningful. Presumably, had there been no preference to either of the annotations, the performance on all these data sets should have been equivalent. Our experiments show that this is usually not the case, and by this reveal a non-trivial property of both the parser and, in those cases where the bias is unanimous, of the structures in question.

The consistent results obtained across five parsers using two learnability measures, which are in turn consistent with the results of a parser-independent predictability experiment, demonstrate the robustness of our results. However, it is still possible that a future parser will exhibit different patterns. Such a parser would very likely be fundamentally different, in some way, from the set of parsers used in this work. Our methodology can thus be used to discover an interplay between parser families and their empirically preferred annotations, an interesting topic in its own right.

Finally, we remark that learnability cannot by itself be used as a criterion for the quality of a scheme. For example, consider the simple right-branching scheme, where each word receives the word to its right as its head. It is trivial to learn despite its inferiority as a dependency scheme. We address this issue by applying our methodology only to compare between annotations that aim to represent the same structure and that were proposed as valid dependency annotation schemes. All considered schemes are derived by combining annotations to VSS's that were proposed in the literature (see Section 3). It is exactly because of the lack of consensus with regard to these structures that applying a complementary criterion, such as learnability, is required.

7 Related Work

7.1 Varying Syntactic Structures

The exact formal manner in which syntax should be represented has been the subject of endless debates. The diversity of approaches yielded a variety of annotation schemes for encoding similar structures.

Representational variation can be seen in virtually any formalism for syntactic annotation. In the field of POS tagging, the Brown Corpus (Francis 1964), the Penn Treebank (PTB) (Marcus et al.

1993), the British National Corpus (BNC) (Aston and Burnard 1998) and the SUSANNE corpus (Sampson 1995) all proposed different schemes for representing grammatical categories. Another example is the different annotation schemes used for noun compounds (Nastase and Szpakowicz 2003; Moldovan et al. 2004). In the field of constituency annotation, (Marcus et al. 1993; Sampson 1995; Kim et al. 2003) vary in the details of their representation of English syntax. Variation in dependency annotation, the focus of this paper, was discussed in (Ivanova et al. 2012) and is described in detail in Section 3. While these examples are all taken from English, variation is found in any language for which sufficient resources are available (Zeman et al. 2012).

Many previous works addressed the difficulties imposed by the lack of established standards for syntactic representation. Jiang and Liu (2009) adapted statistical tools trained with one annotation standard to another. Other works proposed to normalize the different representations into a standard scheme (Ide and Bunt 2010; Zeman et al. 2012). Parsing evaluation is also highly affected by VSS's. Schwartz et al. (2011) suggested Neutral Edge Direction (NED), an evaluation measure for unsupervised dependency parsing that accepts more than one plausible annotation for dependency VSS's. Tsarfaty et al. (2011) suggested a new evaluation measure for supervised dependency parsing to address representational variation. The measure is based on tree edit distance. Tsarfaty et al. (2012) extended this measure for comparing between annotations from different formalisms.

The emphasis of all the above works was mainly to overcome the problems incurred by the lack of standard, and not to select the most advantageous annotation according to some empirical criterion. In contrast, other works addressed the advantages some schemes have over their alternatives, and selected a scheme which best suited their needs.

One of the motivations behind the LTH dependency scheme (Johansson and Nugues 2007) was to facilitate semantic-role-labeling (SRL). They showed that an SRL tool that used their scheme performed better than a tool that used an alternative dependency scheme. While their method provides empirical reasons for using the LTH scheme, our work has a few advantages: first, our methodology examines each VSS individually, while they only compared an annotation scheme as a whole; second, while they performed the comparison on a single (basic) SRL tool, we compared the schemes on five different parsers (four of them state-of-the-art) using two definitions of learnability. Stanford dependencies (de Marneffe et al. 2006) were also designed using empirical considerations, namely to facilitate information extraction. However, they did not attempt to propose a methodology for syntactic selection.

Nilsson et al. (2006) modified the gold standard dependency annotations of two VSS's in order to improve parsing accuracy. They were able to improve performance by training a parser on a transformed corpus, parsing, and re-transforming the induced parse. While their work evaluated against a fixed gold standard, our work provides a methodology for designing an optimal gold standard with respect to learnability considerations. Furthermore, while they experimented with a single parser[11], our experiments use five parsers of different types and two learnability measures. As a result, their findings may be parser-specific, while our consistent results reveal a property of the scheme itself. Therefore, our results are directly applicable to annotation design. Last, our work is more extensive in terms of the number of examined structures (six vs. two). Kübler (2005) and Maier (2006) conducted similar experiments in constituency parsing.

[11] They experimented with two variants of the same parser.

7.2 Learnability

The notion that simpler or more *learnable* structures should be preferred is a recurring theme, both in theoretical linguistics (Chomsky 2006; Clark 2010) and more generally in the discussion of representations in cognitive science (Chater and Vitányi 2003). In the context of language learning, learnability refers to the question of what biases are required in order to learn a language, and in particular its grammar (Pinker 1989). In formal linguistics, learnability using distributional methods has been used as an important consideration in designing the phrase structure formalism (Chomsky 2006).

In Machine Learning, the term learnability refers to the question of whether, under certain assumptions, an underlying hypothesis may be learned given sufficient training samples (prominently, PAC-learnability (Valiant 1984)).

An empirical study by Perfors et al. (2011) used learnability considerations to decide between different syntactic formalisms. This line of research bears resemblance to model selection techniques in statistics, which aim to find which *model* best explains a fixed data set. Our work takes a similar direction. However, our methodology assumes the parsers are acceptable models for the given formalism, and tries to find the most suitable *annotation* from a set of a-priori equally likely alternatives. To the best of our knowledge, no previous work has tackled a similar task.

Predictability. Previous works used information theoretic measures to quantify sentence complexity, taking into account its syntactic representation. Hale (2006) explored a similar measure to predictability in the context of context-free-grammar. Hale (2001) and Levy (2008) explored a different measure ("surprisal"). These works demonstrated that their complexity measures correlate with human judgments on sentence comprehension difficulty.

Conclusion and Perspectives

In this paper we showed that selecting between alternative syntactic representations (syntactic selection) has a substantial and predictable effect on parsing performance. We presented a novel learnability-based methodology for syntactic selection and applied it to six central dependency structures that have several alternative annotations. Our methodology produced highly consistent results, and revealed a unanimity among all parsers in three of the structures. We showed that the gain from selecting the empirically preferred annotations is both substantial (error reduction of up to 19.8%) and additive. That is, selecting all three results in an even more accurate parser.

The higher learnability of the preferred annotations can be seen as an indication for their consistency with the rest of the scheme and has direct implications for parsing performance. We therefore suggest using the preferred annotations when designing future dependency schemes.

Future work will include applying our methodology to languages other than English, in order to assess whether the biases discovered in this work generalize cross-linguistically. We also plan to apply it to deciding between alternative annotations in other syntactic formalisms (such as constituency parsing) and in other NLP tasks such as POS tagging and noun-compound annotation.

Acknowledgments

Omri Abend is grateful to the Azrieli Foundation for the award of an Azrieli Fellowship. We would also like to thank Roi Reichart, Shai Shalev-Shwartz and Valentin I. Spitkovsky for their useful comments. We would also like to thank Richard Johansson for providing us with the code for the *pennconvertor* script.

References

Abdi, H. (2007). Kendall rank correlation. In Salkind, N. J., editor, *Encyclopedia of Measurement and Statistics*, pages 508–510. SAGE. 10

Aston, G. and Burnard, L. (1998). *The BNC handbook. Exploring the British National Corpus with SARA*. Edinburgh University Press. 13

Blunsom, P. and Cohn, T. (2010). Unsupervised induction of tree substitution grammars for dependency parsing. In *Proc. of EMNLP*. 7

Bosco, C. and Lombardo, V. (2004). Dependency and relational structure in treebank annotation. In *Proc. of the Coling Workshop on Recent Advances in Dependency Grammar*. 6

Brody, S. and Kantor, P. (2011). Automatic assessment of coverage quality in intelligence reports. In *Proc. of ACL-HLT*. 10

Chater, N. and Vitányi, P. (2003). Simplicity: a unifying principle in cognitive science. *Trends in Cognitive Science*, 7:19–22. 14

Choi, J. D. and Nicolov, N. (2009). K-best, locally pruned, transition-based dependency parsing using robust risk minimization. In Nicolov, N., Angelova, G., and Mitkov, R., editors, *Recent Advances in Natural Language Processing V*, volume 309 of *Current Issues in Linguistic Theory*, pages 205–216. John Benjamins, Amsterdam & Philadelphia. 7, 9

Chomsky, N. (2006). *Language and Mind*. Cambridge University Press, third edition. 3, 14

Clark, A. (2010). Three learnable models for the description of language. In *LATA*, pages 16–31. 14

Cohen, S. B. and Smith, N. A. (2009). Shared logistic normal distributions for soft parameter tying. In *Proc. of HLT-NAACL*. 7

Collins, M. J. (1999). *Head-driven statistical models for natural language parsing*. PhD thesis, University of Pennsylvania, Philadelphia. 2, 6, 10

Crammer, K. and Singer, Y. (2003). Ultraconservative online algorithms for multiclass problems. *JMLR*, 3:951–991. 7

de Marneffe, M. C., Maccartney, B., and Manning, C. D. (2006). Generating typed dependency parses from phrase structure parses. In *Proc. of LREC*. 13

Dredze, M., Blitzer, J., Talukdar, P. P., Ganchev, K., Graça, J. V., and Pereira, F. (2007). Frustratingly hard domain adaptation for dependency parsing. In *Proc. of the CoNLL 2007 Shared Task. EMNLP-CoNLL*. 6

Francis, W. (1964). *A standard sample of present-day English for use with digital computers*. Brown University. 12

Goldberg, Y. and Elhadad, M. (2010). An efficient algorithm for easy-first non-directional dependency parsing. In *Proc. of HLT-NAACL*. 7, 9

Hale, J. (2001). A probabilistic earley parser as a psycholinguistic model. In *Proc. of NAACL*. 14

Hale, J. (2006). Uncertainty about the rest of the sentence. *Cognitive Science*, 30:643–672. 14

Headden III, W. P., Johnson, M., and McClosky, D. (2009). Improving unsupervised dependency parsing with richer contexts and smoothing. In *Proc. of HLT-NAACL*. 7

Ide, N. and Bunt, H. (2010). Anatomy of annotation schemes: mapping to GrAF. In *Proc. of the Fourth Linguistic Annotation Workshop (LAW)*. 13

Ivanova, A., Oepen, S., Øvrelid, L., and Flickinger, D. (2012). Who did what to whom? a contrastive study of syntacto-semantic dependencies. In *Proc. of the Sixth Linguistic Annotation Workshop (LAW)*. 13

Jiang, W. and Liu, Q. (2009). Automatic adaptation of annotation standards for dependency parsing: using projected treebank as source corpus. In *Proc. of IWPT*. 13

Johansson, R. and Nugues, P. (2007). Extended constituent-to-dependency conversion for English. In *Proc. of NODALIDA*. 2, 6, 8, 13

Kendall, M. G. (1938). A new measure of rank correlation. *Biometrika*, 30:81–93. 10

Kim, J.-D., Ohta, T., Tateisi, Y., and ichi Tsujii, J. (2003). GENIA corpus – a semantically annotated corpus for bio-textmining. In *ISMB (Supplement of Bioinformatics)*. 13

Klein, D. and Manning, C. (2004). Corpus-based induction of syntactic structure: Models of dependency and constituency. In *Proc. of ACL*. 7, 9

Kübler, S. (2005). How do treebank annotation schemes influence parsing results? or how not to compare apples and oranges. In *Proc. of RANLP*. 13

Kübler, S., McDonald, R., and Nivre, J. (2009). *Dependency Parsing*. Morgan And Claypool Publishers. 5, 8

Lapata, M. (2006). Automatic evaluation of information ordering: Kendall's tau. *Computational Linguistics*, 32(4):471–484. 10

Levy, R. (2008). Expectation-based syntactic comprehension. *Cognition*, 106(3):1126–1177. 14

Maier, W. (2006). Annotation schemes and their influence on parsing results. In *Proc. of the ACL Student Research Workshop*. 13

Marcus, M., Santorini, B., and Marcinkiewicz, M. A. (1993). Building a large annotated corpus of English: The Penn Treebank. *Computational Linguistics*, 19:313–330. 2, 8, 12, 13

McDonald, R., Pereira, F., Ribarov, K., and Hajič, J. (2005). Non-projective dependency parsing using spanning tree algorithms. In *Proc. of HLT-EMNLP*. 2, 7, 9

Moldovan, D., Badulescu, A., Tatu, M., Antohe, D., and Girju, R. (2004). Models for the semantic classification of noun phrases. In *Proc. of the HLT-NAACL Workshop on Computational Lexical Semantics*. 13

Nastase, V. and Szpakowicz, S. (2003). Exploring noun-modifier semantic relations. In *Proc. of IWCS*. 13

Nelson, G., Wallis, S., and Aarts, B. (2002). *Exploring natural language: working with the British component of the international corpus of English*. John Benjamins Pub. Co. 2

Nilsson, J., Nivre, J., and Hall, J. (2006). Graph transformations in data-driven dependency parsing. In *Proc. of ACL*. 6, 13

Nivre, J. (2009). Non-projective dependency parsing in expected linear time. In *Proc. of ACL-IJCNLP*. 7, 9

Nivre, J., Hall, J., and Nilsson, J. (2006). Maltparser: A data-driven parser-generator for dependency parsing. In *Proc. of LREC*. 7

Perfors, A., Tenenbaum, J. B., and Regier, T. (2011). The learnability of abstract syntactic principles. *Cognition*, 118(3):306–338. 14

Pinker, S. (1989). *Learnability and Cognition*. MIT Press. 14

Rambow, O., Creswell, C., Szekely, R., Tauber, H., and Walker, M. (2002). A dependency treebank for English. In *Proc. of LREC*. 2, 6

Sampson, G. (1995). *English for the Computer: The Susanne Corpus and Analytic Scheme*. Clarendon Press. 2, 13

Schwartz, R., Abend, O., Reichart, R., and Rappoport, A. (2011). Neutralizing linguistically problematic annotations in unsupervised dependency parsing evaluation. In *Proc. of ACL-HLT*. 2, 13

Spitkovsky, V. I., Alshawi, H., and Jurafsky, D. (2011). Punctuation: Making a point in unsupervised dependency parsing. In *Proc. of CoNLL*. 7

Tsarfaty, R., Nivre, J., and Andersson, E. (2011). Evaluating dependency parsing: Robust and heuristics-free cross-annotation evaluation. In *Proc. of EMNLP*. 13

Tsarfaty, R., Nivre, J., and Andersson, E. (2012). Cross-framework evaluation for statistical parsing. In *Proc. of EACL*. 13

Valiant, L. G. (1984). A theory of the learnable. *Communications of the ACM*, 27(11):1134–1142. 14

Yamada, H. and Matsumoto, Y. (2003). Statistical dependency analysis with support vector machines. In *Proc. of IWPT*. 2, 6

Zeman, D., Mareček, D., Popel, M., Ramasamy, L., Štěpánek, J., Žabokrtský, Z., and Hajič, J. (2012). HamleDT: To parse or not to parse? In *Proc. of LREC*. 13

Zhang, T., Damerau, F., and Johnson, D. (2002). Text chunking based on a generalization of winnow. *JMLR*, 2:615–637. 7

Improving Supervised Sense Disambiguation with Web-Scale Selectors

H. Andrew Schwartz[1] *Fernando Gomez*[2] *Lyle H. Ungar*[1]
(1) University of Pennsylvania, Philadelphia, PA USA
(2) University of Central Florida, Orlando, FL USA
hansens@seas.upenn.edu, gomez@eecs.ucf.edu, ungar@cis.upenn.edu

ABSTRACT
This paper introduces a method to improve supervised word sense disambiguation performance by including a new class of features which leverage contextual information from large unannotated corpora. This new feature class, *selectors*, contains words that appear in other corpora with the same local context as a given lexical instance. We show that support vector sense classifiers trained with selectors achieve higher accuracy than those trained only with standard features, producing error reductions of 15.4% and 6.9% on standard coarse-grained and fine-grained disambiguation tasks respectively. Furthermore, we find an error reduction of 9.3% when including selectors for the classification step of named-entity recognition over a representative sample of OntoNotes. These significant improvements come free of any human annotation cost, only requiring unlabeled Web-Scale corpora.

KEYWORDS: word sense disambiguation, lexical semantics, semi-supervised learning.

Proceedings of COLING 2012: Technical Papers, pages 2423–2440,
COLING 2012, Mumbai, December 2012.

1 Introduction

Supervised word sense disambiguation (*WSD*) systems often rely directly on the local contexts in which target words appear. For example, the state-of-the-art system of Zhong et al. (2008) uses features based on collocations centered on the target word. Models relying on such features do well with copious amounts of training data, but they are prone to errors when the local context of a test instance differs from local context observed during training. Consider the sentences below.

1. *The workers loaded the **port** onto the ship this morning.*
2. *She purchased a couple of bottles of **port** from the store.*
3. *The couple enjoyed their richly-flavored **port**.*

Though referring to the same sense of 'port', "a sweet dark-red dessert wine" (Miller et al., 1993), it is difficult to connect any two instances based on local context; the parts-of-speech even differ substantially. Models for disambiguation can benefit from the addition of a feature that does not rely directly on the local context.

We present a new class of features which encodes an abstraction of a word's context, rather than encoding contents of the local context itself. We refer to this new feature class as *selectors*, borrowing the term from an approach to knowledge-based (unsupervised) word sense disambiguation which uses the idea of searching for words that share the same context (Lin, 1997; Schwartz and Gomez, 2008). More precisely, selectors are words that show up in the same local context as a given instance of another word. For example, selectors for 'port' in sentence 1 might be 'bottles', 'crates', 'passengers', 'wine', 'luggage', etc. Considering that the other sentences may share some of the same selectors such as 'bottles' or 'wine', one can see how this abstraction of context to selectors can be beneficial. Figure 1 demonstrates mapping the context from one instance to selectors, which match the selectors of another instance. In this sense, it is the contexts (or word instances) that have selectors rather than the words themselves.

The contribution of this paper is the introduction of a novel and effective type of feature that improves *WSD* accuracy at no cost in human annotation. Rather than requiring more examples of labeled context to match a given test instance, we need only to match against an orders-of-magnitude-larger unlabeled set of data. Because *selectors* leverage unlabeled data, their inclusion in a supervised system constitutes semi-supervised learning.

The paper proceeds with a discussion of related work in semi-supervised *WSD* and the use of web-scale data in language processing (Section 2). Then, we present our approach to acquiring selectors as features from n-grams, and show how we translate selectors into features (Section 3). The effectiveness of selectors is evaluated within supervised word sense disambiguation classifiers over the SemEval-2007 Task 17 (Pradhan et al., 2007), Senseval 3 English Lexical Sample (Mihalcea et al., 2004), and OntoNotes 4 (Weischedel et al., 2011) (Section 4). We also test selectors as features for the classification step of named-entity recognition over a representative sample of OntoNotes. Lastly, we discuss the robustness of selectors as features by inspecting actual instances from our experimental corpus (Section 5).

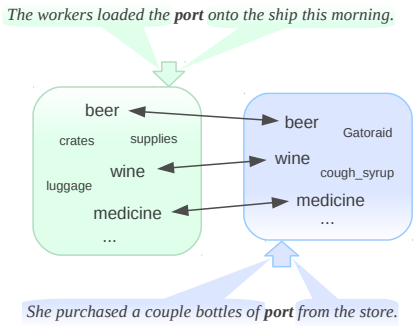


Figure 1: Word instances which do not share context can share selectors.

2 Related Work

The idea of improving a supervised classifier by utilizing unlabeled data has been investigated at different levels. For example, other approaches to disambiguation have used bootstrapped samples (Yarowsky, 1995; Mihalcea and Moldovan, 1999; Mihalcea, 2004; Pham et al., 2005), Wikipedia concepts (Mihalcea, 2007), or parallel corpora (Chan et al., 2007). Most of these approaches, which are considered semi-supervised learning (Zhu, 2008), exploit some facet of unannotated text to collect more training instances. Rather than produce more training instances, we introduce a method to leverage massive unlabeled corpora to create a richer and robust set of features.

One can contrast *selectors* with clusters of words formed via context or distributional similarity (for seminal examples see (Brown et al., 1992; Pereira et al., 1993; Lin, 1998; Schütze, 1998; Pantel and Lin, 2002)). Distributional clusters are made up of words that appear in similar contexts to each other, whereas selectors are words which show up in the specific context of a single instance. In other words, selectors are instance-specific while distributional clusters are created based on observing many instances of context. This key difference should become more clear when we present our method of acquiring selectors.

The traditional use of selectors is in knowledge-based word sense disambiguation systems, not utilizing training data. In Lin (1997), dependency relationships over a small corpus were used to find noun selectors. We previously extended this to the Web, treating context as surrounding text and introduced the ideas of acquiring selectors for additional parts-of-speech as well as for words in context in addition to the target word (Schwartz and Gomez, 2008, 2009). Similar to selectional preferences (Resnik, 1997), *selectors* essentially indicate the types of concepts expected in a given syntactic or grammatical position. In these knowledge-based approaches, disambiguation is performed by computing the semantic distance between *selectors* and senses of the target word. These approaches rely on both a knowledge source such as WordNet (Miller et al., 1993) and a semantic distance metric. In contrast, in the current approach we do not need such a knowledge source or similarity judgments, and since our approach is data-driven, selectors function as an abstraction of word instance context rather than as a list of semantically similar words. Our current goal is to get the most out of supervised training data by leveraging unannotated data via selectors (no use of a knowledge-base or similarity metrics). Consequently,

our system achieves state-of-the-art results in line with top supervised systems while our earlier knowledge-based approaches produce results in line with systems not utilizing training data.

A couple previous works have integrated unannotated data as features into supervised disambiguation systems. Dligach and Palmer (2008) used *dynamic dependency neighbors*, a feature encoding verbs with the same object, according to a dependency parsed corpus, as a given target verb in a verb WSD task. Besides our method not being limited to verbs, selectors are much more specific than dependency neighbors; They are found by matching a larger context and from a much larger, web-scale, dataset. Cárcamo et al. (2008) adapt the predominant sense method of McCarthy et al. (2004) to find the best sense choice for a word instance rather than it's most common sense over a corpus. Yuret (2007) leveraged web-scale data to acquire probability distributions of *substitutes* being within the same context as target instances. Unlike selectors which are open-ended, substitutes were chosen from an *a priori* word list derived from thesauri, and contextual part-of-speech was not considered. Additionally, the substitutes' probability distribution itself was the entire feature set, rather than used to supplement an existing feature set, and the resulting accuracies were lower than those we find with selectors.

Our approach utilizes web-scale N-grams, a source of unlabeled data which has previously been used for many other supervised lexico-semantic tasks including delimiting named entities, preposition selection, spelling correction, search query processing, adjective ordering, verb POS disambiguation, and noun compound bracketing (Downey et al., 2007; Bergsma et al., 2009; Huang et al., 2010; Bergsma et al., 2010). All of these systems utilized n-grams to find frequency information for specific n-grams. In contrast, we use the n-grams as a source for acquiring sets of lexical data (selectors), where we search with context and ask for the missing piece rather than search for a complete n-grams. We believe this is the first work to use web-scale N-grams as a source for selectors; motivation for using this source is discussed in the next section.

3 Acquiring Selectors

A selector is a word which appears in the same local context as a given instance of a *focus word*. For example, in the sentence below, with 'port' as the *focus word*, one might find selectors such as 'bottles', 'cargo', 'crates', 'wine', 'passengers', or 'supplies'.

> *The workers loaded the* ***port*** *onto the ship last night.*

More formally, for a given word instance, w_i, selectors are found based on the particular context of w_i. What defines the context may vary from syntactic or dependency relations (i.e., other nouns which are objects of the verb 'loaded') to simple sequences of tokens (e.g.,finding words that fill in the blank in "The workers loaded the ___ onto the ship last night.").

3.1 Approach

We find selectors by searching for sequences of tokens in the Google N-grams version 2, which contains 4.1 billion n-grams that were automatically part-of-speech tagged (Lin et al., 2010). The primary reason we chose web-scale N-grams as a source is because it has become difficult to get selectors via search engines.[1] Still, using web-scale n-grams for context searches has advantages: there is a decent likelihood of finding selectors for a given instance, the search

[1]The Web search engines which support wildcard queries no longer run public APIs or allow scripted access.

*Workers loaded the **port** onto the ship last night.*
workers loaded $\langle det \rangle$? ($\langle noun \rangle$+) onto
loaded $\langle det \rangle$? ($\langle noun \rangle$+) onto
($\langle noun \rangle$+) onto $\langle det \rangle$ ship last

---

*My objective was to **fight** as a mother for what I hold dearest.*
was to ($\langle verb \rangle$+) as $\langle det \rangle$
to ($\langle verb \rangle$+) as $\langle det \rangle$? mother
objective was to ($\langle verb \rangle$+)

---

*The new economy in the US depends heavily, for one thing, on a deep foundation of basic scientific research, which comes up with revolutionary **products** like genetically modified foods.*
with revolutionary ($\langle noun \rangle$+) like genetically
revolutionary ($\langle noun \rangle$+) like
($\langle noun \rangle$+) like genetically modified

---

*...which comes up with **revolutionary** products like...*
up with ($\langle adj \rangle$+) products like
up with ($\langle adj \rangle$+) products
with ($\langle adj \rangle$+) products

Table 1: Example search sequences produced for the given focus word focus word (***in bold***) and context (*in italic*).'()' surrounds the *focus word*, '?' implies optional match and '+' allows multiple matches. The bottom three examples are from our experimental corpus.

process is offline, this version of the Google N-grams provides part-of-speech information, and they have been shown helpful for other lexico-semantic tasks (Bergsma et al., 2009; Huang et al., 2010; Bergsma et al., 2010). On the downside, because the Google N-grams are at most 5-grams, the selectors can only be found using a relatively small context. – up to 4 tokens. For this first investigation of selectors as features we think this trade-off is worthwhile.

We search the n-grams by constructing 3 to 5 token sequences consisting of words or part-of-speech (*POS*) tags. Determiners, conjunctions, possessives, and symbols in the sequence are replaced with their *POS* tag, and determiners are also marked as optional if they do not begin or end the phrase. The slot of the *focus word*, the word for which selectors are being acquired, is restricted by *POS* and permitted to match multi-word phrases (taking the head-word as the selector in such cases). Examples of search sequences are given in Table 1.

Searching based on all sequences can be expensive in terms of disk IO, so the sequences are sorted such that the process can be halted once enough selectors have been found. In particular, we define four *criteria* of informative value for a given sequence seq:

1. the number of tokens in seq:
$length(seq) = \frac{|tokens(seq)|}{max\_tokens}$
2. the number of content words (noun, verbs, adjective, or adverbs):
$nvar(seq) = \frac{|nouns(seq)|+|verbs(seq)|+|adjectives(seq)|+|adverbs(seq)|}{max\_tokens}$

3. the distance from the focus word to the center: $center(seq) = 1 - \frac{|before(seq) - after(seq)|}{|tokens(seq)|}$

4. if the *focus word* is an edge of seq:
$$\neg edge(seq) = \begin{cases} 0, & \text{if focus word is at front or back} \\ 1, & \text{otherwise} \end{cases}$$

where $max\_tokens = 5$, the maximum number of tokens in a sequence, and $before$ / $after$ are the number of token before and after the focus word. The overall informative value is defined as the sum of weighted (α_i) *criteria* ($c_{1..4} = [length(seq), nvar(seq), center(seq), \neg edge(seq)]$):

$$info(seq) = \sum_{i=1}^{4} \alpha_i c_i$$

Next, we iterate through the list of sorted search sequences in order to aggregate selector(s) frequencies. During aggregation, the selector frequencies are normalized and weighted by $info(seq)$:

$$score(s) = \sum_{seq \in seqs} info(seq) * \frac{freq(s)}{\max_{s' \in sels(seq)} freq(s')}$$

where $sels(seq)$ is the set of selectors found when searching with sequence seq. This favors both selectors occurring with multiple sequences as well as those found based on a more informative context. In practice we break the aggregation loop one iteration after acquiring a soft minimum (k) of selectors to improve runtime. [2]

3.2 Selectors as Features.

We have described acquisition of selectors for an arbitrary *focus word* instance. In order to use selectors as features, we acquire selectors for all target words (words being disambiguated) and encode the top k, according to $score(s)$, as binary features. We selected $k = 50$ as well as weightings $\alpha_{1..4} = [0.2, 0.2, 0.1, 0.5]$ after cross-validating over the *training set* (defined in Section 4).

4 Experiments

We evaluate whether the *selector* class of features can benefit *WSD* classifiers above and beyond a standard set of features in a variety of datasets and situations. Supervised classifiers are trained with and without utilization of *selectors* and we record a simple accuracy of $\frac{|correct\_instances|}{|all\_instances|} * 100$ of the testing data.[3] In particular, we use support vector classifiers implemented with Scikit-learn (Pedregosa et al., 2011) with a radial basis kernel and other parameters set via 5-fold cross-validation over the training set. As a standard point of comparison, *most frequent sense* (MFS) accuracy is also reported, indicating the testing accuracy if the system always predicted the most common sense according to the training data. As often noted, state-of-the-art supervised systems often perform just above the MFS (Navigli et al., 2007; Pradhan et al., 2007).

[2] An implementation of this method is included in supplementary data.

[3] $accuracy = precision = recall$ under the standard (SemEval) definition of precision and recall for WSD, and because we attempt all instances of our samples.

4.1 Data Sets

We test *selectors* over three sense-annotated corpora. For our primary corpus, we use the SemEval-2007 Task 17: Lexical Sample (Pradhan et al., 2007) (results in sections 4.3.1 and 4.3.3). This corpus is an early selection from the Wall Street Journal portion of OntoNotes (Weischedel et al., 2011), and contains coarse-grained noun and verb senses. We also experiment over the Senseval-3 English Lexical Sample data(Mihalcea et al., 2004), containing fined-grained noun, verb, and adjective sense annotations over selections of the British National Corpus (Clear, 1993) (section 4.3.2). The inclusion of adjectives, fine-grained senses, and difference in corpus gives us a more robust evaluation of selectors. Lastly, we experiment with random samples over portions of the full Ontonotes 4.0 in order to test on out-of-domain data and to examine if selectors help for another lexico-semantic task: named-entity classification. Details about the OntoNotes test sets are included when discussing those results (sections 4.3.4 and 4.3.5).

4.2 Baseline Features

As a consistent baseline throughout our experiments, we use the same features as Zhong et al. (2008)'s state-of-the-art system, first explored by Lee and Ng (2002). These features give the best published results that we are aware of over the Wall Street Journal portion of OntoNotes, plus they are the common denominator in many high-performance supervised WSD systems (Cai et al., 2007; Chan et al., 2007; Zhong et al., 2008).

- **collocations** (*coll*). Tokens relative to the target, denoted $c_{i,j}$, starting at i; ending at j.
  1-grams: $c_{-1,-1}$, $c_{+1,+1}$, $c_{-2,-2}$, $c_{+2,+2}$,
  2-grams: $c_{-2,-1}$, $c_{+1,+2}$,
  3-grams: $c_{-3,-1}$, $c_{+1,+3}$, $c_{-1,+1}$,
  4-grams: $c_{-2,+1}$, $c_{-1,+2}$
- **parts-of-speech** (*pos*). The part-of-speech for the following words relative to the target word: $p_{-3}, p_{-2}, p_{-1}, p_0, p_{+1}, p_{+2}, p_{+3}$ (0 is the target word).
- **surrounding words** (*surr*). The bag-of-words from the current, previous, and next sentence.

4.3 Results

4.3.1 SemEval-2007

Table 2 shows the results with and without selectors over the SemEval-2007 corpus. We see that including selectors improves performance over a state-of-the-art set of features with a significant ($p < 0.01$) error reduction of 15.4%.[4] This puts our system just behind the top system participating in SemEval-2007, NUS-ML (Cai et al., 2007), which achieved an accuracy of 88.7. Moreover, we see improvements from selectors for both nouns and verbs.

Tables 3 and 4 break the results down for each word. Though it is possible for selectors to introduce noise leading to occasional errors, we see that both words with many training instances as well as those with fewer ones can benefit from selectors. We will inspect a couple instances where selectors helped prediction in Section 5.

[4] **error reduction**= $\frac{(1-acc1)-(1-acc2)}{1-acc1}$, where $acc1$ and $acc2$ represent the two accuracies.

	base	**w/ sels**	***mfs***	***tests***
noun	87.9	**91.7**	80.9	*2559*
verb	83.3	**83.7**	76.5	*2292*
both	85.7	**87.9**	78.8	*4851*

Table 2: Classifier accuracies without (**base**) and with the selector class of features (**w/ sels**) over SemEval-2007 Task 17. (*mfs*: accuracy of classifying with the most frequent sense of the training data, *tests*: number of instances in the test set.)

word	**base**	**w/ sels**	***mfs***	***tests***	***trains***
area-n	78.4	83.8	70.3	37	326
authority-n	81.0	81.0	23.8	21	90
base-n	40.0	**70.0**	10.0	20	92
bill-n	98.0	98.0	75.5	102	404
capital-n	96.5	96.5	96.5	57	278
carrier-n	71.4	71.4	71.4	21	111
chance-n	**73.3**	60.0	40.0	15	91
condition-n	79.4	79.4	76.5	34	132
defense-n	42.9	**61.9**	28.6	21	120
development-n	65.5	**79.3**	62.1	29	180
drug-n	89.1	**91.3**	87.0	46	205
effect-n	86.7	**93.3**	76.7	30	178
exchange-n	86.9	86.9	73.8	61	363
future-n	**95.2**	94.5	86.3	146	350
hour-n	89.6	**91.7**	89.6	48	187
job-n	**82.1**	79.5	82.1	39	188
management-n	88.9	**93.3**	71.1	45	284
move-n	97.9	97.9	97.9	47	270
network-n	96.4	**98.2**	90.9	55	152
order-n	91.2	91.2	91.2	57	346
part-n	**91.5**	90.1	66.2	71	481
people-n	90.4	**93.9**	90.4	115	754
plant-n	98.4	98.4	98.4	64	347
point-n	90.7	**93.3**	81.3	150	469
policy-n	97.4	97.4	97.4	39	331
position-n	68.9	**88.9**	46.7	45	268
power-n	85.1	**89.4**	27.7	47	251
president-n	**98.7**	98.3	72.9	177	879
rate-n	88.3	**90.3**	86.2	145	1009
share-n	97.1	**97.7**	97.1	525	2534
source-n	80.0	**88.6**	37.1	35	152
space-n	92.9	**100.0**	78.6	14	67
state-n	79.2	**80.6**	79.2	72	617
system-n	68.6	**72.9**	48.6	70	450
value-n	98.3	98.3	98.3	59	335

Table 3: Classifier accuracies for each noun of the SemEval-2007 Task 17 test set. *trains*: number of training instances.

word	**base**	**w/ sels**	***mfs***	***tests***	***trains***
affect-v	100.0	100.0	100.0	19	45
allow-v	**97.1**	91.4	97.1	35	108
announce-v	100.0	100.0	100.0	20	88
approve-v	**91.7**	83.3	91.7	12	53
ask-v	74.1	**87.9**	51.7	58	348
attempt-v	100.0	100.0	100.0	10	40
avoid-v	100.0	100.0	100.0	16	55
begin-v	66.7	**72.9**	56.2	48	114
believe-v	80.0	**83.6**	78.2	55	202
build-v	73.9	**78.3**	73.9	46	119
buy-v	**80.4**	78.3	76.1	46	164
care-v	42.9	42.9	28.6	7	69
cause-v	100.0	100.0	100.0	47	73
claim-v	80.0	80.0	80.0	15	54
come-v	32.6	**51.2**	23.3	43	186
complain-v	85.7	85.7	85.7	14	32
complete-v	93.8	93.8	93.8	16	42
contribute-v	**83.3**	72.2	50.0	18	35
describe-v	100.0	100.0	100.0	19	57
disclose-v	92.9	92.9	92.9	14	55
do-v	90.2	**93.4**	90.2	61	207
end-v	66.7	**90.5**	52.4	21	135
enjoy-v	57.1	42.9	57.1	14	56
estimate-v	100.0	100.0	100.0	16	74
examine-v	100.0	100.0	100.0	3	26
exist-v	100.0	100.0	100.0	22	52
explain-v	88.9	88.9	88.9	18	85
express-v	100.0	100.0	100.0	10	47
feel-v	68.6	**72.5**	68.6	51	347
find-v	82.1	**85.7**	82.1	28	174
fix-v	50.0	50.0	50.0	2	32
go-v	**70.5**	63.9	45.9	61	244
grant-v	80.0	80.0	80.0	5	19
hold-v	50.0	**54.2**	37.5	24	129
hope-v	100.0	100.0	100.0	33	103
improve-v	100.0	100.0	100.0	16	31
join-v	38.9	38.9	38.9	18	68
keep-v	56.2	**58.8**	56.2	80	260
kill-v	87.5	87.5	87.5	16	111
lead-v	**69.2**	66.7	38.5	39	165
maintain-v	90.0	**100.0**	90.0	10	61
need-v	91.1	91.1	71.4	56	195
negotiate-v	100.0	100.0	100.0	9	25
occur-v	90.9	**95.5**	86.4	22	47
prepare-v	**94.4**	88.9	77.8	18	54
produce-v	75.0	75.0	75.0	44	115
promise-v	75.0	**100**	75.0	8	50
propose-v	85.7	**92.9**	85.7	14	34
prove-v	54.5	**81.8**	68.2	22	49
purchase-v	100	100.0	100.0	15	35
raise-v	29.4	**50.0**	14.7	34	147
recall-v	86.7	86.7	86.7	15	49
receive-v	95.8	95.8	95.8	48	136
regard-v	78.6	78.6	71.4	14	40
remember-v	100.0	100.0	100.0	13	121
remove-v	100.0	100.0	100.0	17	47
replace-v	100.0	100.0	100.0	15	46
report-v	91.4	**94.3**	91.4	35	128
rush-v	100.0	100.0	100.0	7	28
say-v	98.7	98.7	98.7	541	2161
see-v	44.4	**59.3**	44.4	54	158
set-v	47.6	**59.5**	28.6	42	174
start-v	44.7	**52.6**	44.7	38	214
turn-v	51.6	**58.1**	38.7	62	340
work-v	60.5	**67.4**	55.8	43	230

Table 4: Classifier accuracies for each verb of the SemEval-2007 Task 17 test set.

	base	**w/ sels**	***mfs***	***tests***
noun	68.5	**72.1**	54.1	*1766*
verb	72.0	72.4	57.9	*1927*
adjective	49.4	**53.4**	54.7	*148*
all	69.4	**71.5**	56.1	*3841*

Table 5: Sense classifier accuracies without (**base**) and with the selector class of features (**w/ sels**) over Seneval-3: English Lexical Sample. (*mfs*: accuracy of classifying with the most frequent sense of the training data, *tests*: number of instances in the test set.)

4.3.2 Senseval-3

We also tested selectors as features over the Senseval-3 data(Mihalcea et al., 2004) to get a more robust idea of their impact. The instances in this sample come from a difference corpus, the British National Corpus, include fine-grained sense annotations, and a limited number of adjectives.

Examining the results in Table 5, we see an improvement from using selectors over the baseline for all three parts-of-speech. Overall error reductions is 6.9%. Selectors seem to help the most for both nouns and adjectives, but in the case of adjectives we actually see the *mfs* just outperforms the supervised systems. We suspect this is partly due to the average adjective having many more possible senses (10.2, versus 5.8 for nouns and 6.3 for verbs), and one should also keep in mind the small number of adjective examples.

4.3.3 Feature Impact Analysis

Results discussed thus far imply *selectors* are contributing information beyond that of the standard set of features. However, since selectors represent an abstraction of context and the baseline features encode various contextual information, it is possible that all information from certain baseline features is subsumed by selectors. In this experiment, we try to understand the type of information being contributed by selectors by observing accuracies when features are removed.

Table 6 shows accuracy results when building classifiers with all combinations of feature types. For these tests, we used the SemEval-2007 data set, the larger and more recent of the two previously mentioned evaluation data sets. We see a clear benefit from the inclusion of selectors across the board. Interestingly, we see that although *surr* class of features gets the system beyond the *mfs* baseline, it seems to provide more distractions than help once other features are included as well. In fact, our best results come from the combination of collocations, part-of-speech information, and selectors with an accuracy of 88.1.

4.3.4 Out-of-Domain Test Data

It is often noted that *WSD* systems perform poorly on test data from a different domain than that of the training data (Zhong et al., 2008; Agirre et al., 2010). We examine whether selectors keep their benefit when tested on out-of-domain data over a portion of OntoNotes 4.0 (Weischedel et al., 2011). We put together all occurrences of a random selection of 100 nouns and verbs over three portions of OntoNotes: The Wall Street Journal ***(WSJ)***, Xinhua New Agency ***(Xh)***, and Sinorama Magazine ***(Sr)***. The Xinhua and Sinorama corpora correspond to a different *source* of newswire data and a different *genre* (magazine) respectively. As is standard, we used

feature types	accuracy w/o sels	w/ sels	err reduc
coll	86.3	**87.9**	11.7 %
pos	83.8	**86.3**	15.3 %
surr	82.5	**86.8**	24.6 %
coll, pos	86.9	**88.2**	9.9 %
pos, surr	86.0	**87.7**	12.1 %
surr, coll	85.5	**87.4**	13.1 %
coll, pos, surr	85.7	**87.9**	15.4 %
sels alone	-	84.7	-
(***mfs***)	78.8	-	-
mean err reduc	-	-	14.5 %

Table 6: Accuracy of classifier utilizing all combinations of feature types on the SemEval-2007 Task 17 test set. **err reduc** is the error reduction when using selectors. Refer to section 4.2 for feature type identifiers.

	base	**w/ sels**	***mfs***	***tests***
WSJ	82.5	**84.3**	80.7	*166*
Xh	77.1	**78.2**	75.4	*564*
Sr	58.1	58.8	46.8	*816*

Table 7: Accuracy of the classifiers when training on the WSJ, and applying to another source of news (Xh) or another genre of text (Sr) within OntoNotes 4.

samples from sections 02-21 of *WSJ* as training, while samples from section 22 of *WSJ* plus all sections of *Xh* and *Sr* were used for testing. We decided to use OntoNotes because our main testing corpus, SemEval-2007 Task 17, is itself derived from OntoNotes, though it lacked multiple genres of text.

We see from Table 7 that *selectors* still give an improvement in the case of another *source* of newswire. When moving to a more distant domain, such as another *genre*, the improvement still exists though it is no longer significant. The difficulty of the out-of-domain task is exemplified by lower MFS values, which are still based on the most frequent sense in the training data (always WSJ in this case). The results demonstrate relative robustness across minor shifts in domain, and potential for greater success if one combined them with a domain-adaptation technique.

4.3.5 Named Entity Classification

We believe *selectors* can benefit other supervised lexical disambiguation tasks. In this experiment, we seek preliminary evidence for such a belief based on improving the classification step of named entity recognition.

In *named entity classification*, one is given a noun phrase representing an entity with its context, and one attempts to classify the named entity into a variety of classes. We build a classifier which labels entities with one of the 18 classes provided by OntoNotes. We sample 1000 randomly selected sentences from The Wall Street Journal, Xinhua, and Sinorama portions of OntoNotes. The data is divided into training and testing samples:

base	w/ sels	*mfc*	*tests*
85.0	**86.4**	20.2	259

Table 8: Named-entity classifier accuracies without (**base**) and with the selector class of features (**w/ sels**) across a random sample of the WSJ, Xh and Sr portions of OnotNotes. (*mfc*: accuracy of predicting the most frequent named entity class in training data, *tests*: number of instances in the test set.)

- The Wall Street Journal ***(WSJ)***: *sections 02 - 21(train); section 22(test)*
- Xinhua New Agency ***(Xh)***: *sections 0000 - 0209(train); sections 0210 - 0325(test)*
- Sinorama Magazine ***(Sr)***: *sections 1001 - 1059(train); sections 1060 - 1078(test)*

For the WSJ we stick with standard training and test sets, while we divide Xh and Sr corpora similarly. Out of the 1,000 randomly selected sentences across these corpora there are 2,106 total named entity instances: 1,847 training examples and 259 test examples. We find this to be a representative sample of the WSJ, Xh, and Sr portions of OntoNotes [5].

We choose our features by looking at the most common types of features used during the CoNLL-2003 Shared Task in Named Entity Recognition(Tjong Kim Sang and De Meulder, 2003), and more recent developments(Ratinov and Roth, 2009; Finkel and Manning, 2009). To the best of our knowledge state-of-the-art features have not been established for labeling all classes of Named Entities in OntoNotes, though Finkel and Manning use the three most common classes and group the others into a *misc* category.

- **character n-grams.** Character sequences of length 1 to 6.
- **case information.** Case of the first, second, and last letter, as well as an indicator for punctuation.
- **lexical information.** The target word, its base form, as well as the same collocations used in WSD: $c_{-1,-1}$, $c_{+1,+1}$, $c_{-2,-2}$, $c_{+2,+2}$, $c_{-2,-1}$, $c_{+1,+2}$, $c_{-3,-1}$, $c_{+1,+3}$, $c_{-1,+1}$, $c_{-2,+1}$, $col_{-1,+2}$
- **parts-of-speech.** The part-of-speech for the following words relative to the target word: $p_{-3}, p_{-2}, p_{-1}, p_0, p_{+1}, p_{+2}, p_{+3}$ (0 is the target word).
- **gazetteers.** Mapping of target tokens (or n-grams initiated at the target) to 31 categories based on lists downloaded from Ratinov & Roth (2009).
- **cluster membership.** Mapping of words to Brown (1992) clusters (also downloaded from Ratinov & Roth) based on these positions relative to the target word: $bc_{-2}, bc_{-1}, bc_0, bc_{+1}, bc_{+2}$.
- **selectors.** Selectors were included for the target word as in WSD.

Table 8 shows the results for named entity classification. Here, we used one classifier and many potential labels, and thus the most frequent class accuracy is very low, corresponding to the prediction of *organization* for each instance. The inclusion of selectors as features increased the accuracy of our named entity classification system, with a significant 9.3% error reduction. These results, combined with the extensive WSD tests lead us to believe that selectors can also be used successfully as features for many tasks requiring contextual information, such as *prepositional phrase attachment* or *semantic role labeling*, could also benefit from the inclusion of selectors as features.

[5]The Pearson correlation between frequencies of each entity type in our sample versus all instances are 0.982 and 0.945 for the *training* and *test* sets respectively.

bill-n.1	*bill-n.2*	*bill-n.3*	*occur-v.1*	*occur-v.2*	*occur-v.3*
bill	bill	**market**	be	go	go
it	**staff**	system	**happen**	get	**look**
legislation	system	**paper**	occur	come	**break**
system	**money**	**note**	go	have	**remove**
program	**time**	bill	**take**	**try**	**find**
law	it	**bond**	work	**lead**	get
plan	**tax**	**stock**	come	**listen**	**place**
you	**work**	**debt**	**see**	work	**keep**
measure	**rent**	**rate**	have	be	**stick**
project	**tuition**	**report**	**change**	**belong**	**stop**

Table 9: The ten most common selectors for each sense of the noun 'bill' and the verb 'occur'. Top selectors which are unique to each sense are emboldened.

5 Discussion: On the Robustness of Selectors

In the previous section we saw that adding selectors to a standard feature space increases classifier accuracy. In this section, we discuss this improvement by examining the values of features extracted for instances in the SemEval-2007 experimental corpus. Particularly, we endeavor to show that selectors contribute robustness to the *WSD* feature space through an abstraction of context that distinguishes senses of words.

The idea of abstracting context is based on the notion that contexts which realize words of similar meaning have similar selectors. Consider the selectors for senses of both words in Table 9: 'bill' and 'occur'. We see that each of the sets of selectors varies depending on the sense of each word. Furthermore, though coarse-grained, one may even infer the sense of each word by considering its most common selectors; they should be similar to the sense.

For the the supervised classifier, selectors are an encoding or abstraction of context to help identify each sense with no need for concept similarity judgments. For example, both sentences below were annotated incorrectly without selectors, but correctly with selectors.

1. *Polls show wide, generalized support for some vague concept of service, but the **bill** now under discussion lacks any passionate public backing.*
2. *Emerson, in his lecture, refers to the "startling experience which almost every person confesses in daylight, that particular passages of conversation and action have **occurred** to him in the same order before, whether dreaming or waking, a suspicion that they have been with precisely these persons in precisely this room, and heard precisely this dialogue, at some former hour, they know not when".*

For sentence 1, selectors of *bill-n.1* seem to best match the instance's local context (i.e. one can imagine inserting selectors from *bill-n.1* in place of 'bill' more easily than selectors from other senses of bill). Though the training set never contained the exact context "...but the ___ now under..", it did produce selectors which match this context. In sentence 2 the immediate context before and after the target word seem contradictory: "... action have **occurred**..." implies *occur-v.1* (to "happen or take place"), while "... **occurred** to him ..." implies *occur-v.1* (to "come to mind"). However, when considering the whole context *occur-v.2* fits best, and in fact, the selectors for this instance match with many of the most frequent selectors for *occur-v.2* such as 'belong', 'lead', 'listen', and 'try'.

5.1 Extensions

We presented evidence that selectors may benefit other lexico-semantic classification tasks in section 4.3.5. Here, we discuss a few extensions to our selector acquisition approach that we believe could bring about further improvements in accuracy. The primary reason we chose to use n-grams as a source for selectors is because the Web search engines that support wildcard search (Yahoo and Google), which is necessary for efficient selector acquisition, no longer support APIs which return all matches. However, because our n-grams were restricted to the order of 5 tokens, the size of local context is limited.

To allow one to search with larger local context, a couple more advanced approaches might be employed. One solution is to use non-wildcard Web search queries (The still-supported Microsoft API could handle this) where candidate selectors are inserted in place of the wildcard. Because this would result in an expensive number of Web queries, one could limit candidates to selectors found through the web-scale 5-grams corpora. Another idea might be to use a smaller corpus than the Web where it is practical to base selectors on grammatical or dependency relationships. A similar approach was done by Lin (1997) without supervision. One can now produce dependency parses over much larger corpora. This would allow one to focus on the important constituents in context as well as capture long distance relationships. Still, part of the attractiveness of web-scale n-grams for selectors is the simplicity. Should our n-gram selectors not contain sufficient local context, one would expect selectors to be ineffective as a type of feature. We found that is not the case.

6 Conclusion

We introduced a novel method for increasing the informative value of a supervised disambiguation set of features by leveraging large unannotated corpora to encode an abstraction of local context via *selectors*. When tested over SemEval-2007 Task 17 and Senseval-3 English Lexical Sample, we found that word sense disambiguation classifiers utilizing selectors performed significantly better than those without. The improvements from selectors come free of any annotation cost, requiring only a web-scale n-gram collection. We believe other tasks, such as *prepositional phrase attachment* or *semantic role labeling*, could also benefit from the inclusion of selectors as features.

Acknowledgments

Support for this research was provided by the Robert Wood Johnson Foundation's Pioneer Portfolio, through a grant to Martin Seligman, "Exploring Concepts of Positive Health."
This work supported in part by the Intelligence Advanced Research Projects Activity (IARPA) via Department of Defense US Army Research Laboratory contract number IARPA W911NF-12-C-0023. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright annotation thereon. Disclaimer: The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of IARPA, DoD/ARL, or the U.S. Government.
Fernando Gomez's research was supported in part by the NASA Engineering and Safety Center under Grant/Cooperative Agreement NNX08AJ98A.

References

Agirre, E., López de Lacalle, O., Fellbaum, C., Hsieh, S., Tesconi, M., Monachini, M., Vossen, P., and Segers, R. (2010). Semeval-2010 task 17: All-words word sense disambiguation on a specific domain. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, pages 75–80, Uppsala, Sweden.

Bergsma, S., Lin, D., and Goebel, R. (2009). Web-scale n-gram models for lexical disambiguation. In *International Joint Conference on Artificial Intelligence*, pages 1507–1512.

Bergsma, S., Pitler, E., and Lin, D. (2010). Creating robust supervised classifiers via web-scale n-gram data. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 865–874, Uppsala, Sweden.

Brown, P. F., deSouza, P. V., Mercer, R. L., Pietra, V. J. D., and Lai, J. C. (1992). Class-based n-gram models of natural language. *Computational Linguistics*, 18:467–479.

Cai, J. F., Lee, W. S., and Teh, Y. W. (2007). NUS-ML: Improving word sense disambiguation using topic features. In *Proceedings of the International Workshop on Semantic Evaluations*, volume 4.

Cárcamo, J., Gelbukh, A., and Calvo, H. (2008). An innovative two-stage wsd unsupervised method. *Procesamiento del lenguaje natural*, 40:99–105.

Chan, Y. S., Ng, H. T., and Zhong, Z. (2007). NUS-PT: Exploiting parallel texts for word sense disambiguation in the english all-words tasks. In *Proceedings of Proceedings of SemEval-2007*, pages 253–256, Prague, Czech Republic.

Clear, J. H. (1993). The british national corpus. *The digital word: text-based computing in the humanities*, pages 163–187.

Dligach, D. and Palmer, M. (2008). Novel semantic features for verb sense disambiguation. In *Proceedings of ACL-08: HLT, Short Papers*, pages 29–32, Columbus, Ohio. Association for Computational Linguistics.

Downey, D., Broadhead, M., and Etzioni, O. (2007). Locating complex named entities in web text. In *Proceedings of the 20th international joint conference on Artifical intelligence*, IJCAI'07, pages 2733–2739.

Finkel, J. R. and Manning, C. D. (2009). Joint parsing and named entity recognition. In *Proceedings of the North American Association of Computational Linguistics (NAACL 2009)*.

Huang, J., Gao, J., Miao, J., Li, X., Wang, K., and Behr, F. (2010). Exploring web scale language models for search query processing. In *19th International World Wide Web Conference (WWW-2010)*, Raleigh, NC.

Lee, Y. K. and Ng, H. T. (2002). An empirical evaluation of knowledge sources and learning algorithms for word sense disambiguation. In *EMNLP '02: Proceedings of the ACL-02 conference on Empirical methods in natural language processing*, pages 41–48, Morristown, NJ, USA.

Lin, D. (1997). Using syntactic dependency as local context to resolve word sense ambiguity. In *Proceedings of the 35th annual meeting on Association for Computational Linguistics*, pages 64–71.

Lin, D. (1998). Automatic retrieval and clustering of similar words. In *Proceedings of COLING-ACL 98*, pages 768–774, Montreal, Canada. Morgan Kaufmann.

Lin, D., Church, K., Ji, H., Sekine, S., Yarowsky, D., Bergsma, S., Patil, K., Pitler, E., et al. (2010). New tools for web-scale n-grams. In *Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC'10)*, Valletta, Malta.

McCarthy, D., Koeling, R., Weeds, J., and Carroll, J. (2004). Finding predominant word senses in untagged text. In *Proceedings of the 42nd Meeting of the Association for Computational Linguistics (ACL'04)*, pages 280–287, Barcelona, Spain. Association for Computational Linguistics.

Mihalcea, R. (2004). Co-training and self-training for word sense disambiguation. In *HLT-NAACL 2004 Workshop: Eighth Conference on Computational Natural Language Learning*, pages 33–40.

Mihalcea, R. (2007). Using wikipedia for automatic word sense disambiguation. In *North American Chapter of the Association for Computational Linguistics (NAACL 2007)*.

Mihalcea, R., Chklovski, T., and Kilgarriff, A. (2004). The senseval-3 english lexical sample task. In *Senseval-3: Third International Workshop on the Evaluation of Systems for the Semantic Analysis of Text*, pages 25–28, Barcelona, Spain. Association for Computational Linguistics.

Mihalcea, R. and Moldovan, D. I. (1999). An automatic method for generating sense tagged corpora. In *Proceedings of the Association for the Advancement of Artificial Intelligence (AAAI-99)*, pages 461–466.

Miller, G., Beckwith, R., Fellbaum, C., Gross, D., and Miller, K. (1993). Five papers on wordnet. Technical report, Princeton University.

Navigli, R., Litkowski, K. C., and Hargraves, O. (2007). Semeval-2007 task 07: Coarse-grained english all-words task. In *Proceedings of SemEval-2007*, pages 30–35, Prague, Czech Republic.

Pantel, P. and Lin, D. (2002). Discovering word senses from text. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining (KDD '02)*, pages 613–619, New York, NY, USA. ACM Press.

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

Pereira, F., Tishby, N., and Lee, L. (1993). Distributional clustering of english words. In *Proceedings of the 31st annual meeting on Association for Computational Linguistics*, pages 183–190, Stroudsburg, PA, USA.

Pham, T. P., Ng, H. T., and Lee, W. S. (2005). Word sense disambiguation with semi-supervised learning. In *Proceedings of the 20th national conference on Artificial intelligence - Volume 3*, AAAI'05, pages 1093–1098.

Pradhan, S. S., Loper, E., Dligach, D., and Palmer, M. (2007). Semeval-2007 task 17: English lexical sample, srl and all words. In *Proceedings of SemEval-2007*, pages 87–92.

Ratinov, L. and Roth, D. (2009). Design challenges and misconceptions in named entity recognition. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning*, pages 147–155.

Resnik, P. (1997). Selectional preference and sense disambiguation. In *Proceedings of the ANLP Workshop: Tagging Text with Lexical Semantics*, Washington, DC, USA.

Schütze, H. (1998). Automatic word sense discrimination. *Computational Linguistics*, 24(1):97–123.

Schwartz, H. A. and Gomez, F. (2008). Acquiring knowledge from the web to be used as selectors for noun sense disambiguation. In *Proceedings of the Twelfth Conference on Computational Natural Language Learning*, Manchester, England.

Schwartz, H. A. and Gomez, F. (2009). Using web selectors for the disambiguation of all words. In *Proceedings of the NAACL-2009 Workshop on Semantic Evaluations (SEW-2009)*, Boulder, Colorado.

Tjong Kim Sang, E. F. and De Meulder, F. (2003). Introduction to the conll-2003 shared task: Language-independent named entity recognition. In Daelemans, W. and Osborne, M., editors, *Proceedings of CoNLL-2003*, pages 142–147, Edmonton, Canada.

Weischedel, R., Palmer, M., Marcus, M., Hovy, E., Pradhan, S., Ramshaw, L., Xue, N., Taylor, A., Kaufman, J., Franchini, M., et al. (2011). Ontonotes release 4.0. In *LDC2011T03, Philadelphia, Penn.* Linguistic Data Consortium.

Yarowsky, D. (1995). Unsupervised word sense disambiguation rivaling supervised methods. In *Proceedings of the 33rd Annual Meeting of the Association for Computational Linguistics*, pages 189–196.

Yuret, D. (2007). KU: Word sense disambiguation by substitution. In *Proceedings of SemEval-2007*, pages 207–214, Prague, Czech Republic.

Zhong, Z., Ng, H. T., and Chan, Y. S. (2008). Word sense disambiguation using OntoNotes: An empirical study. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 1002–1010, Honolulu, Hawaii.

Zhu, X. (2008). Semi-supervised learning literature survey. technical report.

The French Social Media Bank: a Treebank of Noisy User Generated Content

Djamé Seddah[1,2] Benoit Sagot[1] Marie Candito[1]
Virginie Mouilleron[1] Vanessa Combet[1]
(1) Alpage, Inria Paris-Rocquencourt & Université Paris 7
175 rue du Chevaleret, 75013 Paris, France
(2) Université Paris Sorbonne
7 rue Victor Cousin, 75006, Paris, France
firstname.lastname@inria.fr

ABSTRACT
In recent years, statistical parsers have reached high performance levels on well-edited texts. Domain adaptation techniques have improved parsing results on text genres differing from the journalistic data most parsers are trained on. However, such corpora usually comply with standard linguistic, spelling and typographic conventions. In the meantime, the emergence of Web 2.0 communication media has caused the apparition of new types of online textual data. Although valuable, e.g., in terms of data mining and sentiment analysis, such user-generated content rarely complies with standard conventions: they are *noisy*. This prevents most NLP tools, especially treebank based parsers, from performing well on such data. For this reason, we have developed the French Social Media Bank, the first user-generated content treebank for French, a morphologically rich language (MRL). The first release of this resource contains 1,700 sentences from various Web 2.0 sources, including data specifically chosen for their high noisiness. We describe here how we created this treebank and expose the methodology we used for fully annotating it. We also provide baseline POS tagging and statistical constituency parsing results, which are lower by far than usual results on edited texts. This highlights the high difficulty of automatically processing such noisy data in a MRL.

KEYWORDS: Treebanking, User Generated Content, Parsing, Social Media.

Proceedings of COLING 2012: Technical Papers, pages 2441–2458,
COLING 2012, Mumbai, December 2012.

1 Introduction

Complaining about the lack of robustness of statistical parsers whenever they are applied on out-of-domain text has almost became an overused *cliché* over the last few years. It remains true that such parsers only perform well on texts that are comparable to their training corpus, especially in terms of genre. As noted by Foster (2010) and Foster et al. (2011b), most studies on out-of-domain statistical parsing have been focusing mainly on slightly different newspaper texts (Gildea, 2001; McClosky et al., 2006a,b), biomedical data (Lease and Charniak, 2005; McClosky and Charniak, 2008) or balanced corpora mixing different genres (Foster et al., 2007). The common point between these corpora is that they are edited texts. This means that their underlying syntax, spelling, tokenization and typography remain standard, even if they slightly depart from the newspaper genre. Therefore, standard NLP tools can be used on such corpora. Now, new forms of electronic communication have emerged in the last few years,namely social media and Web 2.0 communication media, either synchronous (micro-blogging) or asynchronous (forums), and the need for comprehensive ways of coping with the new languages types carried by those media is becoming of crucial importance.

In fact, the main consequence of the *Web 2.0 revolution* is that what was formerly restricted to one's inner circle of relations has now became widely available and is furthermore seen as containing potentially the same informativeness as written broadcast productions, that have undergone a full editorial chain, and that serve, most of the time, as the basis of our treebanks. Anyway, if those unlimited stream of texts were all written with the same level of proficiency as our canonical data source, the problem would be *simply*[1] a matter of domain adaptation. Yet, this is far from being the case as shown by Foster (2010). Indeed, in her seminal work on parsing web data, different issues preventing reasonably good parsing performance were highlighted; most of them were tied to lexical differences (coming from either genuine unknown words, typographical divergences, bad segmentation, etc.) or syntactic structures absent from training data (imperative usage, direct discourse, slang, etc.). This suboptimal parsing behavior on web data was in turn confirmed in follow-up works on Twitter and IRC chat (Foster et al., 2011a; Gimpel et al., 2010; Elsner and Charniak, 2011). They were again confirmed during the SANCL shared task, organized by Google, aimed at assessing the performances of parsers on various genres of Web texts (Petrov and McDonald, 2012).

Needless to say, such observations are likely to be even more true on web data written in morphologically rich languages (MRLs). These languages are already known to be arguably harder to parse than English for a variety of reasons (e.g., small treebank size, rich inflexion, free word order, etc.) exposed in details in (Tsarfaty et al., 2010). However, a lot of progress has been made in parsing MRLs using, for examples, techniques built on richer syntactic models, lexical data sparseness reduction or rich feature set. See (Tsarfaty and Sima'an, 2008; Versley and Rehbein, 2009; Candito and Crabbé, 2009; Green and Manning, 2010) to name but a few. The questions are thus to know: (1) to what extend MRL user generated content is *parsable*? and (2) more importantly, what is needed to fill that performance gap?

To answer question 1, we introduce the first release of the French Social Media Treebank, a representative gold standard treebank for French user-generated data. This treebank consists in around 1,700 sentences extracted from various types of French Web 2.0 user generated content (Facebook, Twitter, video games and medical board). This treebank was developed independently from the Google Web Treebank (Bies et al., 2012), the treebank used as development and test data for the above-mentioned SANCL shared task. In order to get first insights

[1] Cf. (McClosky et al., 2010) for numerous evidences of the non-triviality of that task.

into question 2, we provide a first set of POS tagging and parsing results using state-of-the-art systems trained on the French Treebank (FTB, Abeillé et al. (2003)), using our treebank as an evaluation corpus. These results show how difficult it is to process French user-generated data: for example, parsing results range from an astoundingly low 39.11% of labeled brackets F-score for the noisiest type of texts to 71-72% for better edited web parts — to be compared with the 86-89% regularly obtained on the FTB.

In the remaining of this paper, we first describe how we built the French Social Media Bank and the underlying motivations; we then introduce our annotation scheme which is based on the French Treebank (Abeillé et al., 2003) guidelines but extends it in many ways, due to the specificities of user-generated content. Next, we describe our annotation methodology, including the pre-processing tools that we developed and used, which were specifically adapted to deal with noisy texts. Finally, we discuss the results of baseline evaluation experiments on POS tagging, including results when using a dedicated wrapper for dealing with noisy texts, and constituency parsing. Since our tools were only trained on the FTB, which means that our results are baselines for future work based on our new French Social Media Bank.

2 Motivation and Corpus

As its English counterpart, the French web 2.0 generates a virtually unlimited stream of textual data. This term covers a wide range of practices, among which we decided to focus on microblogging (FACEBOOK and TWITTER) and on two types of web forums: one dedicated to general health issues for a wide public audience, DOCTISSIMO and one centered around video games issues (platform, help centers), JEUXVIDEOS.COM.[2] As we said in the introduction, we want to use these corpora to evaluate how difficult it is to parse raw user generated content in French and establish a realistic baseline using techniques we have successfully applied on well-written French texts.

To this end, we selected our corpora by direct examination through various search queries and ranked the texts according to our perception of how far they were from the French Treebank style (see below for details). We further added some very noisy texts to serve as a stress test for French statistical parsing. Table 1 presents some properties of our main corpora.

	# sent.	# tokens	avg. Length	std dev.
DOCTISSIMO	771	10834	14.05	10.28
high noisiness subcorpora	36	640	17.78	17.63
other subcorpora	735	10194	13.87	9.74
JEUXVIDEOS.COM	199	3058	15.37	14.44
TWITTER	216	2465	11.41	7.81
high noisiness subcorpora	93	1126	12.11	8.51
other subcorpora	123	1339	10.89	7.20
FACEBOOK	452	4200	9.29	8.17
high noisiness subcorpora	120	1012	8.43	7.12
other subcorpora	332	3188	9.60	8.49

Table 1: Corpus properties

Measuring noisiness In order to quantitatively corroborate our intuitions concerning the level of noise in our corpora, and for measuring their various levels of divergence compared to

[2] `http://facebook.fr`, `https://twitter.com` (automatically configured to provide French tweets first based on IP address geo-localization), `http://forum.doctissimo.fr/` and `http://www.jeuxvideo.com/`

the French Treebank (Abeillé et al., 2003), we defined an ad-hoc *noisiness* metrics. It is simply defined as a variant of the Kullback–Leibler divergence[3] between the distribution of trigrams of characters in a given corpus and the distribution of trigrams of characters in a reference corpus.[4] As can be seen from Table 2, to which we added scores for the FTB dev and test sets for comparison purposes, our intuitions are confirmed by this metric. It shows that we cover various levels of noisiness, and that our reference corpus actually diverges more from the subcorpora that we have tagged as particularly *noisy*. As we shall see below, we have used this information for deciding for each subcorpus whether to pre-annotate it in a standard way or using a dedicated noise-tolerant architecture described in section in Section 5.1.

	noisiness score
DOCTISSIMO	0.37
high noisiness subcorpora	1.29
other subcorpora	0.31
JEUXVIDEOS.COM	0.81
FTB DEV	0.03
FTB TEST	0.003

	noisiness score
TWITTER	1.24
high noisiness subcorpora	1.46
other subcorpora	1.08
FACEBOOK	1.67
high noisiness subcorpora	2.44
other subcorpora	1.30

Table 2: Noisiness scores computed on tokenized version of the various sub-corpora. The (tokenized) FTB training set is used as a reference.

2.1 Corpus Overview

In this section we briefly introduce our corpora. All but the JEUXVIDEOS.COM corpus were collected in two phases: A first one dedicated to a light study of their contents (lexical differences, required level of preprocessing, etc.). At first glance, they seemed almost too edited and almost *too easy* for our parser. So in a second phase, we decided to look explicitly for texts harder to read and understand for average French speakers. We used French slang words in our search queries, including *verlan*[5] words, as well as urban youth idiomatic constructions such as *grave*[6] or *sa race*[7]. This lead to subcorpora that we found noisier, as was confirmed by the above-described metrics (see Table 2). In the case of the DOCTISSIMO part, we gathered texts from a forum dedicated to sexual intercourse problems between young adults. This choice was not without causing ethical concerns but given the fact that all private mentions were of course anonymized and all explicit references were filtered out, we ended up with 50 extremely noisy sentences, but greatly diverging from the newswire genre, and thus extremely

[3]It differs from a standard Kullback–Leibler distance because we apply a preliminary pre-processing to the corpora involved: (i) URLs, e-mail addresses, Twitter hashtags and mentions are removed, (ii) all characters that do not belong to an extensive list of characters that might be used in French sentences are replaced by a unique "non-standard character," (iii) non-content sentences are ignored (e.g., tweet headers such as *Firstname Lastname @firstnamelastname*).

[4]Preliminary experiments on character bigrams and 4-grams have shown that the former are not informative enough and the latter lead to similar results than with trigrams. We also tried comparing distributions of tokens. Correlation with both intuition and parsing accuracy prove similar to that obtained with character 3-grams. However, token-based distribution divergences are less adequate for several reasons, among which: (i) correctly spelled unknown words affect more heavily token distributions than character-based distributions, whereas they should not affect the noisiness measure (e.g., they have a limited impact on tagging and parsing accuracy); (ii) character trigram distributions are less sparse than token distributions; (iii) there are more character trigrams than tokens in a sentence. Put together, the two last reasons show that a sound noisiness score on small corpora, or even at the sentence level, is more likely to be sound when working on character trigrams than on tokens.

[5]Very common French slang words where syllables are inverted to form new words which can in turn be *verlanized*. For instance, *arabe* 'arabic' is turned into *beur* which has been inverted again into *rebeu*.

[6]Post or pre-verbal intensifier adverb. *J'ai adoré grave!*, similar in meaning and style to *I totally enjoyed it!*.

[7]Post-verbal intensifier adverbial phrase, with the same usage as "one's a.. off."

interesting to evaluate our parsing chain.
Let us now describe and give examples for the various extracted subcorpora.

JEUXVIDEOS.COM [8] We collected data from 4 threads: *Call of Duty: Modern Warfare 3©*, *PC*, *Xbox 360©*, *Wii©* and *Linux*. Apart from spelling errors often involving phonetic spelling, which is found in all our corpora, this corpus is interesting because of the frequent use of English words, if not phrases, and a highly specialized lexicon.
In the examples below, the first line reproduces the original text, the second line is a standardized French version and the third line, an English translation attempt.

(1) a. Ces pas possible déjà que battelfield a un passe online
Ce n'est pas possible, Battlefield a déjà un pass en ligne
It's not possible, Battlefield already has an online pass

b. Si y'a que Juliet &Zayn qui sont co' sur le RPG,et qui font leur vie tranquilles
Si, il n'y a que Juliet et Zayn qui sont connectés sur le jeux de rôle et qui font leur vie tranquilles
Only Juliet and Zayn are connected on the RPG and are quiet doing their own business

DOCTISSIMO This corpus is made of two parts, each focusing on a different subtopic concerning birth control: patch birth control and pregnancy test related questions. These topics are populated by women of different ages and with different writing styles. The latter one being filled by younger women, the writing style is somewhat more sloppy. Moreover, as mentioned above, we added 50 extremely noisy sentences from the sexual intercourse section.

(2) a. pt que les choses ont changé depuis ?
Peut-être que les choses ont changé depuis ?
Maybe things have changed since then?

b. lol vu que 2-3 smaine apres qd jai su que j'etai enceinte jetai de 3 semaine.....
lol, vu que 2-3 semaines après, quand j'ai su que j'étais enceinte, je l'étais de 3 semaines....
Lol, given that 2-3 weeks later, when I learned I was pregnant, I was for 3 weeks...

c. car je ne me senté pa desiré, pa aimé, pa bel du cou, g t pa grd chose en fet.
Car je ne me sentais pas désirée, pas aimée, pas belle du coup, je n'étais pas grand chose en fait.
Because I didn't feel desired, nor loved, thus not beautiful, I wasn't much actually.

TWITTER This corpus is made of two parts: the first one focuses on news events of late November 2011.[9] Because all these tweets seemed to originate from semi-professional writers (mostly bloggers, journalists, politically engaged people), we built a second part with genuine non edited text. We used a list of keywords to gather such tweets and selected a balanced subset of those, as far as the style of writing is concerned.

(3) a. Je soupçonnes que "l'enfarineuse" était en faite une cocaineuse vu la pêche de #Hollande ce soir à #Rouen.
Je soupçonne que l'enfarineuse était en faite une cocaïneuse vu la pêche de #Hollande ce soir à #Rouen.
I suspect that the "flouring-lady" was actually a cocaïn-lady given the energy of #Hollande this night at #Rouen.

[8] Collected on November 9, 2011.

[9] An incident involving the left-wing candidate to the French presidency election, a so-called French hidden son of Adolf Hitler, and the then new right-wing election motto.

b. @IziiBabe C mm pa élégant wsh tpx mm pa marshé a coté dsa d meufs ki fnt les thugs c mm pa leur rôle wsh
Ce n'est même pas élégant voyons, tu ne peux même pas marcher à coté de sa petite amie qu'ils font les voyous, ce n'est même pas leur rôle voyons.
It is not even elegant. One cannot even walk besides his girl friend, they already start bullying people. It is not even their role.[10]

FACEBOOK This corpus was built using publicly available comment threads on public profiles, with a focus on relatively known reality TV pseudo-artists known for their personal usage of French mixed with English. More texts were added using queries based on common first names (Sophie, Romain) and some French public personalities . One of the difficulties of FACEBOOK lies in the varying usage of the displayed login name in comments: it can either be part of the sentence (e.g., "[Spiderman] is tired") or not (e.g., "[Spiderman] I'm tired"). We decided to systematically keep these logins. We leave it to a post-processing step to remove them if appropriate. Note that the noisiest part of this corpus was not taken into account while adapting our POS tagger as described below.

(4) a. L' Ange Michael vraiment super conten pour toi mé tora plus grace a moi tkt love you !
L'Ange Michael: (Je suis) Vraiment super content pour mais tu auras plus grace à moi. Ne t'inquiètes pas. Je t'aime !
The Angel Michael: (I am) Really very happy for him but you'll get more because of me. Don't worry. I love you!

b. Afida: Viens on se check dans la vibes du moove pour voir comment on peut faire la hype à Hollywood avec Jane et Bryan
Afida: n/a
Afida: Come on, we'll check in into the moove's vibe to see how we can be hip in Hollywood with Jane and Bryan

3 Linguistics of user generated content

It is important to note that the aim of our work is to provide a sample of user-generated texts that are particularly difficult to parse for any parser based on an edited text treebank. It does not correspond to an single homogenous domain, although some specificities of user-generated content are found across various types of web data. Moreover, in some cases, and most notably TWITTER, such data include both linguistic content and media-specific meta-language. This meta-language (such as TWITTER's "RT" ("Retweet"), at-mentions and hashtags) is to be extracted before parsing *per se* or other types of linguistic processing. In this work, we focused on the linguistic content. Therefore, we deal with meta-language tokens only when they are embedded within or adjacent to purely linguistic content (e.g., the tweet itself, provided it consists of one or several sentences).

Prevalent idiosyncrasies in user generated content can be characterized on two axes: one which can be roughly describe as "the encoding simplification axis" which covers ergographic (1) and transverse phenomena (2) and the other "sentiment expression axis" which cover phenomena, qualified below as marks of expressiveness (3), emulating the same goal as sentiment expressed through prosody and gesture in direct interaction.

1. **Ergographic phenomena**, that is phenomenon aiming at reducing the writing effort, perceived as first glance as genuine misspell errors, cover in fact such a various set

[10]The translation is most certainly not accurate as even the original text is barely understandable for us.

of strategies it can be seen as a simplification of the encoding. Besides obvious typos, such as letter inversion[11], errors in letter doubling[12], wrong present participle[13] and so on, misspellings can be hard to categorize as such if they result in simpler word forms. This is why we include this category in the following list of phenomenon even if its intentionality is not always attested.

Phenomenon	Attested example	Standard counterpart	Gloss
a. Diacritic removal	*demain c'est l'***ete**	*demain c'est l'été*	'tomorrow this is summer'
b. Phonetization	*je suis* **oqp**	*je suis occupé*	'I'm busy'
c. Simplification	*je* **sé**	*je sais*	'I know'
d. Spelling errors	*tous mes* **examen** **son** *normaux*	*tous mes examens sont normaux*	'All my examinations are normal'

To this list we can also note the somewhat frequent omission of copula verbs and more generally different forms of elision (the subject pronoun, the negative adverb "ne"). Although not strictly lexical, this omission results also in less writing efforts. This is also noted in the Google Web Treebank where they compare this tendency in user generated content English to pro-drop languages' clitic elision.

2. **Transverse phenomenon: Contractions and typographic diaeresis.** Some phenomena can affect the number of tokens, compared to standard French, either by replacing several standard language tokens by only one, which we shall call a *contraction*, or conversely by splitting one standard language token into several tokens, called *typographic diaeresis*. Such phenomena are frequent in our corpora, and they need a specific annotation scheme (cf. Section 4). Note that the resulting non-standard tokens might be homographs of existing words, bringing more ambiguities if not properly analyzed.

   Contractions are way more diverse than in standard French (as instanced in the FTB). The only contractions that exist in standard French involve the prepositions *à* and *de* when followed by the definite article *le(s)* or the (rare) relative pronoun *le(s)quel(s)*. For instance, *à les* 'to the$_{\text{PLUR}}$' mandatorily becomes *au*.
   Within the FRENCH SOCIAL MEDIA BANK, we found sequences such as: (i) contraction between a subject clitic, a verbal form and a negation particle (e.g., *lapa* for *elle n'a pas*, 'she has not') ; (ii) contraction of interrogative verbal forms (e.g., *atu* for *as-tu*, 'have you') ; (iii) contraction and apocope of word compounds (e.g., *nimp* for *n'importe quoi*, 'rubbish'); (iv) contraction of determiners and nouns (e.g., *lesprit* for *l'esprit*, 'the spirit') and (v) contraction of object relative pronouns (or subordinate conjunction) and subject clitic (e.g., *qil* for *qu'il*, literally 'that he').

   Typographic diaeresis can be illustrated by *c a dire*, where these three tokens stand for the standard one-token conjunction *c'est-à-dire* (standing for "that is" or "namely"). It can also happen on top of a presumably already contracted token (e.g., *c t* for *ct/c'était*,'it was'). Note that many contractions and typographic diaeresis are built around a verb which is prefixed by a clitic. Therefore such contractions can project function labels and most of the time are the head of the sentence. Their mishandling can therefore propagate errors way beyond their immediate morpho-syntactic context and thus impacts parser performance very strongly.

[11] As in *J'ia* instead of *J'ai*/'I have'.
[12] e.g., *Dévelloppement* instead of *développement*/development
[13] e.g., "-ent" instead of "-ant"/ing, which are pronounced the same.

3. **Marks of expressiveness**
   Our treebank focuses in providing a sample of French Social Media web data, therefore most of its content describe dialogs and various forms of interaction between users through social media interface. Lacking ways of expressing sentiments, irony or anger through prosody and gesture, users deploy a wide range of strategy to add another dimension to their text stream. Most of them are evident, like graphical stretching, overuse of strong punctuation marks (as in "BEST. MOVIE. EVER." or "!!!!Greaaat!!!), abuse of emoticons, sometimes used as a verb (e.g., *Je t'<3* for *Je t'aime*,'I love you') or inside usernames, mixing between lowercase and uppercase and so on. Some are more anecdotal and tied to the particular type of software used to support a web forum, which allows the inclusion of url pointing to an animated picture, itself used to replace an emoticon. Needless to say that such urls add a considerable amount of noise in web forum texts. We list the main cases of such phenomenon in the table below.

Phenomenon	*Attested example*	*Standard counterpart*	Gloss
e. Punctuation transgression	*Joli !!!!!!*	*Joli !*	'nice!'
f. Graphemic stretching	*superrrrrrrrr*	*super*	'great'
g. Typographic transgression	*N* *U* *L*	*nul*	'bad'
h.Emoticons/smileys	*:-),<3*	–	–

Obviously the main effect of those different writing artifacts is to considerably increase the level of unknown words (compared to a treebank). More importantly, the new morphology brought by the those phenomenon complicates any processing based on regular unknown word identification through suffix analysis.
However, our general annotation strategy consists in staying as close as possible from the French Treebank guidelines (Abeillé et al., 2003) in order to have a data set as compatible, as much as possible, with existing resources.

4 Annotation scheme

In order to obtain evaluation treebanks compatible with parsers trained on the FTB, we have used as basis the FTB annotation scheme and followed as much as possible the corresponding annotation guidelines for morphology, syntagmatic structure and functional annotation (Abeillé et al., 2003). More precisely, we started from a slight modification of this annotation scheme, referred to as the FTB-UC and added specific guidelines for handling idiosyncrasies user-generated content corpora.

4.1 FTB-UC vs. FTB

We targeted the annotation scheme of the FTB-UC (Candito and Crabbé, 2009), that was obtained by automatic modification of the FTB. The modifications with respect to the orignal FTB concern the tagset, the standardization of preposition and complementizer projections, and multi-word units:

- Multi-word units are very frequent in the FTB: 17% of the tokens belong to a compound. Compounds range from very frozen multi-word expressions like *y compris* 'including' (literally 'there included') to named entities. They include syntactically regular compounds with compositional semantics, such as *loi agraire* 'land law', that are encoded

as compounds because of a non-free lexical selection. These syntactically regular compounds tend to be inconsistently encoded in the FTB.[14] Further, the FTB includes "verbal compounds" that are potentially discontinuous, which provoke variable annotations. In the FTB-UC, these syntactically regular compounds are automatically mapped to a regular syntagmatic representation. We followed this rule for the annotation of the French Social Media Treebank. This has the virtue of uniformity, but clearly requires further treatment to spot clear cases of compounds (with non compositional semantics).

- Tagset: the tagset includes 28 POS tags —originally tuned by Crabbé and Candito (2008) to optimize parsing—, which are a combination of one of the 13 coarse-grained categories and other information that is encoded in the FTB as features, such as verbal mood information, proper versus common noun distinction, wh-feature, etc.
- Complementizers and prepositions: we annotate so that prepositions project a PP independently of the category of their object, contrary to the FTB's guidelines, in which prepositions with nominal objects project a PP, but those with infinitival objects don't. Further, we systematically use a sentential phrase as sister node to complementizers, contrary to the flat structure of the FTB.

Other notable additions to the annotation scheme concern the non-terminal tagset to which we added the FRAG label. It concerns phrases that cannot be syntactically attached to the main clause of a syntactic unit, e.g., mostly salutations, time stamp, meta sentential marks of emotion (emoticons, strong interjections). It also covers the case of usernames, at-mentions, and URL appended to a post/sentence (cf. FACEBOOK sentence 4a).

4.2 Additional extensions

A first extension needed for dealing with our data was to add two new POS tags, namely *HT* for TWITTER hashtags and *META* for meta-textual tokens, such as TWITTER's "RT". Note that TWITTER at-mentions as well as URLs and e-mail addresses have been tagged *NPP*. The rationale for this is to remain consistent with our tagging and parsing models trained on the FTB, which do not contain such tokens. This constitutes the main difference with other works on user-generated data.

However, the main extensions we added to the FTB annotation scheme are related to contraction and typographic diaeresis phenomena described in Section 3. The way we annotate (and automatically preannotate) such sequences is illustrated in Table 4. Let us now provide a few more details on each of these two cases.

Contracted tokens are associated with a combined POS tag which lists the sequence of each underlying words' tag. Let us illustrate this on *qil*, a non-standard contraction for *qu' il*. At least in some contexts, the tokens in the standard version *qu' il* would have been tagged respectively *CS* and *CLS*. In such contexts, the non-standard contracted token *qil* is tagged *CS+CLS*. Table 3 lists all such compound tags occurring more than twice in the treebank (37 more remaining).In some cases, such contractions involve two underlying forms, one being a verb and the other an argument of the verb (e.g., *jai* for *j' ai* 'I have'). In such cases, function labels are associated directly with the contracted token.

On the other hand, in cases of typographic diaeresis, the category of the multi-token standard counterpart is given to the last token, all others receive the special tag *Y*. Taking *c a dire* as an

[14] For instance *pays industrialisés* (*industrialized countries*) appears twice as a compound and 41 times as two words; *taux d'intérêt* (*interest rate*) appears 80 times as a compound and 25 times as two words.

Compound tag	Tag occ.	Attested example	Standard counterpart	Gloss
CLS+V	54	*c*	*c' est*	'it is'
ADV+CLO	12	*ni*	*n' y*	'(neg. adv.) (loc. clitic)'
CS+CLS	12	*qil*	*qu' il*	'that it/he'
CLS+CLO	11	*jen*	*j' en*	'I (gen. clitic)'
CLO+V	9	*ma*	*m' a*	'me$_{\text{dative}}$ has'
DET+NC	9	*lamour*	*l' amour*	'the love'
ADV+V	7	*non*	*n' ont*	'(neg. adv.) have$_{\text{3rd plur}}$'

Table 3: Non-standard compound tags occurring at least 3 times.

example, which stands for the coordination conjunct *c'est-à-dire*, the first two tokens is tagged *Y* and *dire* is tagged *CC*. Note that this is consistent with the way such cases are annotated in the Google Web Treebank.[15]

Note that both phenomena can appear together. This is the case for example with *c t* instead of *c' était* 'it was' (tag sequence: *CLS V*): both letters *c* and *t* are used phonetically — as in using *U* for *you* in English — and sound like the two syllables of *c'é* and *tait*. Therefore, mapping *c* to *c'* and *t* to *était* is not adequate. In this case, we consider that a contraction followed by an typographic diaeresis has occurred, and associate *c* with the tag *Y* and *t* with *CLS+V*.

5 Annotation Methodology

We built manually validated treebank following a now well established methodology: we first defined a sequence of annotation layers, namely (i) sentence splitting, tokenization and POS tagging, (ii) syntagmatic parsing, (iii) functional annotation. Each layer is annotated by an automatic preprocessing that relies on previously annotated layers, followed by validation and correction by human annotators. At each step, annotators were able modify the choices made at previous stages. Our methodology is summarized as follows and detailed section 5.1:

- Segmentation, tokenization and POS tagging followed by manual validation and correction by one expert annotator.
- Constituency parsing followed by manual validation and correction by two annotators followed by an adjudication step.
- Functional annotation followed by manual validation and correction by two annotators followed by and adjudication step.

5.1 Pre-annotation strategies for the tokenization and POS layers

As mentioned above, we used two different strategies for tokenization and POS pre-annotation, depending on the noisiness score.

For less *noisy* corpora (those with a noisiness score below 1), we used a slightly extended version of the tokenization and sentence splitting tools from our standard FTB-based parsing architecture, Bonsai (Candito et al., 2010). This is because we want to have a tokenization that is as close as possible from the principles underlying the FTB's tokenization. Next, we used the POS-tagger MORFETTE (Chrupała et al., 2008) as a pre-annotator.

For corpora with a high noisiness score, we used a specifically developed pre-annotation process. This is because in such corpora, spelling errors are even more frequent, but also because

[15] In the Google Web Treebank, the counterpart of our tag *Y* is the tag *GW*.

the original tokens rarely match sound linguistic units, as can be seen on the example in Table 4 taken from the DOCTISSIMO file with the highest noisiness score. The idea underlying this pre-processing is to wrap the POS tagger (in this case, MElt) in such a way that it actually has to tag a sequence of tokens that is as close as possible to standard French, or, rather, to its training corpus (in this case, the FTB). Hence the following process, illustrated on a real example in Table 4:

1. We first apply several regular-expression-based grammars taken from the SxPipe pre-processing chain (Sagot and Boullier, 2008) for detecting smileys, URLs, e-mail addresses, Twitter hashtags and similar entities, in order to consider them as one token even if they contain whitespaces.
2. Next, we use the same tokenizer as for less *noisy* corpora.
3. We apply a set of 327 rewriting rules that were forged as follows: first, we extracted from our development corpus (all subcorpora but for the *noisy* Facebook subcorpus) *n*-gram sequences involving unknown tokens or occurring at an unexpectedly high frequency; then we manually selected the relevant ones and provided them manually with a corresponding "correction". The number of "corrected tokens" obtained by applying these rules might be different from the number of original tokens. In such cases, we use 1-to-*n* or *n*-to-1 mappings. For example, the rule *ni a pa* → *n'␣y a pas* explicitely states that *ni* is an amalgam for *n'* and *y*, whereas *pas* is the correction of *pa*.
4. We use the MElt tagger (Denis and Sagot, 2009), trained on the FTB-UC and the Le*fff* lexicon (Sagot, 2010), for POS-tagging the sequence of corrected "tokens".
5. We apply a set of 15 generic and almost language-independent manually crafted rewriting rules, originally developed for English data (see below), that aim at assigning the correct POS to tokens that belong to categories not found in MElt's training corpus, i.e., the FTB; for example, all URLs and e-mail addresses are post-tagged as proper nouns whatever the tag provided by MElt; likewise, all smileys get the POS for interjections.
6. We assign POS tags to the original tokens based on the mappings between corrected POS-tagged tokens and original ones, and following the guidelines given in section 4.2. If a unique corrected token is mapped to more than one original tokens, all tokens but the last one are assigned the tag *Y*, and the last one receives the tag of the unique corrected token. If more than one corrected tokens are mapped to one original token, it is assigned a tag obtained by concatenating the tags of all corrected tokens, separated by the '+' sign. If the mapping is one-to-one, the POS tag provided by MElt for the corrected token is assigned to the corresponding original token.

This architecture is now available as part of the MElt distribution. It was also applied on English web data in the context of the SANCL shared task on parsing web data (Petrov and McDonald, 2012), with state-of-the-art results (Seddah et al., 2012).

5.2 Annotation strategy for constituency and functional annotation

Parse pre-annotation was achieved using a state-of-the-art statistical parser trained on the FTB-UC, provided with the manually validated tagging. The parser we used was the Berkeley parser (Petrov and Klein, 2007) adapted to French (Crabbé and Candito, 2008). Note that when the validated pos tags were discarded by the parser, in case of too many unknown word-pos pairs, those were reinserted.
To assess the quality of annotation, we calculated the inter annotator agreement using the

Original tokens	Gold corrected "tokens"	Automatically corrected and POS-tagged "tokens"	POS tags automatically assigned to the original tokens	Manually corrected POS tags for the original tokens
sa	ça	ça/*PRO*	sa/*PRO*	sa/*PRO*
fé	fait	fait/*V*	fé/*V*	fé/*V*
o moin	au_moins	au/*P+D* moins/*ADV*	o/*P+D* moin/*ADV*	o/*P+D* moin/*ADV*
6	6	6/*DET*	6/*DET*	6/*DET*
mois	mois	mois/*NC*	mois/*NC*	mois/*NC*
qe	que	que/*PROREL*	qe/*PROREL*	qe/*CS*
les	les	les/*DET*	les/*DET*	les/*DET*
preliminaires	préliminaires	preliminaires/*NC*	preliminaires/*NC*	preliminaires/*NC*
sont	sont	sont/*V*	sont/*V*	sont/*V*
sauté	sautés	sauté/*VPP*	sauté/*VPP*	sauté/*VPP*
c a dire	c'est-à-dire	c'est-à-dire/*CC*	c/*Y* a/*Y* dire/*CC*	c/*Y* a/*Y* dire/*CC*
qil	qu' il	qu'/*CS* il/*CLS*	qil/*CS+CLS*	qil/*CS+CLS*
yen	y en	y/*CLO* en/*CLO*	yen/*CLO+CLO*	yen/*CLO+CLO*
a	a	a/*V*	a/*V*	a/*V*
presk	presque	presque/*ADV*	presk/*ADV*	presk/*ADV*
pa	pas	pas/*ADV*	pa/*ADV*	pa/*ADV*

Table 4: Gold and automatic correction and POS tags for the following sentence extracted from the DOCTISSIMO file with the highest noisiness score 'Forplay have disappeared for at least 6 months, that is there is almost none.'

Parseval F-measure metric between two functionally annotated set of parses. Agreements range between 93.4 for FACEBOOK data and 97.44 for JEUXVIDEOS.COM (Table 5) and are on the same range than the DCU's Twitter corpus agreement score (Foster et al., 2011a). Similarly to that corpus, the disagreements involve fragments, interjections and the syntactic status to assign to meta-tokens elements. We note that our agreement scores are higher than those reported in other out-of-domain initiatives for French (Candito and Seddah, 2012). This small annotation error rate comes from the fact that the same team annotated both treebanks and was thus highly trained for that task. Maybe more importantly, social media sentences tend to be shorter than their edited counterparts so once POS tagging errors are solved, the annotation task is made relatively easier.

DOCTISSIMO	95.05	JEUXVIDEOS.COM	97.44
TWITTER	95.40	FACEBOOK	93.40
DCU'S TWITTERBANK	95.8	-	-

Table 5: Inter Annotator agreement

6 Preliminary experiments

Experimental Protocol In the following experiments, we used the FTB-UC as training data set, in its classical settings (test set: first 10%, dev set: next 10% and train set: the remaining.), see (Candito et al., 2009; Seddah et al., 2009) for details.

POS tagging experiments We have conducted preliminary evaluation experiments on the MElt POS-tagger(Denis and Sagot, 2009), used as such or within the normalization and correction wrapper described in the previous section. In Table 6, we provide POS-tagging accuracy results over the various subcorpora, following the DEV/TEST split described above. The results indicate that using the normalization and correction wrapper leads to significant improve-

ments in POS tagging accuracy. One can note that our accuracy results on standard TWITTER subcorpora are similar to the figures reported by Foster et al. (2011a) on English TWITTER data, although these figures are obviously not directly comparable, as they concern different languages using different tagsets. Another interesting observation is that these accuracy results are correlated with the noisiness metrics defined above.[16]

	DEV		TEST	
	MElt−corr	MElt+corr	MElt−corr	MElt+corr
DOCTISSIMO				
high noisiness subc.	56.41	80.78	–	–
other subcorpora	86.57	88.42	87.78	89.18
JEUXVIDEOS.COM	81.20	82.41	82.64	83.63
TWITTER				
high noisiness subc.	80.21	84.51	74.50	81.65
other subcorpora	84.09	89.00	86.23	88.24
FACEBOOK				
high noisiness subc.	–	–	67.00	70.75
other subcorpora	71.75	76.87	78.66	82.00
all	*80.64*	*84.72*	*83.10*	*85.28*
FTB (edited Text)	97.42	97.42	97.79	97.78

Table 6: Accuracy results for the MElt POS-tagger, embedded or not within the normalization and correction wrapper ("MElt+corr" and "MElt−corr" respectively). See text for details.

Baseline statistical parsing experiments In addition to the POS tagging experiments which showed that performance could greatly be improved using our normalization and correction wrapper, we performed a set of baseline experiments on the raw (tokenized) corpora using the PCFG-LA parser of Petrov et al. (2006) adapted to handle French morphology by (Crabbé and Candito, 2008). We used the PARSEVAL metrics applied to all sentences. Note that full scale experiments aimed at getting optimum parsing performance on this data set are out of the scope of this paper. We instead insist on providing baseline results, setting out a lower bound which assesses the difficulty of French User generated content parsing.
As expected, the results[17] provided in Table 7 show that there exists a large room for improvements. Interestingly, our user generated content data set seems even more difficult to parse than French biomedical data (67.79% vs 81.25% of F_1 score, on the Emea French test set for sentences of length lesser than 41), known to contain a high amount of unknown words and unusual phrase structures (Candito et al., 2011). Surprisingly, our parser performs poorly on FACEBOOK data, more than it does on TWITTER. In order to test their similarity, we can compare the respective noisiness scores of their subcorpora (FACEBOOK with respect to TWITTER = 1.42, TWITTER with respect to FACEBOOK 0.85). This shows that TWITTER data are more homogeneous than their FACEBOOK counterparts. Part of the reason lies in the inner nature of those social media: TWITTER is a live micro blogging platform, meaning that the content for a given trending topic shows fewer lexical divergences in a very short amount of time, whereas FACEBOOK public post are more distributed over time and posters.

The next step will involve collecting large unlabeled corpora to perform experiments with self-training techniques (McClosky and Charniak, 2008; Foster et al., 2011b) and unsupervised

[16]Simple linear regressions lead to the following results: without the normalization and correction wrapper, the slope is -4.8 and the correlation coefficient is 0.77; with the wrapper, the slope is -7.2 with a correlation coefficient as high as 0.88 (coefficients of determination are thus respectively 0.59 and 0.77).
[17]For convenience, we provide also baseline results on the FTB, see (Candito and Seddah, 2010).

	Dev set					Test set				
	LR	LP	F1	Pos acc.	OOVs	LR	LP	F1	Pos acc.	OOVs
Doctissimo										
high noisiness	37.22	41.20	39.11	51.72	40.47	-	-	-	-	-
other	69.68	70.19	69.94	77.96	15.56	70.10	71.68	70.88	79.14	15.42
JeuxVideos.com	66.56	66.46	66.51	74.56	20.46	70.59	71.44	71.02	75.70	19.88
Twitter										
high noisiness	62.07	64.14	63.09	64.89	31.50	54.67	58.16	56.36	64.40	32.84
other	68.06	69.21	68.63	79.70	24.70	71.29	73.45	72.35	78.88	24.47
Facebook										
high noisiness	-	-	-	-	-	55.26	59.23	57.18	54.64	50.40
other	55.90	58.71	57.27	64.34	38.25	60.98	61.79	61.38	70.68	29.52
all	64.13	65.48	64.80	72.69	23.40	66.69	68.50	67.58	74.43	22.81
Ftb	-	-	83.81	96.44	5.2	-	-	84.10	96.97	4.89

Table 7: Baseline parsing results split by sub corpora and noisiness level

word clustering within a PCFG-LA framework. Indeed, we have successfully applied these techniques for French out-of-domain parsing (Candito et al., 2011), as well as for parsing *noisy* English web data (Seddah et al., 2012). On the longer term we intend to apply our normalization and correction module before parsing. The parser will then be provided with corrected tokens, closely matching our regular training data, instead of unedited ones. This will compensate the lack of user generated content large unlabeled corpora, still lacking for French.

7 Conclusion

As mentioned earlier, the *French Social Media Bank* shares with the Google web bank a common will to extend the traditional treebank domain towards user generated content. Although of a smaller scale, it constitutes one of the very first usable resources to validate social media parsing and POS tagging, among the DCU Twitter and football BBC forums treebank (Foster et al., 2011a,b) and the Twitter data set from Gimpel et al. (2011). Moreover, it is the first set of syntactically annotated data for Facebook public web text.

Regarding the Google web bank, the way annotation guidelines had to be extended to deal with user generated content is largely consistent between both treebanks. However, our treebank differs from the Google Web Treebank in several aspects. First, French not only has a morphology richer than English, entailing a tedious disambiguation process when facing *noisy* data. Although the first version of our treebank is smaller than the Google Web Treebank, it includes richer annotations (compound POS, corrected token form of contractions) and includes subcorpora exhibiting a very high level of noise.

To conclude, we presented a new data set devoted on French user generated content. We proposed a first round of evaluation showing that simple techniques could be used to improve POS tagging performance. We presented baseline statistical parsing results, showing that performance on French user generated data were lying far behind those on newspaper in-domain texts. The take home message is that despite what is commonly said, parsing and POS tagging are far from being solved and that working on real text from real users is of crucial importance.

Acknowledgments We thank the reviewers for their insightful comments and Yoav Goldberg, Reut Tsarfaty and Jennifer Foster for their remarks on an earlier version of this work. This work was partly funded by the French ANR project EDyLex (ANR-09-CORD-008).

References

Abeillé, A., Clément, L., and Toussenel, F. (2003). *Building a Treebank for French*. Kluwer, Dordrecht.

Bies, A., Mott, J., Warner, C., and Kulick, S. (2012). English web treebank. Technical report, Linguistic Data Consortium,, Philadelphia, PA, USA.

Candito, M. and Crabbé, B. (2009). Improving generative statistical parsing with semi-supervised word clustering. In *Proceedings of the 11th International Conference on Parsing Technologies (IWPT'09)*, pages 138–141, Paris, France. Association for Computational Linguistics.

Candito, M., Crabbé, B., and Seddah, D. (2009). On statistical parsing of french with supervised and semi-supervised strategies. In *EACL 2009 Workshop Grammatical inference for Computational Linguistics*, Athens, Greece.

Candito, M., Henestroza Anguiano, E., and Seddah, D. (2011). A word clustering approach to domain adaptation: Effective parsing of biomedical texts. In *Proceedings of the 12th International Conference on Parsing Technologies*, pages 37–42, Dublin, Ireland. Association for Computational Linguistics.

Candito, M., Nivre, J., Denis, P., and Anguiano, E. H. (2010). Benchmarking of statistical dependency parsers for french. In *Proceedings of COLING 2010*, Beijing, China.

Candito, M. and Seddah, D. (2010). Parsing word clusters. In *Proceedings of the NAACL/HLT Workshop on Statistical Parsing of Morphologically Rich Languages (SPMRL 2010)*, Los Angeles, CA.

Candito, M. and Seddah, D. (2012). Le corpus sequoia : annotation syntaxique et exploitation pour l'adaptation d'analyseur par pont lexical. In *In Proceedings of Traitement Automatique des Langues Naturelles (TALN 2012)*, Grenoble, France.

Chrupała, G., Dinu, G., and van Genabith, J. (2008). Learning morphology with morfette. In *In Proceedings of LREC 2008*, Marrakech, Morocco. ELDA/ELRA.

Crabbé, B. and Candito, M. (2008). Expériences d'analyse syntaxique statistique du français. In *Actes de la 15ème Conférence sur le Traitement Automatique des Langues Naturelles (TALN'08)*, pages 45–54, Avignon, France.

Denis, P. and Sagot, B. (2009). Coupling an annotated corpus and a morphosyntactic lexicon for state-of-the-art pos tagging with less human effort. In *Proc. of PACLIC*, Hong Kong, China.

Elsner, M. and Charniak, E. (2011). Disentangling chat with local coherence models. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, pages 1179–1189. Association for Computational Linguistics.

Foster, J. (2010). "cba to check the spelling": Investigating parser performance on discussion forum posts. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 381–384, Los Angeles, California. Association for Computational Linguistics.

Foster, J., Cetinoglu, O., Wagner, J., Le Roux, J., Hogan, S., Nivre, J., Hogan, D., and van Genabith, J. (2011a). # hardtoparse: Pos tagging and parsing the twitterverse. In *Workshops at the Twenty-Fifth AAAI Conference on Artificial Intelligence*.

Foster, J., Cetinoglu, O., Wagner, J., Le Roux, J., Nivre, J., Hogan, D., and van Genabith, J. (2011b). From news to comment: Resources and benchmarks for parsing the language of web 2.0. In *Proceedings of 5th International Joint Conference on Natural Language Processing*, pages 893–901, Chiang Mai, Thailand. Asian Federation of Natural Language Processing.

Foster, J., Wagner, J., Seddah, D., and Van Genabith, J. (2007). Adapting wsj-trained parsers to the british national corpus using in-domain self-training. In *Proceedings of the Tenth IWPT*, pages 33–35.

Gildea, D. (2001). Corpus variation and parser performance. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 167–202, Pittsburgh, USA.

Gimpel, K., Schneider, N., O'Connor, B., Das, D., Mills, D., Eisenstein, J., Heilman, M., Yogatama, D., Flanigan, J., and Smith, N. (2010). Part-of-speech tagging for twitter: Annotation, features, and experiments. Technical report, DTIC Document.

Gimpel, K., Schneider, N., O'Connor, B., Das, D., Mills, D., Eisenstein, J., Heilman, M., Yogatama, D., Flanigan, J., and Smith, N. A. (2011). Part-of-speech tagging for twitter: Annotation, features, and experiments. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 42–47, Portland, Oregon, USA. Association for Computational Linguistics.

Green, S. and Manning, C. (2010). Better arabic parsing: Baselines, evaluations, and analysis. In *Proceedings of the 23rd International Conference on Computational Linguistics*, pages 394–402. Association for Computational Linguistics.

Lease, M. and Charniak, E. (2005). Parsing biomedical literature. *Natural Language Processing–IJCNLP 2005*, pages 58–69.

McClosky, D. and Charniak, E. (2008). Self-training for biomedical parsing. In *Proceedings of ACL-08: HLT, Short Papers*, pages 101–104, Columbus, Ohio. Association for Computational Linguistics.

McClosky, D., Charniak, E., and Johnson, M. (2006a). Effective self-training for parsing. In *Proceedings of the Human Language Technology Conference of the NAACL, Main Conference*, pages 152–159, New York City, USA. Association for Computational Linguistics.

McClosky, D., Charniak, E., and Johnson, M. (2006b). Reranking and self-training for parser adaptation. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, pages 337–344. Association for Computational Linguistics.

McClosky, D., Charniak, E., and Johnson, M. (2010). Automatic domain adaptation for parsing. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 28–36. Association for Computational Linguistics.

Petrov, S., Barrett, L., Thibaux, R., and Klein, D. (2006). Learning accurate, compact, and interpretable tree annotation. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, Sydney, Australia. Association for Computational Linguistics.

Petrov, S. and Klein, D. (2007). Improved inference for unlexicalized parsing. In *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; Proceedings of the Main Conference*, pages 404–411, Rochester, New York. Association for Computational Linguistics.

Petrov, S. and McDonald, R. (2012). Overview of the 2012 Shared Task on Parsing the Web. In *Proceedings of the First Workshop on Syntactic Analysis of Non-Canonical Language (SANCL), a NAACL-HLT 2012 workshop*, Montreal, Canada.

Sagot, B. (2010). The L*efff*, a freely available and large-coverage morphological and syntactic lexicon for french. In *Proceedings of LREC'10*, Valetta, Malta.

Sagot, B. and Boullier, P. (2008). SxPipe 2 : architecture pour le traitement présyntaxique de corpus bruts. *Traitement Automatique des Langues (T.A.L.)*, 49(2).

Seddah, D., Candito, M., and Crabbé, B. (2009). Cross parser evaluation and tagset variation: A French Treebank study. In *Proceedings of the 11th Internation Conference on Parsing Technologies (IWPT'09)*, pages 150–161, Paris, France. Association for Computational Linguistics.

Seddah, D., Sagot, B., and Candito, M. (2012). The Alpage Architecture at the SANCL 2012 Shared Task: Robust Preprocessing and Lexical bridging for user-generated content parsing. In *Notes of the First Workshop on Syntactic Analysis of Non-Canonical Language (SANCL)*, Montréal, Canada.

Tsarfaty, R., Seddah, D., Goldberg, Y., Kuebler, S., Versley, Y., Candito, M., Foster, J., Rehbein, I., and Tounsi, L. (2010). Statistical parsing of morphologically rich languages (spmrl) what, how and whither. In *Proceedings of the NAACL HLT 2010 First Workshop on Statistical Parsing of Morphologically-Rich Languages*, pages 1–12, Los Angeles, CA, USA. Association for Computational Linguistics.

Tsarfaty, R. and Sima'an, K. (2008). Relational-realizational parsing. In *Proceedings of the 22nd International Conference on Computational Linguistics-Volume 1*, pages 889–896. Association for Computational Linguistics.

Versley, Y. and Rehbein, I. (2009). Scalable discriminative parsing for german. In *Proceedings of the 11th International Conference on Parsing Technologies (IWPT'09)*, pages 134–137, Paris, France. Association for Computational Linguistics.

Initial explorations on using CRFs for Turkish Named Entity Recognition

Gökhan Akın Şeker[1] Gülşen Eryiğit[2]
(1) ITU Informatics Institute Graduate Program in Computer Science, Istanbul Technical University Istanbul, 34469, Turkey
(2) Department of Computer Engineering, Istanbul Technical University Istanbul, 34469, Turkey
{sekerg, gulsen.cebiroglu}@itu.edu.tr

ABSTRACT
This paper reports the highest results (95% in MUC and 92% in CoNLL metric) in the literature for Turkish named entity recognition; more specifically for the task of detecting person, location and organization entities in general news texts. We give an in depth analysis of the previous reported results and make comparisons with them whenever possible. We use conditional random fields (CRFs) as our statistical model. The paper presents initial explorations on the usage of rich morphological structure of the Turkish language as features to CRFs together with the use of some basic and generative gazetteers.

KEYWORDS: Named Entity Recognition , Turkish , Conditional Random Fields , ENAMEX.

Proceedings of COLING 2012: Technical Papers, pages 2459–2474,
COLING 2012, Mumbai, December 2012.

1 Introduction

Named Entity Recognition (NER) can be basically defined as identifying and categorizing certain type of data (i.e. person, location, organization names, date-time expressions). NER is an important stage for several natural language processing (NLP) tasks including machine translation, sentiment analysis and information extraction. MUC (Sundheim, 1995; Chinchor and Marsh, 1998) and CoNLL (Tjong Kim Sang, 2002; Tjong Kim Sang and De Meulder, 2003) conferences define three basic types of named entities; these are 1- ENAMEX (person, location and organization names), 2- TIMEX (date and time entities) and 3- NUMEX (numerical expressions like money and percentages). But NER research is not limited to only these types; different application areas concentrate to determining alternative entity types such as protein names, medicine names, book titles.

The NER research was firstly started in early 1990s for English. In 1995, with the high interest of the research community, the success rates for English achieved nearly the human annotation performance on news texts (Sundheim, 1995). Nadeau and Sekine (2007) gives a survey of the research for English NER between 1991 to 2006. The satisfaction on English NER task directed the field to new research areas such as multilingual NER systems (Tjong Kim Sang, 2002; Tjong Kim Sang and De Meulder, 2003), NER on informal texts (LIU et al., 2011; Rüd et al., 2011; Mohit et al., 2012), transliteration (Zhang et al., 2012) and coreference (Na and Ng, 2009) of named entities .

Conditional Random Fields (Lafferty et al., 2001) is a very popular method used in NLP. It is also widely used for named entity recognition task in various domains (LIU et al., 2011; Ekbal and Bandyopadhyay, 2009; Settles, 2004). Stanford NER (Finkel et al., 2005) which is a well-known NER tool also uses CRFs as its machine learning method.

Morphologically rich languages pose interesting challenges for NLP tasks as it is the case for NER (Hasan et al., 2009). Turkish being one of such languages attracts attention of the NLP community. Nevertheless, the results for Turkish NER remain still very behind the reported accuracies for English. The first published work on this topic is Cucerzan and Yarowsky (1999) which is a language independent system tested on Romanian, English, Greek, Turkish and Hindi. This system is trained with a small training data and learns from unannotated text using a bootstrapping algorithm. It reports an F-Measure of 53.04% for Turkish. The other studies[1] on Turkish NER are as follows: Tür et al. (2003), Bayraktar and Temizel (2008), Yeniterzi (2011), Özkaya and Diri (2011), Tatar and Cicekli (2011) and Küçük and Yazıcı (2012). Although some of these studies try to use CRFs for Turkish NER task, this is the first study which introduces a successful CRF model which beats the state of the art results for this problem. Yeniterzi (2011) uses CRFs with an IG[2] based tokenization. Özkaya and Diri (2011) reports 84% F-measure on e-mail messages by using CRFs, but since they are using features specific to email domain only (such as from, subject fields) their work may not be extended to general texts.

This is an initial study which aims to propose a successful CRF model for Turkish NER. With this purpose, we firstly focused on general news domain where there exists many reported results for comparison. We obtained the highest scores in the literature on ENAMEX types. We made an initial exploration on the use of morphological features and gazetteers. But we believe there is still room for improvement by using more expansive gazetteers and using the

[1] The studies are investigated in more detail in section §5.3 for comparison.

[2] IG is used for inflectional units smaller than words.

morphological features more efficiently. As a future work, we want to work on TIMEX and NUMEX types as well as informal texts.

This paper is organized as follows. §2 gives brief information about Turkish, §3 explains our framework, §4 gives information about datasets, evaluation metrics and features, §5 gives our experiments and evaluates the results by comparing with related work and §6 gives the conclusion.

2 Turkish

This section briefly states the characteristics of the Turkish language which we believe have impact on the NER task.

Turkish is a morphologically rich and highly agglutinative language. In most of the Turkish NLP studies, lemmas are used instead of word surface forms in order to decrease lexical sparsity. For example a Turkish verb "gitmek" (*to go*) may appear in hundreds of different surface forms[3] depending on the tense, mood and the person arguments whereas the same verb in English has only five different forms (going, go, goes, went, gone). But for the proper nouns, in formal texts, the inflectional suffixes are separated from the lemma by an apostrophe. As a result, although it seems that it is unnecessary to make an automatic morphological processing for the stemming of the proper nouns, the stemming of the surrounding words of the proper nouns has influence on the success of NER. §5 investigates the impact of using lexical information for the named entity recognition task.

Turkish person (first) names are usually selected from the words used in daily conversation such as İpek*(silk)*, Kaya*(rock)*, Pembe*(pink)*, Çiçek *(flower)*. Only the proper nouns and the initial words of the sentences start with an initial capital letter.

Turkish is a free word order language, so the position of the word in a sentence doesn't give us information about being a named entity or not. All of the three sentences: "Ahmet yarın Mehmet ile konuşmaya gidecek.", "Yarın Mehmet ile konuşmaya Ahmet gidecek" and "Yarın Ahmet, Mehmet ile konuşmaya gidecek." are valid Turkish sentences all with the English translation of "Tomorrow, Ahmet will go to talk to Mehmet".

3 Proposed Framework

This section describes our CRF based NER framework trained using morphological features and gazetteers. Figure 1 shows the outline of the framework.

3.1 Tokenization

We tokenized our data so that each word is represented as a token except for proper nouns which go under inflection. Since the suffixes separated by an apostrophe are not part of the named entities(NEs), we partitioned such proper nouns into two tokens (the tokens before and after the apostrophe.) All punctuation characters are considered as a token. Sentences are separated from each other by an empty line. Tokenization of a sample sentence can be seen in Table 2.

[3] Some surface forms of "gitmek" (only in simple present tense for different person arguments): gidiyorum, gidiyorsun, gidiyor, gidiyoruz, gidiyorsunuz, gidiyorlar.

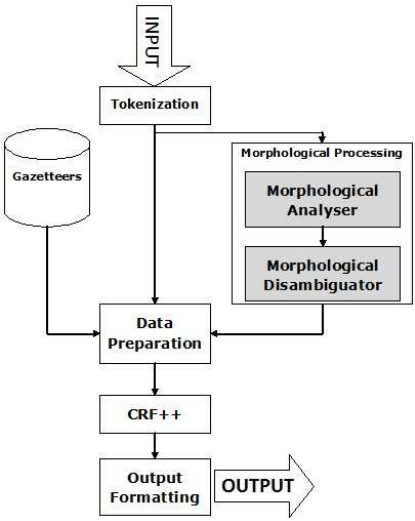


Figure 1: Proposed Framework

3.2 Morphological Processing

We used a two-level morphological analyzer (Oflazer, 1994) for producing the possible analyses for each word. We then give the output to a morphological disambiguator (Sak et al., 2008) in order to get the most probable analysis in the given context. For example, the analyzer produces three different possible analyses for the word "Teknik"(*Technical*) which corresponds to an adjective, a noun and a proper noun accordingly; the disambiguator selects most probable one for us :

Teknik teknik+Adj

Teknik teknik+Noun+A3sg+Pnon+Nom

Teknik teknik+Noun+Prop+A3sg+Pnon+Nom

The output of the analyzer both includes the stem of the word and the morphological features[4] which we use as features for our CRF model. One should keep in mind that, this is an automatic processing and it possesses its own error margin. Eryiğit (2012) gives the performance of this morphological pipeline on raw data.

[4]The abbreviations after the plus sign stand for: +Adj: Adjective, +Noun: Noun, +A3sg: 3sg number-person agreement, +Pnon: Pronoun (no overt possessive agreement), +Nom: Nominative case, +Prop: Proper noun

3.3 Gazetteers

In this work, we prepared two kind of gazetteers[5] which we call base and generator gazetteers. Table 1 gives the details for each one. Base gazetteers are the ones which include words with high probability of occurring in a named entity. These are large gazetteers with thousands of tokens. We collected person names from different sources. We split them into first name and surname gazetteers in order to be able to detect different combinations of these. We compiled the location gazetteer so that it includes all location names in Turkish postal code system[6] , all country names from international telephone code system[7], city and states of those countries[8] and geographical names from different sources.

	Gazetteer	# of tokens
Base	First names	44.048
	Surnames	138.844
	Location names	33.551
Generator	Location	44
	Organization	60
	Person	22

Table 1: # of distinct tokens in gazetteers

Our generator gazetteers are relatively small compared to the base gazetteers. They include the stems of some basic named entity generator words. To give an example: the stem "bakanlık" (*ministry*) which could come after some regular words such as spor, tarım (*sports, agriculture*) to construct organization NEs such as "Tarım Bakanlığı" (*Ministry of Agriculture*).

3.4 Data Preparation

In this stage, we use the information coming from the raw data, the gazetteers and the morphological processing in order to prepare the feature vectors for our training/test instances. For the related class labels at the training stage, we use "Raw Tags". In this format, we use the labels "PERSON", "ORGANIZATION", "LOCATION" and "O" (other - for the words which do not belong to a NE) without any position information (that is without any prefix). In §5.1, we also give the results of our experiments of using different formats for class labels.

3.5 Conditional Random Fields

Conditional random fields (CRFs) (Lafferty et al., 2001) is a framework for building probabilistic models to segment and label sequence data. CRFs offer several advantages over hidden Markov models (HMMs), stochastic grammars and maximum entropy Markov models (MEMMs). CRF is a discriminative model better suited to including rich, overlapping features focusing solely on the conditional distribution $p(\mathbf{y}|\mathbf{x})$. We use linear chain CRFs where $p(\mathbf{y}|\mathbf{x})$ is defined as:

$$p_\theta(\mathbf{y}|\mathbf{x}) = \frac{1}{Z_\theta(\mathbf{x})} \exp\left\{\sum_{t=1}^{T}\sum_{k=1}^{K}\theta_k f_k(y_{t-1}, y_t, x_t)\right\} \quad (1)$$

[5]available from http://web.itu.edu.tr/gulsenc/ner.html
[6]https://interaktifkargo.ptt.gov.tr/posta_kodu/
[7]http://www.ttrehber.turktelekom.com.tr/trk-web/ulkekodlari.html
[8]mostly collected from wikipedia.com

where $f_k(y_{t-1}, y_t, x_t)$ is the function for the properties of transition from the state y_{t-1} to y_t with the input x_t and θ_k is the parameter optimized by the training. $Z_\theta(\mathbf{x})$ is a normalization factor calculated by:

$$Z_\theta(\mathbf{x}) = \sum_{\mathbf{y} \in Y^T} \exp \left\{ \sum_{t=1}^{T} \sum_{k=1}^{K} \theta_k f_k(y_{t-1}, y_t, x_t) \right\} \quad (2)$$

For the named entity task, each state y_t is the named entity label and each feature vector x_t contains all the components of the global observations x that are needed for computing features at time t. Sutton and McCallum (2011) gives detailed information on mathematical foundations and many examples about the usage of CRFs. In this study we used CRF++[9] which is an open source implementation of CRFs.

3.6 Output Formatting

Our system has the capability of labeling the output with two different type of tags: 1. Raw tags and 2. IOB2 tags. Raw tag format which is introduced in §3.4 is also used during the training. The experiments related to the selection of the training format is given in §5.1. IOB2 (Tjong Kim Sang, 2002) is one of the most common formats used in the literature for labeling named entities. I-O-B denotes **I**n, **O**ut and **B**egin. In this tagging format, the first token of a NE is labeled with a "B-" prefix while other words in NE are labeled with an "I-" prefix. Tokens that are not part of any NE are tagged with the label "O". Table 2 shows a sample tokenized sentence (*Mustafa Kemal Atatürk went to Samsun in 1919.*) tagged in both formats.

Token	IOB2 Tags	RAW Tags
Mustafa	B-PERSON	PERSON
Kemal	I-PERSON	PERSON
Atatürk	I-PERSON	PERSON
1919	O	O
yılında	O	O
Samsun	B-LOCATION	LOCATION
'a	O	O
çıktı	O	O
.	O	O

Table 2: IOB2 tagging vs RAW tagging

4 Configuration

This section gives detailed information about the used datasets, evaluation metrics, feature categories and feature templates. §4.3 (Feature categories) presents the features which are provided for each token in our input file. §4.4 (Feature templates) presents the templates used for creating CRF feature vectors for each instance, using the given categories for the current token and its context together with some combinations of these.

[9] http://crfpp.googlecode.com/svn/trunk/doc/index.html

4.1 Data Sets

In our experiments we used the data from Tür et al. (2003). This data consists of 500K words and is annotated only for ENAMEX types with 24,101 person names, 16,105 location names and 13,540 organization names. We reserved one tenth of the data (47,344 words) for testing and used the remaining for training purposes by exactly the same way in Yeniterzi (2011).

4.2 Evaluation Metrics

There are two main metrics in the literature for the evaluation of NER systems: CoNLL and MUC. The MUC metric is the average F-Measure of MUC TEXT and MUC TYPE. MUC TYPE evaluates the performance of assigning the correct NE type to each word without taking into account if the NE boundaries are detected correctly. MUC TEXT makes evaluation only on NE boundaries without looking if the correct NE type is assigned or not. The CoNLL metric on the other hand evaluates an assignment to be correct if both the type and the boundary of a NE is determined correctly. The details of the calculation for these metrics may be investigated from Nadeau and Sekine (2007).

Although CoNLL metric seems to be preferred in recent studies, we evaluate[10] our results using both CoNLL and MUC in order to be able to make comparisons with previous works.

4.3 Feature Categories

In our **base model (BM_surf)** we used word tokens converted to lower case in their surface form. The idea behind converting tokens to lowercase is avoiding one of the major problems of the Turkish language studies; the sparse data problem. Other features added to this model can be grouped into three main categories: morphological, lexical and gazetteer lookup features.

4.3.1 Morphological Features

The morphological features are extracted from the analysis produced after the automatic morphological processing of each word.

Stem : The stem information. For the inflected proper nouns where the inflections after the apostrophe are treated as a separate token, the same surface form after the apostrophe is assigned as the stem of the token representing inflections.

Part of Speech Tag (POS) : The final part of speech category for each word. In Turkish, with the use of derivations, words may change their part of speech categories within a single surface form. The final form of the word determines its syntactic role within a sentence. Therefore, we use the final POS form of each word. We assigned a special POS tag ("APOST") to the tokens separated by an apostrophe from the proper nouns.

Noun Case (NCS) : The case argument. This feature is 0 for non nominal tokens and one of the following values for nominals: Nominative(NOM), Accusative/Objective(ACC), Dative (DAT), Ablative(ABL), Locative(LOC), Genitive(GEN), Instrumental(INS), Equative(EQU). Ex: the value will be NOM for the word "Teknik" with the morphological analysis "teknik+Noun+Prop+A3sg+Pnon+Nom".

[10] We use the evaluation script from CoNLL 2000 shared task (http://www.cnts.ua.ac.be/conll2000/chunking/output.html) for CoNLL and MUC TYPE scores (with the option "-r").

Proper Noun (PROP) : A binary feature indication that the "+Prop" tag exists (1) in the selected morphological analysis or not (0). Ex: The value will be 1 for the word "Teknik" given above. It is useful to mention that the morphological pipeline tags all unknown words as proper nouns.

All Inflectional Features (INF) : All inflectional tags after the POS category. If a derivation exists then the inflectional tags after the last derived POS category is used. Ex: the value will be "Prop+A3sg+Pnon+Nom" for the word "Teknik" with the above morphological analysis.

4.3.2 Lexical Features

Case Feature (CS) : The information about lowercase and uppercase letters used in the current token. This feature takes 4 different values: lowercase(0), UPPERCASE(1), Proper Name Case(2) and miXEd CaSe(3)

Start of the Sentence (SS) : A binary feature indicating that the current token is the beginning of a sentence (1) or not (0).

4.3.3 Gazetteer Lookup Features

Six different features used for each of the six gazetteers introduced in §3.3. Lookup features for base gazetteers (BG) have a 1 value if the token exists in the corresponding gazetteer and 0 otherwise. Generator gazetteer lookup features (GG) are binary features as well but this time the stem of the word is checked instead of the full surface form.

4.4 Feature Templates

CRFs are log-linear models. In order to take advantage of the useful feature combinations, one needs to provide these as new features to the CRFs. In some studies, it is shown that the useful feature conjunctions may be determined incrementally and provided to the system automatically (McCallum, 2003). But, in this study, we used the approach proposed in Sha and Pereira (2003) and selected useful features manually for our initial explorations. Although this approach generally results with a huge number of features, we didn't have any memory problem by using the combinations.

We provided our atomic features within a window of {-3,+3} and some selected combinations of these as feature templates to CRF++. Two sample feature templates are given in the below example. The templates are given in [pos,col] format, where pos stands for the relative position of the token in focus and col stands for the feature column number in the input file.

$$U15 : \%x[-2,2]$$

$$U50 : \%x[0,10]/\%x[0,6]$$

U15 is the template for using the 2nd feature (part-of-speech tag) of the second previous word. U50 is the template for using the conjunction of the existence of the current word in the location name gazetteer (LG) (col=10) and its case feature (col=6) such as *exists in LG written in lowercase; exists in LG and the first letter is capitalized;...*

We use the bigram option of the CRF++ in order to automatically generate the edge features using the previous label y_{-1} and the current label y_0.

As a result, for the 14 feature categories presented in the previous section, we formulated 92 feature templates[11] in our final model which resulted to $\sim$ 20M binary features.

5 Experiments & Evaluation

This section comprises the experiments we conducted to make the decision for our training format (§5.1), to measure the impact of our proposed models (§5.2) and to compare our best model with previous studies (§5.3).

5.1 Training format experiments

We experimented with different training data formats. These are IOB, IOB2, raw labels and fictitious boundary model of Tür et al. (2003). In all of these experiments, we converted the test output to IOB-2 style and evaluated in the same manner. This conversion is made in a straightforward manner by assuming the consecutive tokens labeled with the same NE type as the part of the same NE. This is an acceptable assumption since in Turkish, words in a sentence are separated with a comma if they have the same syntactic function. And also, the subject of a sentence is written separately by the use of a comma if it appears in a confusing context. Tür et al. (2003) gives the probability of two consecutive tokens to have both "B-PERSON" tags as 0.006076. Our training and test sets do not include such NEs at all.

Our experiments show that, we obtain the highest performance by using the RAW labels whereas using the IOB formats reduces the performance by 0.4% and the fictitious boundary format by 2% in all metrics in our base model.

5.2 Feature-related experiments

	MUC TYPE				MUC TEXT	MUC
Feature	PER	ORG	LOC	Overall		
BM_stem	85.31	79.89	86.87	84.03	83.95	83.54
BM_surf	83.83	82.71	86.67	84.19	85.81	85.00
+STEM	85.62	83.26	87.26	85.30	87.08	86.19
+POS	87.34	83.08	87.47	86.06	88.11	87.09
+NCS	87.46	83.95	87.27	86.33	88.85	87.59
+PROP	88.87	85.12	88.68	87.68	90.98	89.33
+INF	89.65	85.32	89.79	88.38	91.60	89.99
+CS	92.76	89.09	89.85	90.92	94.73	92.83
+SS	92.75	89.01	90.15	90.97	94.68	92.83
+BG	94.00	89.82	92.20	92.27	95.50	93.89
+GG	**94.81**	**91.09**	**93.35**	**93.29**	**95.89**	**94.59**

Table 3: F-measure in MUC TYPE, MUC TEXT and MUC Metrics

In our base model (BM_surf) we trained CRF using only one feature; the surface form of the word that appeared in the sentence. Tokens are converted to lowercase to avoid sparse data

[11] The used templates and our NER tool is available from http://web.itu.edu.tr/gulsenc/ner.html

Feature	PER	ORG	LOC	Overall
BM_stem	81.84	74.94	86.82	81.78
BM_surf	80.77	77.86	87.66	82.28
+STEM	82.76	78.78	87.95	83.47
+POS	84.75	78.16	88.39	84.33
+NCS	84.65	79.08	88.00	84.38
+PROP	85.90	80.61	89.20	85.69
+INF	86.71	81.97	89.88	86.59
+CS	90.65	86.12	90.74	89.59
+SS	90.46	85.95	91.01	89.55
+BG	92.23	87.28	92.14	91.02
+GG	**92.94**	**88.77**	**92.93**	**91.94**

Table 4: F-measure in CONNL Metric

problem. i.e. Our training data includes the name of a city "AKSARAY" as a token only one time and doesn't include "Aksaray". This usage converted both to the same token "aksaray". The disadvantage of this usage is losing the effect of an important property of proper nouns in Turkish; the first letter of a proper noun is capitalized so that the noun "gül" (rose) differs from the person name "Gül". Effect of converting all tokens to lowercase is about 1% F-Measure loss in base model in our experiments, but this loss is recoverable with added case (CS) feature. We also wanted to see the performance of using only the word stems instead of the surface form in the sentence so generated a second model (BM_stem).

Table 3 and 4 give detailed evaluation results in all metrics. A plus(+) sign before the feature name indicates that this feature is added (together with suitable feature templates) to the model at the above line. These tables also show the contribution of each feature. While adding a new feature to the model, we also made experiments using different feature template combinations for this new feature to determine the best n-gram relations and their interaction with the previously added features. These experiments are not added to the paper due to the space constraints.

From Table 3 and 4, it can be seen that BM_surf has higher performance than BM_stem. But using the stem information (+STEM) together with the BM_surf raises the performance. This supports our claim about the influence of the stemming of the surrounding words in §2.

All added morphological features raised the performance. But the performance gain acquired by the +INF feature is surprising. One should keep in mind that the +NCS and +PROP features are the atomic units extracted from the +INF feature. Since Turkish is an agglutinative language, the possible number of different values for the +INF feature is very high. For this reason, in many recent studies the usage of the inflectional features as a block[12] is not a preferred approach. We think this result indicates that there is still room for improvement using more fine grained usage of these inflectional features.

The high increase obtained by the +CS feature is as expected due to the known disadvantage of our lowercased base model. But the low performance (and some negative effects) of the feature SS is surprising.

Our base gazetteer features (+BG) performed under our expectations but added a gain of

[12] many atomic features concatenated to each other

about 1% in all metrics. An interesting observation about these features is the gain on organization name identification performance. We don't have any gazetteer of organization names, but using our gazetteers raised the number of identified person and location names. As a result, the number of false positive identifications of organization names decreased.

Impressive performance of the generator gazetteers (+GG) with their modest sizes encourages us adding more of such gazetteers for future work.

5.3 Comparison with related work

This section tries to make a detailed analysis on the related studies: At the time of writing of this paper none of the tools were publicly available so that it wasn't possible to train and test them on the same data set. Table 5 gives the reported results of each related work. We give the results of our pairwise comparisons in the running text whenever possible.

Related work	Best Result	Ev.Metr.	Domain	NE Types
Özkaya and Diri (2011)	84.24	*n/a*	E-mail texts	ENAMEX
Küçük and Yazıcı (2012)	90.13	OTHER	General news	ENAMEX,TIMEX,NUMEX
Tür et al. (2003)	91.56	MUC	General news	ENAMEX
Bayraktar and Temizel (2008)	81.97	MUC	Financial Texts	PERSON NAMES
OURS	**94.59**	**MUC**	**General news**	**ENAMEX**
Tatar and Cicekli (2011)	91.08	CoNLL	Terrorism news	ENAMEX,TIMEX
Yeniterzi (2011)	88.94	CoNLL	General news	ENAMEX
OURS	**91.94**	**CoNLL**	**General news**	**ENAMEX**

Table 5: Comparison with related work (The reported results in each paper)

The performances listed in Table 5 is organized in decreasing order of credit given to partial matches during evaluation. Most of the results are on MUC and CoNLL metrics, therefore we listed our results twice in both of these. Note that the test sets, evaluation metrics (3^{rd} column), working domain (4^{th} column) and entity types[13] (5^{th} column) in focus of each work are different from each other. Table 5 tries to give an overview of these features for each work.

The first NER work specific to Turkish is Tür et al. (2003). The study focuses on three Information Extraction (IE) tasks, namely, sentence segmentation, topic segmentation and name tagging. For name tagging task they use lexical, morphological and contextual features of the words to generate an HMM based model. They evaluate their results in MUC metrics. They use the same training data, but different test data which is not available. Their performance (91.56%) given in Table 5 are comparable with our result in MUC metrics (94.59%)

Bayraktar and Temizel (2008) work on financial texts to find person names. They apply the local grammar based approach of Traboulsi (2006) to Turkish. They construct a list of Turkish reporting verbs. Since they work on a different domain focusing only to person names, their results are not comparable with none of the related work given in this section.

Küçük and Yazıcı (2012) adds statistical methods (Rote learning- (Freitag, 2000)) to their rule based study (Küçük and Yazıcı, 2009) raising the F-measure on general news text from 87.96

[13]Detailed information on the terms ENAMEX(person, organization and location names), TIMEX(dates and times) and NUMEX(currency values and percentages) included in NE Types column can be found at (Sundheim, 1995).

to 90.13. This study is the only published work of Turkish NER comparing performances for different domains. They evaluate their system on general news texts, financial news texts, historical texts and child stories. In Table 5 we took the results on general news texts domain which sounds similar to our domain. Their evaluation metric gives more credit to partial matches and not comparable with none of our metrics. They work on ENAMEX, TIMEX and NUMEX entity types but they do not provide the scores for each of these. In order to be able to make a fair comparison between the two studies, we measure the performance of their system on our test data and calculate the overall ENAMEX performance (F-Measure) as 69.78% in CoNLL metrics and 74.59% in MUC TYPE metrics. We think the reasons of the observed difference between the performances reported in their work and on our tests are the evaluation criteria, the working test domain (our dataset consists of older news texts) and the performance drop due to the lack of TIMEX and NUMEX types (where they have higher performances).

Tatar and Cicekli (2011) propose an automatic rule learning system exploiting morphological features. Although they don't namely mention that they use the CoNLL metric, the evaluation strategy of looking for the exact match is compatible with the CoNLL metric. Their overall score includes the performance on dates and time expressions which is higher than the performance for the NE types of our interest. Their reported accuracy is 91.08% on ENAMEX and TIMEX types. The relevant F-measure for only ENAMEX types is calculated as 90.63%; this result is comparable with our reported F-measure 91.94% in CoNLL metric (except the fact that the evaluations are made on different test sets).

Yeniterzi (2011) uses CRFs and exploits the impact of morphology for Turkish NER. In this work, she uses the inflectional units (IG) as tokens. This work is the one which is most similar to ours but we use morphological features in a different way and add the use of gazetteers. We use the same training and test data, so our results given in CoNLL metrics are fully comparable with this work. One should note that our performance before adding the gazetteers (89.55%) is still higher than her best result (88.94%) which shows that the increase may not be credited to only to the use of gazetteers.

Özkaya and Diri (2011) also uses CRFs on informal texts so it is not fair to compare the results with any of the work discussed in this section all working on formal texts. They do not provide their evaluation metrics and their overall results, but we calculate overall precision, recall and F-measure values as 92.89%, 77.07% and 84.24 respectively using the token counts provided in their paper.

6 Conclusion & Future Work

In this work we presented a Turkish NER model using conditional random fields trained with morphological and lexical features. We compiled large scale person and location names gazetteers which will be available for the researchers. We obtained state of the art results which we believe will act as the baseline for future Turkish NER research. We also tried to compare the results and the evaluation metrics of recent NER work in Turkish. We believe this is also an important contribution since the results given in previous works were not comparable because of first, different evaluation metrics giving different credits to partial matches were used and second, the studies focused to different sets of named entity types and provided their results as the average of these. As future work, we will test our model in different formal and informal domains, investigate the ways to better use the morphological properties collected in the inflectional (INF) feature such as using them as atomic units. We aim to im-

prove our model extending and adding new generator gazetteers. We also aim to add NUMEX and TIMEX entity types to our system. The automatic determination of the useful features and their combinations is another subject of our future work.

Acknowledgments

The authors want to thank the following people without whom it would be impossible to produce this work: Reyyan Yeniterzi and Ilyas Çiçekli for providing their datasets, Gökhan Tür for the helpful discussions, Dilek Küçük and Adnan Yazıcı for processing the test data with their NER tool.

References

Bayraktar, O. and Temizel, T. T. (2008). Person Name Extraction From Turkish Financial News Text Using Local Grammar Based Approach. In *23rd International Symposium on Computer and Information Sciences (ISCIS'08)*, Istanbul. ISBN 978-1-4244-2880-9 electronic version (4 pp.).

Chinchor, N. A. and Marsh, E. (1998). Muc-7 information extraction task definition. In *Proceeding of the Seventh Message Understanding Conference (MUC-7), Appendices*.

Cucerzan, S. and Yarowsky, D. (1999). Language independent named entity recognition combining morphological and contextual evidence. In *In Proceedings of the joint SIGDAT conference on empirical methods in natural language processing and very large corpora*.

Ekbal, A. and Bandyopadhyay, S. (2009). A conditional random field approach for named entity recognition in Bengali and Hindi. *Linguistic Issues in Language Technology*, 2(1).

Eryiğit, G. (2012). The impact of automatic morphological analysis & disambiguation on dependency parsing of turkish. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC)*, Istanbul, Turkey.

Finkel, J. R., Grenager, T., and Manning, C. (2005). Incorporating non-local information into information extraction systems by gibbs sampling. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, ACL '05, pages 363–370, Stroudsburg, PA, USA. Association for Computational Linguistics.

Freitag, D. (2000). Machine learning for information extraction in informal domains. *Machine Learning*, 39(2/3):169–202.

Hasan, K. S., Rahman, A., and Ng, V. (2009). Learning-based named entity recognition for morphologically-rich, resource-scarce languages. In *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics*, pages 354–362.

Küçük, D. and Yazıcı, A. (2009). Named entity recognition experiments on turkish texts. In *Proceedings of the 8th International Conference on Flexible Query Answering Systems*, FQAS '09, pages 524–535, Berlin, Heidelberg. Springer-Verlag.

Küçük, D. and Yazıcı, A. (2012). A hybrid named entity recognizer for turkish. *Expert Syst. Appl.*, 39(3):2733–2742.

Lafferty, J. D., McCallum, A., and Pereira, F. C. N. (2001). Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *ICML*, pages 282–289.

LIU, X., ZHANG, S., WEI, F., and ZHOU, M. (2011). Recognizing named entities in tweets. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 359–367, Portland, Oregon, USA. Association for Computational Linguistics.

McCallum, A. (2003). Efficiently inducing features of conditional random fields. In *UAI*, pages 403–410.

Mohit, B., Schneider, N., Bhowmick, R., Oflazer, K., and Smith, N. A. (2012). Recall-oriented learning of named entities in arabic wikipedia. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 162–173, Avignon, France. Association for Computational Linguistics.

Na, S.-H. and Ng, H. T. (2009). A 2-poisson model for probabilistic coreference of named entities for improved text retrieval. In *Proceedings of the 32nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2009*, pages 275–282.

Nadeau, D. and Sekine, S. (2007). A survey of named entity recognition and classification. *Linguisticae Investigationes*, 30(1):3–26. Publisher: John Benjamins Publishing Company.

Oflazer, K. (1994). Two-level description of Turkish morphology. *Literary and Linguistic Computing*, 9(2):137–148.

Özkaya, S. and Diri, B. (2011). Named entity recognition by conditional random fields from turkish informal texts. In *Proceedings of the IEEE 19th Signal Processing and Communications Applications Conference (SIU 2011)*, pages 662–665.

Rüd, S., Ciaramita, M., Müller, J., and Schütze, H. (2011). Piggyback: Using search engines for robust cross-domain named entity recognition. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 965–975, Portland, Oregon, USA. Association for Computational Linguistics.

Sak, H., Güngör, T., and Saraçlar, M. (2008). Turkish language resources: Morphological parser, morphological disambiguator and web corpus. In *GoTAL 2008*, volume 5221 of *LNCS*, pages 417–427. Springer.

Settles, B. (2004). Biomedical named entity recognition using conditional random fields and rich feature sets. In *Proceedings of the International Joint Workshop on Natural Language Processing in Biomedicine and its Applications*, JNLPBA '04, pages 104–107, Stroudsburg, PA, USA. Association for Computational Linguistics.

Sha, F. and Pereira, F. C. N. (2003). Shallow parsing with conditional random fields. In *HLT-NAACL*.

Sundheim, B. (1995). Overview of results of the muc-6 evaluation. In *MUC*, pages 13–31.

Sutton, C. and McCallum, A. (2011). An introduction to conditional random fields. *Foundations and Trends in Machine Learning*. To appear.

Tatar, S. and Cicekli, I. (2011). Automatic rule learning exploiting morphological features for named entity recognition in turkish. *Journal of Information Science*, 37(2):137–151.

Tjong Kim Sang, E. F. (2002). Introduction to the conll-2002 shared task: Language-independent named entity recognition. In *Proceedings of CoNLL-2002*, pages 155–158.

Tjong Kim Sang, E. F. and De Meulder, F. (2003). Introduction to the conll-2003 shared task: Language-independent named entity recognition. In *Proceedings of CoNLL-2003*, pages 142–147.

Traboulsi, H. N. (2006). *Named Entity Recognition: A Local Grammar-based Approach*. PhD thesis, Department of Computing School of Electronics and Physical Sciences University of Surrey.

Tür, G., Hakkani-Tür, D., and Oflazer, K. (2003). A statistical information extraction system for turkish. *Natural Language Engineering*, 9:181–210.

Yeniterzi, R. (2011). Exploiting morphology in turkish named entity recognition system. In *Proceedings of the ACL 2011 Student Session*, pages 105–110, Portland, OR, USA.

Zhang, M., Li, H., Liu, M., and Kumaran, A. (2012). News 2012 shared task on machine transliteration. In *Proceedings of the 4th Named Entities Workshop 2012 at ACL 2012*.

Differential Evolution based Feature Selection and Classifier Ensemble for Named Entity Recognition

Utpal Kumar Sikdar Asif Ekbal* Sriparna Saha*
Department of Computer Science and Engineering
Indian Institute of Technology Patna, Patna, India
{utpal.sikdar,asif,sriparna}@iitp.ac.in

ABSTRACT
In this paper, we propose a differential evolution (DE) based two-stage evolutionary approach for named entity recognition (NER). The first stage concerns with the problem of relevant feature selection for NER within the frameworks of two popular machine learning algorithms, namely Conditional Random Field (CRF) and Support Vector Machine (SVM). The solutions of the final best population provides different diverse set of classifiers; some are effective with respect to recall whereas some are effective with respect to precision. In the second stage we propose a novel technique for classifier ensemble for combining these classifiers. The approach is very general and can be applied for any classification problem. Currently we evaluate the proposed algorithm for NER in three popular Indian languages, namely Bengali, Hindi and Telugu. In order to maintain the domain-independence property the *features are selected and developed mostly without using any deep domain knowledge and/or language dependent resources*. Experimental results show that the proposed two stage technique attains the final F-measure values of 88.89%, 88.09% and 76.63% for Bengali, Hindi and Telugu, respectively. The key contributions of this work are two-fold, *viz.* (i). proposal of differential evolution (DE) based feature selection and classifier ensemble methods that can be applied to any classification problem; and (ii). scope of the development of language independent NER systems in a resource-poor scenario.

KEYWORDS: Named Entity Recognition, Differential Evolution, Feature Selection, Classifier Ensemble.

Authors equaly contributed for this paper. * corresponding authors

Proceedings of COLING 2012: Technical Papers, pages 2475–2490,
COLING 2012, Mumbai, December 2012.

1 Introduction

Named Entity Recognition (NER) aims to identify and classify each word of a document into some predefined target categories such as person, location, organization, miscellaneous (e.g., date, time, number, percentage, monetary expressions etc.) and none-of-the-above. Over the decades, it has shown success in almost all application areas of Natural Language Processing (NLP) that includes information retrieval, information extraction, machine translation, question-answering and automatic summarization etc. Proper identification and classification of NEs are very crucial and pose a big challenge to the NLP researchers. The main challenge is due to the fact that named entity (NE) expressions are hard to analyze using traditional NLP because they belong to the open class of expressions, i.e., there is an infinite variety and new expressions are constantly being invented.

The problem of NER was actually formulated in Message Understanding Conferences (MUCs) [MUC6; MUC7] (Chinchor, 1995, 1998). The issues of correct identification of NEs were specifically addressed and benchmarked by the developers of information extraction system, such as the GATE system (Cunningham, 2002). The existing approaches for NER can be grouped into three main categories, namely rule-based, machine learning based and hybrid approach. Majority of the research focussed on machine learning (ML) approaches (Bikel et al., 1999; Borthwick, 1999; Sekine, 1998; Lafferty et al., 2001a; Yamada et al., 2001) because these are easily trainable, adaptable to different domains and languages as well as their maintenance are also being less expensive. In contrast, rule-based approaches lack the ability of dealing with the problems of robustness and portability. Each new source of text requires significant updates to the rules to maintain optimal performance and the maintenance costs could be quite steep.

Literature shows that most of the works carried out in this direction cover mostly English and European languages. There are also significant amount of works in some of the Asian languages like Chinese, Japanese and Korean. India is a multilingual country with great linguistic and cultural diversities. People speak in 22 different official languages that are derived from almost all the dominant linguistic families. Some works on NER for Indian languages can be found in (Ekbal and Saha, 2011; Ekbal and Bandyopadhyay, 2008; Ekbal et al., 2007; Ekbal and Bandyopadhyay, 2009b, 2007; Ekbal et al., 2008; Li and McCallum, 2004; Patel et al., 2009; Srikanth and Murthy, 2008; Shishtla et al., 2008; Vijayakrishna and Sobha, 2008). However, the works related to NER in Indian languages are still in the nascent stages due to the potential facts such as (Ekbal and Saha, 2011):

- Unlike English and most of the European languages, Indian languages lack capitalization information, which plays a very important role in NE identification.
- Indian person names are more diverse compared to the other languages and a lot of these words can be found in the dictionary with specific meanings.
- Indian languages are highly inflectional language providing one of the richest and most challenging sets of linguistic and statistical features resulting in long and complex word-forms.
- Indian languages do not conform to any fixed word-order.
- Indian languages are resource poor in nature- annotated corpora, name dictionaries, good morphological analyzers, Part-of-Speech (PoS) taggers etc. are not yet available in the required measure.
- Although Indian languages have a very old and rich literary history, technological devel-

opments are of recent origin.

- Web sources for name lists are available in English, but such lists are not available in Bengali, Hindi and Telugu forcing the use of transliteration.

In this paper we propose a two-stage approach for NER in Indian languages. The first step solves the problems of feature selection for NER within the frameworks of two robust machine learning algorithms, namely Conditional Random Field (CRF)(Lafferty et al., 2001a) and Support Vector Machine (SVM) (Vapnik, 1995). The performance of any classification technique depends on the features of training and test datasets. Feature selection, also known as variable selection, feature reduction, attribute selection or variable subset selection, is the technique, commonly used in machine learning, of selecting a subset of relevant features for building robust learning models. In a machine learning approach, feature selection is an optimization problem that involves choosing an appropriate feature subset. In CRF or SVM based models, relevant feature selection is a very crucial problem and also a key issue to improve the classifier's performance. Usually, heuristics are used to find the appropriate set of features in these classification models. In this paper we propose an evolutionary algorithm for automatic feature selection. The second stage deals with the problem of classifier ensemble. Classifier ensemble [1] is relatively a new direction of machine learning. The main idea behind classifier ensemble is that ensembles are often much more accurate than the individual classifiers that make them up. It is to be noted that all the existing ensemble techniques should have a way of combining the decisions of a set of classifiers. Existing approaches (e.g., stacking, AdaBoost, bagging etc.) combine the outputs of all classifiers by using either majority voting or weighted voting. The weights of votes depend on the error rate/performance of the individual classifiers. However, in reality, in an ensemble system all the classifiers are not equally efficient in detecting all types of output classes. Thus, while combining the classifiers using weighted voting, weights of voting should vary among the different output classes in each classifier. The weight should be high for that particular output class for which the classifier performs good. Otherwise, weight should be low for the output class for which its output is not very reliable. So, it is a crucial issue to select the appropriate weights of votes for all the classes in each classifier. The proposed algorithms for feature selection and classifier ensemble are based on an evolutionary algorithm, differential evolution (DE) (Storn and Price, 1997). To train and test the classifiers we use a set of features that are *mostly selected and developed without using any deep domain knowledge and/or language dependent resources*. The first stage of the algorithm produces a population that contains a set of solutions, some of them are effective with respect to recall whereas some are effective with respect to precision. In the second stage we combine the decisions of these solutions using the proposed DE based classifier ensemble technique. The proposed approach is evaluated for three resource-poor Indian languages, namely Bengali, Hindi and Telugu. Bengali is the seventh most spoken language in the world, second in India and the national language of Bangladesh. Hindi is the third most spoken language in the world and the national language of India. Results show that this two stage DE based technique attains the final F-measure values of 88.89%, 88.09% and 76.63% for Bengali, Hindi and Telugu, respectively. The proposed approach is compared with the conventional majority and weighted voting techniques, popular ensemble methods like stacking, AdaBoost and Error Correcting Output Codes. We also compare our proposed method with two ensemble techniques (Ekbal and Saha, 2010, 2011) based on evolutionary genetic algorithm. Our analysis shows that the proposed approach can attain superior performance in comparison to the exist-

[1] We use 'ensemble classifier' and 'classifier ensemble' interchangeably

ing methods. The key contributions of this work are two-fold, *viz.* (i). proposal of differential evolution (DE) based feature selection and classifier ensemble methods that can be applied to any classification problem; and (ii). scope of the development of language independent models for NER in a resource-poor scenario.

2 Overview of Differential Evolution

Differential Evolution (DE) (Storn and Price, 1997) is a parallel direct search method which performs search in complex, large and multi-modal landscapes, and provides near-optimal solutions for objective or fitness function of an optimization problem. In DE, the parameters of the search space are encoded in the form of strings called chromosomes. A collection of such strings is called a population denoted by P. It is a collection of NP number of d-dimensional parameter vectors $x_{i,G} = [x_{1,i,G}, x_{2,i,G}, \ldots, x_{D,i,G}]$ $, i = 1, 2, \ldots, NP$ for each generation G. The value of D represents the number of real parameters on which optimization or fitness function depends. The value of NP does not change during the minimization process. The initial vector population is chosen randomly which represents different points in the search space and should cover the entire parameter space. An objective or a fitness function is associated with each string that represents the degree of goodness of the string. Differential evolution generates new parameter vectors by adding the weighted difference between two population vectors to a third vector. This operation is called mutation. The mutated vector's parameters are then mixed with the parameters of another predetermined vector, the target vector, to yield the so-called trial vector. Parameter mixing is often referred to as "crossover". If the trial vector yields a lower cost function value than the target vector, the trial vector replaces the target vector in the following generation. This last operation is called selection. The process of selection, crossover and mutation continues for a fixed number of generations or till a termination condition is satisfied.

3 Proposed Method for Feature Selection

In this section we firstly formulate the problem of relevant feature selection within the framework of differential evolution, and then present the proposed approach.

3.1 Problem Formulation for Feature Selection

Suppose, the M number of available features for a given classifier are denoted by $F_1, \ldots, F_M$. Let, $\mathscr{A} = \{F_i : i = 1; M\}$. The feature selection problem is then stated as follows:

Find the appropriate subset of features $\mathscr{A}' \subseteq \mathscr{A}$ such that the classifier trained using these features should have optimized some metrics. In this work we optimize the F-measure value which is the combination of both recall and precision.

3.2 Chromosome Representation and Population Initialization

If the total number of features is F, then the length of the chromosome is F. As an example, the encoding of a particular chromosome is represented in Figure 1. Here, $\mathscr{F} = 12$ (i.e., total 12 different features are available). The chromosome represents the use of 7 features for constructing a classifier (first, third, fourth, seventh, tenth, eleventh and twelfth features). The entries of each chromosome are randomly initialized to either 0 or 1. Here, if the i^{th} position of a chromosome is 0 then it represents that i^{th} feature does not participate in constructing the classifier. Else, if it is 1 then the i^{th} feature participates in constructing the classifier.

If the population size is P then all the P number of chromosomes of this population are initialized in the above way.

3.3 Fitness Computation

For the fitness computation, the following steps are executed.

1. Suppose, there are N number of features present in a particular chromosome (i.e., there are total N number of 1's in that chromosome).
2. Construct a classifier with only these N features.
3. Here, initially the training data is divided into 3 parts. The above classifier is trained using 2/3 parts of the training data with the features encoded in that chromosome and evaluated with the remaining 1/3 part.
4. Now, the overall recall, precision and F-measure values of this classifier for the 1/3 training data are calculated.
5. Steps 3 and 4 are repeated 3 times to perform 3-fold cross validation. The average F-measure value is used as the objective function. Thus, the objective function corresponding to a particular chromosome is $f_1 = \textit{F-measure}_{avg}$. The objective is to maximize this objective function using the search capability of DE.

3.4 Mutation

For each target vector $x_{i,G}$; $i = 1,2,3,\ldots,NP$, a mutant vector/donor vector is generated according to

$$v_{i,G+1} = x_{r1,G} + F(x_{r2,G} - x_{r3,G}), \tag{1}$$

where $r1$, $r2$, $r3$ are the random indices and belong to $\{1,2,\ldots,NP\}$. These are some integer values, mutually different and $F > 0$. The randomly chosen integers $r1$, $r2$ and $r3$ are also chosen to be different from the running index i, so that NP must be greater or equal to four to allow for this condition. F is a real and constant factor 0.5 $[0,1]$ which controls the amplification of the differential variation $(x_{r2,G} - x_{r3,G})$.

3.5 Crossover

In order to increase the diversity of the perturbed parameter vectors, crossover is introduced. This is well-known as the recombination. To this end, the trial vector:

$$u_{i,G+1} = (u_{1i,G+1}, u_{2i,G+1}, \ldots, u_{Di,G+1}) \tag{2}$$

is formed, where

$$u_{j,i,G+1} = v_{j,i,G+1} \quad \text{if } (randb(j) \leq CR) \text{ or } j = rnbr(i) \tag{3}$$

$$= x_{j,i,G} \quad \text{if } (randb(j) > CR) \text{ and } j \neq rnbr(i) \tag{4}$$

for $j = 1,2,\ldots,D$,

In Equation 3, $randb(j)$ is the jth evaluation of an uniform random number generator with outcome belongs to $[0,1]$. CR is the crossover constant belongs to $[0,1]$ which has to be determined by the user. Here the value of CR is 0.5. $rnbr(i)$ is a randomly chosen index x belongs to $\{1,2,\ldots,D\}$ which ensures that $u_{i,G+1}$ gets at least one parameter from $v_{i,G+1}$.

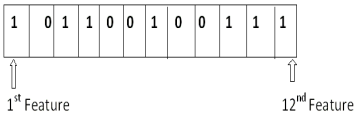


Figure 1: Chromosome representation for DE based feature selection

3.6 Selection

To decide whether or not it should become a member of generation G+1, the trial vector $u_{i,G+1}$ is compared to the target vector $x_{i,G}$ using the greedy criterion. If vector $u_{i,G+1}$ yields a smaller cost function value than $x_{i,G}$, then $x_{i,G+1}$ is set to $u_{i,G+1}$, otherwise, the old value $x_{i,G}$ is retained.

3.7 Termination Condition

In this approach, the processes of mutation, crossover (or, recombination), fitness computation and selection are executed for a maximum number of generations. The best string seen up to the last generation provides the best subset of features. Here the best string contains a set of features. This set is the optimal subset of features for NER problem for a specific language.

4 Proposed Method for Classifier Ensemble

The first step yields a set of solutions on the final best population. There is a single best solution, and along with that the population also contains many other solutions. Some of these solutions are good with respect to recall whereas some are good with respect to precision. All these solutions are equally important from the algorithmic point of view. We generate several different classifiers using these feature combinations. These are then combined using the proposed classifier ensemble creation technique in the second step. The proposed technique determines the best weights of NE classes for classifier combination.

4.1 Problem Formulation

Suppose, the N number of available classifiers be denoted by $C_1, \ldots, C_N$. Let, $\mathscr{A} = \{C_i : i = 1; N\}$ and there are M number of output classes. The proposed classifier ensemble is then stated as follows:

Find the weights of votes V per classifier which will optimize a function $F(V)$. Here, V is a real array of size $N \times M$. $V(i, j)$ denotes the weight of vote of the i^{th} classifier for the j^{th} class. More weight is assigned for that particular class for which the classifier is more confident; whereas the output class for which the classifier is less confident is given less weight. $V(i, j) \in [0, 1]$ denotes the degree of confidence of the i^{th} classifier for the j^{th} class. These weights are used while combining the outputs of classifiers using weighted voting. Here, F is a classification quality measure of the combined classifier. We choose F-measure as the objective function to optimize.

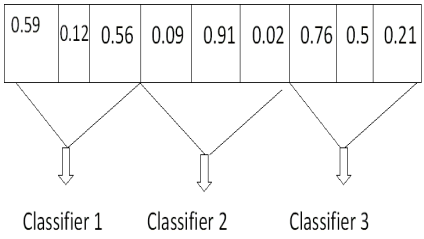


Figure 2: Chromosome Representation for Weighted Vote Based Classifier Ensemble Selection

4.1.1 Chromosome Representation and Population Initialization

If the total number of available classifiers is M and total number of output classes is O, then the length of the chromosome is $M \times O$. Each chromosome encodes the voting weights for possible O classes in each classifier. As an example, the encoding of a particular chromosome is represented in Figure 2. Here, $M = 3$ and $O = 3$ (i.e., total 9 votes can be possible). The chromosome represents the following ensemble:

The weights of votes for 3 different output classes for classifier 1 are 0.59, 0.12 and 0.56, respectively. Similarly, weights of votes for 3 different output classes are 0.09, 0.91 and 0.02, respectively for classifier 2 and 0.76, 0.5 and 0.21, respectively for classifier 3.

We use real encoding, and the entries of each chromosome are randomly initialized to a real value (r) between 0 and 1. Here, $r = \frac{rand()}{RAND\_MAX+1}$. If the population size is P then all the P number of chromosomes of this population are initialized in the above way.

4.1.2 Objective Functions Computation

Initially, the F-measure values of all the available classifiers for each of the output classes are calculated based on the development data. Thereafter, we execute the following steps to compute the objective values.

1. Suppose, there are total M number of classifiers. Let, the overall F-measure values of these M classifiers on the development data be F_i, $i = 1 \ldots M$.

2. We have M classes (each from a different classifier) for each token in the development data. Now for the ensemble classifier, the output class label for each token is determined using the weighted voting of these M classifiers' outputs. The weight of the output class provided by the i^{th} classifier is equal to F_i. The combined score of a particular class for a particular token t is:

$$f(c_i) = \sum F_m \times I(m,i),$$
$$\forall m = 1 \text{ to } M \text{ and } op(t,m) = c_i$$

   Here, $I(m,i)$ is the entry of the chromosome corresponding to the m^{th} classifier and i^{th} class; and $op(t,m)$ denotes the output class provided by the classifier m for the token t. The class receiving the maximum combined score is selected as the joint decision.

Examples: Let us consider the chromosome in Figure 2. Let the three classes be 'PER' (class 1), 'LOC' (class 2) and 'ORG' (class 3). Suppose the final F-measure values of 3 classifiers are 0.8, 0.7 and 0.85, respectively. Then let for a token *'kolkAtA* (Kolkata)' the 3 classifiers predict outputs as follows: Classifier 1: 'PER'; Classifier 2:'LOC'; Classifier 3:'LOC'. Then f('PER')=0.8*0.59=0.472; and f('LOC')=0.91*0.7+0.5*0.85=1.062. Henceforth, all the Bengali glosses are written in ITRANS notation [2]. Thus final class label selected for this particular token is 'LOC' as f('LOC')>f('PER').

3. We use the following objective function: F-measure$_{avg}$. This is maximized using the search capability of DE.

4.1.3 Operators

Other operators of this DE based ensemble approach are similar to those of the DE based feature selection technique described in the previous section.

5 Language Independent Features for NER

The main features for the NER task are identified based on the different possible combinations of available word and tag contexts. We use the following features for constructing the various models of CRF and SVM classifiers. The most important characteristics of our system is that it is both language as well as domain independent in nature. In order to maintain this property the features are *identified and generated without using any deep domain knowledge and/or language specific resources*. Due to this language independent behavior, the features can be easily obtained for almost all the languages.

1. **Context words**: These are the preceding and following tokens surrounding the current token. This is based on the observation that surrounding words carry effective information for the identification of NEs.

2. **Word suffix and prefix**: We use fixed length (say, *n*) word suffixes and prefixes as the features. Actually, these are the character strings stripped either from the rightmost (for suffix) or from the leftmost (for prefix) positions of the words. Morphological analyzers or stemmers could be more effective to extract the meaningful affixes of the wordforms. But there are no such freely available high quality tools for the Indian languages. We also wanted to build our system without using any language dependent resource or tool.

3. **First word**: This is a binary valued feature that checks whether the current token is the first word of the sentence or not. We consider this feature with the observation that the first word of the sentence is most likely a NE. This is the most useful feature for Bengali as NEs generally appear in the first position of the sentence in news-wire data.

4. **Length of the word**: This binary valued feature checks whether the number of characters in a token is less than a predetermined threshold value (here, set to 5). This feature is defined with the observation that very short words are most probably not the NEs.

5. **Infrequent word**: This is a binary valued feature that checks whether the current word appears in the training set very frequently or not. For each of the languages, we compile

[2] http://www.aczone.com/itrans/

the lists of most frequently occurring words for all the three languages. The threshold frequencies depend on the sizes of the datasets. In the present work, we consider the words to be infrequent if they have less than 10 occurrences in Bengali and Hindi datasets, and less than 5 occurrences in the Telugu dataset. A binary valued feature is then defined that fires if and only if the word does not appear in this list. We include this feature as the frequently occurring words are most likely not the NEs.

6. **Digit features**: Several digit features are defined depending upon the presence and/or the number of digits and/or symbols in a token. These features are digitComma (token contains digit and comma), digitPercentage (token contains digit and percentage), digitPeriod (token contains digit and period), digitSlash (token contains digit and slash), digitHyphen (token contains digit and hyphen) and digitFour (token consists of four digits only).

7. **Dynamic NE information**: This is the output label(s) of the previous token(s). The value of this feature is determined dynamically at run time. For CRF we use the bigram feature template that computes all the feature combinations of current and previous tokens.

8. **Content words in surrounding contexts**: At first we extract all unigrams in contexts $w_{i-3}^{i+3} = w_{i-3} \ldots w_{i+3}$ of w_i (crossing sentence boundaries) for the entire training data. Thereafter, we convert tokens to lower case, remove stopwords, numbers and punctuation symbols. We define a feature vector of length 10 using the 10 most frequent content words. Given a classification instance, the feature corresponding to token t is set to 1 iff the context w_{i-3}^{i+3} of w_i contains t. In order to compute this feature for the test set we first pass it through a NER system (Ekbal and Bandyopadhyay, 2009a) to extract the NE information of each token.

6 Datasets, Experimental Setup and Evaluation Results

In this section, we report the datasets used for the experiment, experimental setup and evaluation results with necessary discussions.

6.1 Datasets for NER

Indian languages are resource-constrained in nature. For NER, we use a Bengali news corpus (Ekbal and Bandyopadhyay, 2008), developed from the archive of a leading Bengali newspaper available in the web. One of the authors manually annotated a portion of this corpus containing approximately 250K wordforms with a coarse-grained NE tagset of four tags namely, PER (*Person name*), LOC (*Location name*), ORG (*Organization name*) and MISC (*Miscellaneous name*). The *Miscellaneous name* includes date, time, number, percentages, monetary expressions and measurement expressions. The data is collected mostly from the *National*, *States*, *Sports* domains and the various sub-domains of *District* of the particular newspaper. We also use the IJCNLP-08 NER on South and South East Asian Languages (NERSSEAL)[3] Shared Task data of around 100K wordforms that were originally annotated with a fine-grained tagset of twelve tags. This data is mostly from the agriculture and scientific domains. For Hindi and Telugu, we use approximately 502,913 and 64,026 tokens obtained from the NERSSEAL shared

[3] http://ltrc.iiit.ac.in/ner-ssea-08

Table 1: Named entity tagset for Indian languages (IJCNLP-08 NERSSEAL Shared Task Tagset)

NE Tag	Meaning	Example
NEP	Person name	*sachIna*/NEP, *sachIna ramesha tenDUlkara* / NEP
NEL	Location name	*kolkAtA*/NEL, *mahatmA gAndhi roDa* / NEL
NEO	Organization name	*yadabpUra bishVbidyAlYa*/NEO, *bhAbA eytOmika risArcha sentAra* / NEO
NED	Designation	*cheYArmAn*/NED, *sA.msada*/NED
NEA	Abbreviation	*bi e*/NEA, *ci em di a*/NEA, *bi je pi*/NEA, *Ai.bi.em*/ NEA
NEB	Brand	*fYAntA*/NEB
NETP	Title-person	*shrImAna*/NED, *shrI*/NED, *shrImati*/NED
NETO	Title-object	*AmericAn biUti*/NETO
NEN	Number	*10*/NEN, *dasha*/NEN
NEM	Measure	*tina dina*/NEM, *p.NAch keji*/NEM
NETE	Terms	*hidena markbha madela*/NETE, *kemikYAla riYYAkchYAna*/NETE
NETI	Time	*10 i mAgha 1402* / NETI, *10 ema*/NETI

Table 2: Statistics of the datasets

Language	# tokens in training	#NEs in training	#tokens in test	#NEs in test
Bengali	312,947	37,009	37,053	4,413
Hindi	444,231	43,021	58,682	3,005
Telugu	61,684	5,004	2,342	128

task. The underlying reason to adopt the finer NE tagset in the shared task is to use the NER system in various NLP applications, particularly in machine translation. The IJCNLP-08 NERSSEAL shared task tagset is shown in Table 1. One important aspect of the shared task was to identify and classify the maximal NEs as well as the nested NEs, i.e. the constituents of a larger NE. In the present work, we consider only the tags that denote person names (NEP), location names (NEL), organization names (NEO), number expressions (NEN), time expressions (NETI) and measurement expressions (NEM). The NEN, NETI and NEM tags are mapped to the *Miscellaneous name* tag that denotes miscellaneous entities. Other tags of the shared task are mapped to the 'other-than-NE' category denoted by 'O'. Statistics of the data sets in terms of the number of tokens in the training set, number of tokens in test set and number of NEs in training and test sets are given in Table 2.

6.2 Experimental Setup

The parameters of the proposed algorithm are selected by conducting a thorough sensitivity analysis on the development data. A part of the training dataset is used as the development set.

The parameters of DE are determined based on the development sets. The parameters of DE technique are as follows: population size = 100, CR (probability of crossover)=0.5, number of generations = 50 and F (mutation factor) = 0.5. We define the following *baseline* ensemble techniques:

- *Baseline 1*- This baseline is constructed by considering the following feature combination: Context of previous two and next two tokens along with all the features listed in Section 5.
- *Baseline 2*- This baseline is trained using the following feature combination: Context of previous two and next two tokens along with all the features listed in Section 5.
- *Baseline 3*- In this *baseline* model, all the individual classifiers selected from the first stage of the algorithm are combined together into a final system based on the majority voting of the output class labels. If all the outputs differ then anyone is selected randomly.
- *Baseline 4*- All the classifiers selected from the first stage of the algorithm are combined together with the help of a weighted voting approach. In each classifier, weight is calculated based on the F-measure value of the 3-fold cross validation on the training data. The final output label is selected based on the highest weighted vote.

In this work, we use CRF and SVM as the base classifiers. CRF (Lafferty et al., 2001b) considers a global exponential model. It has the freedom to include arbitrary features and the ability of feature induction to automatically construct the most useful feature combinations. For constructing CRF based classifiers, we use the C^{++} based CRF^{++} package [4], a simple, customizable, and open source implementation of CRF for segmenting or labeling sequential data. The SVM technique (Joachims, 1999; Vapnik, 1995) takes a strategy that maximizes the margin between the critical samples and the separating hyperplane. In particular, SVMs achieve high generalization even with training data of a very high dimension. Moreover, with the use of *kernel function*, SVMs can handle non-linear feature spaces, and carry out the training considering combinations of more than one feature. For constructing SVM based classifiers, we use YamCha[5] toolkit, an SVM based tool for detecting classes in documents and formulating the NER task as a sequential labeling problem. Here, we use both the *one-vs-rest* and *pairwise multi-class decision* methods, and the *polynomial kernel function* of degree 2.

At first DE based feature selection technique is used to determine the most relevant set of features for CRF and SVM. It produces a set of solutions that also includes the best solution. Based on the F-measure values we sort these solutions in the descending order. We select in total 14 solutions, 7 each for CRF and SVM. These effective classifiers are then used to construct the ensemble in the second stage. The performance of these (7+7=14) classifiers (with their feature combinations) are reported in Table 3 for Bengali. The CRF-based model exhibits the best performance (with respect to the overall F-measure value) yielding the recall, precision and F-measure values of 88.05%, 86.58% and 87.31%, respectively. Our first baseline which is constructed by including all the features in CRF model yields the recall, precision and F-measure values of 87.75%, 85.27% and 86.49%, respectively. Thus the DE based feature selection technique improves the F-measure value by 0.82 points. With SVM, the feature selection approach

[4] http://crfpp.sourceforge.net
[5] http://chasen-org/ taku/software/yamcha/

yields the recall, precision and F-measure values of 86.25%, 85.33% and 85.79%, respectively. This is an increment of 0.89 points over the second baseline where SVM is trained with all the available features. Overall evaluation results of our proposed two stage approach along with the best individual classifier and four different *baseline* ensembles are reported in Table 4. The proposed two-stage algorithm shows the recall, precision and F-measure values of 89.79%, 88.01% and 88.89%, respectively. This is an improvement of 1.58 points over the first stage, i.e. feature selection technique. It also demonstrates the overall performance increments of 1.62 and 0.78 percentage F-measure points over the third and fourth baselines, respectively.

Thereafter the proposed techniques are applied to Hindi and Telugu data sets. For both of these languages, we select 14 solutions, 7 each from CRF and SVM from the first stage, i.e. DE based feature selection approach. For Hindi, the CRF-based feature selection model exhibits the best performance with the recall, precision and F-measure values of 87.79%, 86.05% and 86.91%, respectively. Overall evaluation results of our proposed two stage approach along with the best individual classifier and four different *baseline* ensembles are reported in Table 4. The DE based feature selection approach shows an improvement of 0.59 points over the first baseline which is constructed by considering all the features into the model. The ensemble based approach attains the recall, precision and F-measure values of 88.88%, 87.35% and 88.09%, respectively. This is an improvement of 1.18 points over the feature selection approach. We clearly observe the increments of 1.28 and 1.25 points over the majority voted (i.e. Baseline-3) and weighted voted ensemble (i.e. Baseline-4) methods, respectively. For Telugu, the feature selection approach shows the recall, precision and F-measure values of 76.05%, 72.05% and 74.00%, respectively. These are the increments of 2.43 and 4.34 F-measure points over the first and second baseline, respectively. Overall evaluation results are reported in Table 4. Finally, the ensemble approach yields the overall recall, precision and F-measure values of 78.58%, 74.78% and 76.63%, respectively. This is superior to the feature selection approach as well as the other two ensemble methods. In order to show that the proposed two stage approach really outperforms the best individual classifier (i.e. feature selection approach) and four *baseline* ensembles, statistical analysis of variance (ANOVA) (Anderson and Scolve, 1978) is performed, when each is executed ten times. ANOVA tests show that for all the languages differences in mean recall, precision and F-measure are statistically significant as p value is less than 0.05 in each of the cases. We also compare the performance of our proposed approach with three state-of-the-art ensemble methods, namely stacking (Wolpert, 1992), AdaBoost (Freund and Schapire, 1995), ECOC (Error correcting Output Codes) (Dietterich and Bakiri, 1995) and two genetic algorithm (GA) based ensemble methods (Ekbal and Saha, 2010, 2011). In stacking, we used CRF as a meta-classifier. For ECOC, initially we generate binary classifiers for each NE class. Binary classifiers are generated using CRF and SVM. Thereafter, ECOC (Dietterich and Bakiri, 1995) method is applied to solve the multi-class problem. The code matrix is generated exhaustively and minimum Hamming distance based method is applied to determine the appropriate class label of each test instance. In case of AdaBoost we used CRF as a weak classifier. The GA based ensemble methods are executed on the classifiers obtained from the first stage of our proposed technique. We used the same parameter settings as used in (Ekbal and Saha, 2011) and (Ekbal and Saha, 2010). The techniques proposed in these papers were evaluated with set of features presented in this paper. Comparative evaluation results are shown in Table 5 for Bengali. In (Ekbal and Saha, 2010) authors have proposed a GA based simple classifier ensemble technique. The proper weights of votes were not determined like (Ekbal and Saha, 2011). Comparison shows that our proposed algorithm achieves superior performance compared to the previous two approaches. But approach proposed in (Ekbal and Saha, 2010) obtains better

Table 3: Evaluation results with various feature combinations for the CRF and SVM based classifiers for Bengali. Here, the following abbreviations are used: 'CW':Context words, 'PS': Size of the prefix, 'SS': Size of the suffix, 'WL': Word length, 'IW': Infrequent word, 'FW':First word, 'TD': 'Two-Digit', 'FD': 'Four-Digit', 'DH': 'Digit-Hyphen', 'DS': 'Digit-Slash', 'DD': 'Digit-Dot', 'DP': 'Digit-Percentage', 'DC': 'Digit-Comma', 'Sem': Content words, 'P', 'C' and 'N': Previous, current and next tokens, '$-i, j$': Words spanning from the i^{th} left position to the j^{th} right position, Current token is at 0^{th} position, 'X': Denotes the presence of the corresponding feature (we report in percentages), 'r': recall, 'p': precision, 'F': F-measure

Classifier	CW	TD	FD	DH	DS	DD	DP	DC	IW	WL	SS	PS	Sem	FW	p	r	F
CRF_1	-2,2	X	X	X	X	X	-	X	-	X	X	X	X	X	88.05	86.58	87.31
CRF_2	-3,3	X	X	X	X	X	X	X	-	X	X	X	X	X	87.98	86.53	87.25
CRF_3	-3,3	X	X	-	X	X	-	X	-	X	X	X	X	X	87.94	86.51	87.22
CRF_4	-2,2	X	-	X	X	X	X	X	-	X	X	X	X	X	87.98	86.57	87.26
CRF_5	-2,2	X	X	-	X	X	X	-	-	X	X	X	X	-	88.00	86.45	87.24
CRF_6	-2,2	X	-	X	X	X	X	X	-	X	X	X	X	X	87.88	86.56	87.21
CRF_7	-2,2	X	-	X	X	X	X	X	-	X	X	X	X	-	87.86	86.53	87.19
SVM_1	$-1,1$	X	-	X	-	X	-	X	X	-	X	X	X	X	86.25	85.33	85.79
SVM_2	$-2,2$	X	-	X	X	X	X	X	X	X	X	X	X	X	86.17	85.31	85.74
SVM_3	$-2,2$	X	X	X	-	X	X	-	-	X		X	X	X	85.61	84.18	84.89
SVM_4	$-2,2$	X	X	X	-	X	X	-	-	X		X	X	X	85.55	84.02	84.78
SVM_5	$-2,2$	-	-	X	X	X	X	X	-	X		X	X	X	85.10	84.00	84.54
SVM_6	$-2,2$	-	-	-	-	X	X	X	-	X		X	X	X	85.09	83.90	84.49
SVM_7	$-2,2$	X	X	-	X	X	X	X	X	X	X	X	X	-	85.19	83.79	84.49

Table 4: Overall results (we report in percentages) for Bengali, Hindi and Telugu; here 'r': recall, 'p': precision, 'F': F-measure, ME: Method, BIC: Best individual classifier, B1: Baseline-1, B2: Baseline-2, B3: Baseline-3, B4:Baseline- 4, TA: proposed two stage approach.

ME.	Bengali			Hindi			Telugu		
	p	r	F	p	r	F	p	r	F
BIC	88.05	86.58	87.31	87.79	86.05	86.91	76.05	72.05	74.00
B1	87.75	85.27	86.49	87.21	85.45	86.32	72.29	70.87	71.57
B2	86.01	83.82	84.90	87.34	83.79	84.52	70.52	68.82	69.66
B3	88.03	86.53	87.27	87.66	85.99	86.81	74.78	72.19	73.47
B4	88.69	87.54	88.11	87.58	86.12	86.84	75.57	72.05	73.77
TA	89.79	88.01	88.89	88.88	87.35	88.09	78.58	74.78	76.63

F-measure values for some data sets than the proposed method. This is due to the use of many language specific features extracted from Part-of-Speech (PoS) tagger and several gazetteers. Experimental analysis suggests that errors are mostly due to boundary detection and the conflicts between the organization and location classes.

Results show that our proposed two-stage DE based technique performs better than the best individual classifier, four different baseline ensemble techniques and some other existing state-of-the-art ensemble methods (c.f. Table 4 and Table 5). It is already established in machine learning literature that proper ensemble of classifiers should always perform better in comparison to the existing base classifiers. The first stage of our algorithm that concerns with the relevant feature selection performs better than the two baseline models, where classifiers are trained with all the available features. This clearly shows the necessity of determining appropriate feature set for the problem. Our proposed DE based ensemble shows significant performance gains over all the baselines. In the third and fourth baselines, we combined the

Table 5: Comparative Evaluation Results (we report in percentages)

Classification Scheme	recall	precision	F-measure
Stacking	87.88	86.24	87.05
ECOC	87.91	86.23	87.06
AdaBoost	88.53	86.84	87.68
GA based ensemble (Ekbal and Saha, 2010)	88.01	85.89	86.93
GA based ensemble (Ekbal and Saha, 2011)	88.72	87.06	87.88
DE based Approach	89.79	88.01	88.89

classifiers blindly. As a result, in some cases, the combined model even shows inferior performance compared to the best individual classifier(s) that correspond to our feature selection method. The effectiveness of DE in determining proper weights of voting is also the another reason of showing better performance. The nature of performance supports our underlying hypothesis that determining appropriate weights of voting for each class in each classifier is very crucial. Results show that the proposed algorithm performs best for Bengali followed by Hindi and Telugu. The possible reasons could be (i). the ratios of positive (NEs) and negative examples (non-NEs) in the respective training data, i.e. 1:9 (Bengali), 1:9.33 (Hindi). and 1:13 (Telugu), (ii). agglutinative nature of Telugu that may possibly require some language specific rules and (iii). less amount of training data for Telugu compared to others. This may be due to the fact that all the datasets are highly imbalanced, and hence greatly biased towards the negative categories. Thus, there are not enough NE instances that could be more effective for NE identification. This observation leads to the path of investigating appropriate sampling strategy in order to make the ratios of positive and negative examples more balanced.

Conclusion

In this paper we have proposed a two stage differential evolution based technique for NER in three different languages. Here at first DE based technique is used to select relevant sets of features for different classifiers. We used CRF and SVM as the underlying classification methods. Thereafter effective classifiers were selected based on the F-measure scores, and combined using a DE based classifier ensemble technique. Evaluation results show the encouraging performance in all the settings. Comparisons with the conventional baselines and some state-of-the-art ensemble methods show the superiority of our proposed technique. The key contributions of this work are are two-fold, *viz.* (i). proposal of differential evolution (DE) based feature selection and classifier ensemble methods that can be applied to any classification problem; and (ii). scope of the development of language independent NER systems in a resource-poor scenario. In future we will study the effects of various language dependent features that can be extracted from morphological analyzers and gazetteers.

Overall evaluation results suggest that there is still the room for further improvement. In this work, we have considered the problem of feature selection and classifier ensemble as a single objective optimization problem. In future, we will develop some multi-objective DE based techniques to solve these feature selection and classifier ensemble methods.

References

Anderson, T. W. and Scolve, S. (1978). *Introduction to the Statistical Analysis of Data*. Houghton Mifflin.

Bikel, D. M., Schwartz, R. L., and Weischedel, R. M. (1999). An Algorithm that Learns What's in a Name. *Machine Learning*, 34(1-3):211–231.

Borthwick, A. (1999). *Maximum Entropy Approach to Named Entity Recognition*. PhD thesis, New York University.

Chinchor, N. (1995). MUC-6 Named Entity Task Definition (Version 2.1). In *MUC-6*. Maryland.

Chinchor, N. (1998). MUC-7 Named Entity Task Definition (Version 3.5). In *MUC-7*. Fairfax.

Cunningham, H. (2002). GATE, a General Architecture for Text Engineering. *Computers and the Humanities*, 36:223–254.

Dietterich, T. G. and Bakiri, G. (1995). Solving multiclass learning problems via error correcting output codes. *Journal of Artificial Intelligence Research*, 2:263–286.

Ekbal, A. and Bandyopadhyay, S. (2007). Lexical Pattern Learning from Corpus Data for Named Entity Recognition. In *Proceedings of the 5th International Conference on Natural Language Processing (ICON)*, pages 123–128, India.

Ekbal, A. and Bandyopadhyay, S. (2008). A Web-based Bengali News Corpus for Named Entity Recognition. *Language Resources and Evaluation Journal*, 42(2):173–182.

Ekbal, A. and Bandyopadhyay, S. (2009a). A Conditional Random Field Approach for Named Entity Recognition in Bengali and Hindi. *Linguistic Issues in Language Technology (LiLT)*, 2(1):1–44.

Ekbal, A. and Bandyopadhyay, S. (2009b). Voted NER System using Appropriate Unlabeled Data. *Proceedings of the 2009 Named Entities Workshop: Shared Task on Transliteration (NEWS 2009), ACL-IJCNLP 2009*, pages 202–210.

Ekbal, A., Naskar, S., and Bandyopadhyay, S. (2007). Named Entity Recognition and Transliteration in Bengali. *Named Entities: Recognition, Classification and Use, Special Issue of Lingvisticae Investigationes Journal*, 30(1):95–114.

Ekbal, A., R.Haque, and Bandyopadhyay, S. (2008). Named Entity Recognition in Bengali: A Conditional Random Field Approach . In *Proceedings of the 3rd International Joint Conference on Natural Language Processing (IJCNLP 2008)*, pages 589–594.

Ekbal, A. and Saha, S. (2010). Classifier ensemble selection using genetic algorithm for named entity recognition. *Research on Language and Computation*, 8:73–99.

Ekbal, A. and Saha, S. (2011). Weighted vote-based classifier ensemble for named entity recognition: A genetic algorithm-based approach. *ACM Trans. Asian Lang. Inf. Process.*, 10(2).

Freund, Y. and Schapire, R. (1995). A decision-theoretic generalization of on-line learning and an application to boosting. In *Proceedings of the Second European Conference on Computational Learning Theory*, pages 23–37, Taipei, Taiwan.

Joachims, T. (1999). *Making Large Scale SVM Learning Practical*, pages 169–184. MIT Press, Cambridge, MA, USA.

Lafferty, J. D., McCallum, A., and Pereira, F. C. N. (2001a). Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data. In *ICML*, pages 282–289.

Lafferty, J. D., McCallum, A., and Pereira, F. C. N. (2001b). Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth International Conference on Machine Learning*, ICML '01, pages 282–289, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Li, W. and McCallum, A. (2004). Rapid Development of Hindi Named Entity Recognition using Conditional Random Fields and Feature Induction. *ACM Transactions on Asian Languages Information Processing*, 2(3):290–294.

Patel, A., Ramakrishnan, G., and Bhattacharya, P. (2009). Relational Learning Assisted Construction of Rule Base for Indian Language NER. In *Proceedings of ICON 2009: 7th International Conference on Natural Language Processing*, India.

Sekine, S. (1998). Description of the Japanese NE System used for MET-2. In *MUC-7*, Fairfax, Virginia.

Shishtla, P. M., Pingali, P., and Varma, V. (2008). A Character n-gram Based Approach for Improved Recall in Indian Language NER. In *Proceedings of the IJCNLP-08 Workshop on NER for South and South East Asian Languages*, pages 101–108.

Srikanth, P. and Murthy, K. N. (2008). Named Entity Recognition for Telugu. In *Proceedings of the IJCNLP-08 Workshop on NER for South and South East Asian Languages*, pages 41–50.

Storn, R. and Price, K. (1997). Differential evolution - a simple and efficient heuristic for global optimization over continuous spaces. *J. of Global Optimization*, 11(4):341–359.

Vapnik, V. N. (1995). *The nature of statistical learning theory*. Springer-Verlag New York, Inc., New York, NY, USA.

Vijayakrishna, R. and Sobha, L. (2008). Domain Focused Named Entity Recognizer for Tamil using Conditional Random Fields. In *Proceedings of the IJCNLP-08 Workshop on NER for South and South East Asian Languages*, pages 93–100.

Wolpert, D. (1992). Stacked generalization. *Neural Networks*, 5:241–259.

Yamada, H., Kudo, T., and Matsumoto, Y. (2001). Japanese Named Entity Extraction using Support Vector Machine. *In Transactions of IPSJ*, 43(1):44–53.

Noun Group and Verb Group Identification for Hindi

Smriti Singh[1], Om P. Damani[2], Vaijayanthi M. Sarma[2]
(1) Insideview Technologies (India) Pvt. Ltd., Hyderabad
(2) Indian Institute of Technology Bombay, Mumbai, India
smriti.singh@insideview.com, damani@cse.iitb.ac.in,
vsarma@iitb.ac.in

ABSTRACT

We present algorithms for identifying Hindi Noun Groups and Verb Groups in a given text by using morphotactical constraints and sequencing that apply to the constituents of these groups. We provide a detailed repertoire of the grammatical categories and their markers and an account of their arrangement. The main motivation behind this work on word group identification is to improve the Hindi POS Tagger's performance by including strictly contextual rules. Our experiments show that the introduction of group identification rules results in improved accuracy of the tagger and in the resolution of several POS ambiguities. The analysis and implementation methods discussed here can be applied straightforwardly to other Indian languages. The linguistic features exploited here are drawn from a range of well-understood grammatical features and are not peculiar to Hindi alone.

KEYWORDS : POS tagging, chunking, noun group, verb group.

Proceedings of COLING 2012: Technical Papers, pages 2491–2506,
COLING 2012, Mumbai, December 2012.

1 Introduction

Chunking (local word grouping) is often employed to reduce the computational effort at the level of parsing by assigning partial structure to a sentence. A typical chunk, as defined by Abney (1994:257) consists of a single content word surrounded by a constellation of function words, matching a fixed template. Chunks, in computational terms are considered the truncated versions of typical phrase-structure grammar phrases that do not include arguments or adjuncts (Grover and Tobin 2006). For Abney, chunks are connected subgraphs of a sentence's parse tree. They are defined in terms of major heads and have their own syntactic structure that can be represented in the form of a tree. However a chunk does not include all the descendants of the root node that may be present in the parse-tree of the complete sentence. It only represents the root node (the head of the chunk) and its modifiers (auxiliaries in the case of verbs). Two heads of the same lexical category are not allowed inside a chunk. Consequently, 'Ram's son' in English and 'rām kā betā' in Hindi will have two chunks each [Ram's] [son] and [raam kā] [betā]. Similarly, verb complements are not grouped inside the verb chunk; they form separate chunks. The English sentence 'The bald man was sitting on his suitcase,' can be grouped into three chunks – [The bald man], [was sitting] and [on his suitcase]. The parallel sentence in Hindi 'ganjā ādmī apne sandūk pe baithā thā' will have the chunks [ganjā ādmī] [apne sandūk pe] [baithā thā] in that order.

The task of the chunker is to divide a sentence into chunks leaving out some words that are not grouped into any of the identified chunks. The output of the chunker is a shallow syntactic analysis employing simple, context sensitive grammars to detect the boundaries of syntactic groups such as a Noun Group (NG) or a Verb Group (VG). It identifies major constituents of a sentence without further identifying a hierarchical structure that connects and arranges the chunks (Abney 1991, Ramshaw and Mitchell 1995). The chunked structures (or groups, as we shall refer to them from here on) do not correspond straightforwardly to any structure in a typical phrase-structure analysis. A chunker makes use of the POS information provided by a tagger to form groups. A Noun Group consists of a head noun along with its qualifiers and modifiers (including particles). A Verb Group contains a single main verb and any auxiliaries, negation markers, and focus particles. The grammatical information of a word group depends on the order of its constituent morphemes and the information associated with those constituents. The group identification module makes use of the morphological information and the POS information provided by a morphological analyser and a POS tagger respectively. The group identification module can also be employed before POS Tagging in which case it works with possible POS tags of a given word given in a lexicon.

The focus of this work is Hindi word group identification. We performed a detailed corpus analysis and came up with word grouping rules. Local word grouping in Hindi was first discussed by Bharati et al. (1995). They built a Paninian Parser (using karaka or semantic case relations) that internally uses a morphological analyser as well as a Local word grouper (LWG). Following Bharati et al., Ray et al. (2003) attempted local word grouping using a list of regular expressions to form groups. From the list of ten possible modifier-modified structures discussed by Bharati et al., Ray et al. worked on the five structures that rely only on local modifier-modified relationships and do not need long distance dependencies. Hindi word grouping has also been attempted using statistical models including those by Baskaran (2006) using an HMM based approach and Singh A. et al. (2005) and Dalal et al. (2006) using Maximum Entropy Models. These systems rely very little on linguistic knowledge and instead use a large corpus for automatic learning. For Marathi, a close cousin of Hindi, verb group identification was deployed in a CRF based POS tagging system in Gune H. et. al. (2010). Limited noun phrase chunking has also been done for Turkish (Kutlu M. 2010) and Tamil(Vijay and Sobha 2010). While the

focus of our work is on Hindi, the analysis and implementation methods discussed here can be applied straightforwardly to other Indian languages. The linguistic features exploited here are drawn from a range of well-understood grammatical features and are not peculiar to Hindi alone.

2 Need for Group Identification in a POS Tagging System

On analysing the output a CRF (Conditional Random Fields) based POS Tagger, we discovered that most of the systems errors were due to its inability to disambiguate POS tags in the absence of large training corpora. The system needed a detailed group level analysis to resolve the ambiguities between adjective and noun, main verb and auxiliary verb or demonstrative and pronoun. In other words, a granular group level analysis was needed that made use of the morphotactical arrangement both within a word form and in between words. To motivate this further, we provide a detailed error analysis in the following.

a) Demonstrative-Personal Pronoun POS ambiguity: Data-driven learning may not help much in resolving the ambiguity because of a number of qualifiers that may appear between a demonstrative and the head noun. In most cases, a word with the demonstrative-personal pronoun ambiguity is assigned the tag PRON (pronoun) if it is not immediately followed by a noun. Hence, the tagger incorrectly tags some demonstratives (DEM) as pronouns as shown in 1a. In 1b, in contrast, *'us'* appears as a PRON and not as DEM.

1) a) *us kāl*-e ghod-e ko rok-o
that-obl black-obl horse-obl ACC stop-imp
'Stop that black horse'

b) us ko roko
He-obl ACC stop
'Stop him'

The first word in 1a should be tagged as Dem while the tagger incorrectly tags it as Pronoun because of a lack of representative training data. Similarly, sentences 2a and 2b are ambiguous for the system. In1a-b and 2a-b, *'us'* and '*ve'* are valid candidates for both DEM and PRON tags. The ambiguity arises for the system because it seeks to resolve it by looking at the words in the immediate vicinity. Note that these sentences are not ambiguous for a native speaker.

2) a) *vo kāl*-e ghod-e ko rok rəh-*ā* hai
he black-obl horse-obl ACC stop prog-masc,sg be-pres
'He is stopping the black horse'

b) vo *kālā ghodā* so rəh-*ā* hai
that black horse sleep prog-masc,sg be-pres
'That black horse is sleeping'

The TAM (tense, aspect and modality) information of the verbs in the two sentences can also help in resolving the ambiguity but requires subject-object information in the sentence along with a syntactic analysis of the sentence.

b) Adjective-Noun ambiguity: An adjective may function as a head noun if the noun is dropped, and bears the same inflection as the nominal head, as may be seen in 3 and 4.

3) *ə*cch-e *kām kā nətijā ə*cch*ā nikə*l-t-*ā hai*
good-obl deed of result good turn-hab,masc,sg be-pres
'Do good have good'

4) *ə*cch-e *kā nətijā ə*cch*ā nikə*l-t-*ā hai*
good-obl of result good turn-hab,masc,sg be-pres
'Do good have good'

If the case marker orpostposition immediately follows the adjective, it is treated as a nominal head. Since the occurrence of *ə*cche (or any other adjective) as an adjective is more likely than its occurrence as a noun in any learning corpus, this will result in incorrect learning and consequently, in incorrect tagging. NG identification rules help in resolving such ambiguity by using the featural information of the NG constituents.

d) Noun-Verb ambiguity: Many nouns may appear as verbs (even when inflected) and vice versa in Hindi[1]. Verbs may appear as verbal nouns in their infinitival form and may function as nouns. Nevertheless, a verbal noun retains many of its verbal properties. While functioning as a noun, it appears only in the 'singular, oblique' and the 'singular, direct' cases and inflects like other */ā/* ending masculine nouns in the language.

5) tair-*nā bəhut lābhkārī hai*
swim-Inf very beneficial be-pres
'Swimming is very beneficial'

6) tair-n-e ke b*əhut lābh* haĩ
swim-Inf-obl Poss many benefits be-pres,pl
'Swimming has many benefits'

Infinitival verbs as either main verbs or verbal nouns have identical forms. The POS ambiguity is easy to resolve when the verbal noun is in the oblique and is followed by a postposition as shown in 6 above. More difficult are sentences where it appears in the direct form. In 7 the verb j*ānā* appears inside a VG and should be tagged as an infinitival verb. While in 8, it appears as a verbal noun and is also modified by a possessive pronoun *merā.* These two occurrences of *jānā* as a noun and an infinitival verb yields POS ambiguity (N or V). However, when an infinitival verb is immediately preceded by a possessive pronoun or a genitive postposition, as in 8, it should be tagged as a verbal noun, and the NG information can be successfully exploited by the POS tagger.

7) mujh-*e [jā-nā hai]*
I-DAT go-Inf be-pres
'I have/want to go'

8) *[merā jā-nā] zərūrī hai*
My go-Inf important be-pres
'For me to go is important'

3 Noun Groups in Hindi

Nominal groups are defined by Halliday (1977:7) as *"...nouns plus their determi*ners and any *other modifiers...."*. Specifiers and modifiers/qualifiers are optional while the headword (noun) constitutes the obligatory element in the structure of an NG. Specifiers can be determiners, ordinals, and cardinals. Qualifiers include adjectives, prepositional, or postpositional groups or a relative clause. The idea of an NG is not far removed from that of a syntactic NP but it does not straightforwardly match the constituents of syntax. The constituents of a group always appear in a particular default order and this is subject to cross-linguistic variation. Hindi being a head-final language, the head of an NG is the rightmost constituent in the group. The head may be preceded

[1]For example, in the sentence 'mere*kə*īkh*ātehaĩ*'' the token'kh*āte'* isambiguous for NOUN (pl, direct) and VERB (habitual, pl, masc/fem).

by the words that belong to pre-nominal categories. NGs are formed around a noun or a pronoun that acts as a nucleus in the group. Types of Hindi NGs include:

- N, e.g., mez (table)
- Dem Pron+N, e.g., vo mez (that table)
- Poss pronoun+N, e.g., *merā kəmrā* (my room)
- Adj+N, e.g., sundər lə*ɽ*kī (beautiful girl)
- Dem pron+Adj+N, e.g., vo sundər lə*ɽ*kī (that beautiful girl)
- Card+N, e.g., *cār* gho*ɽ*e (four horses)
- Ord+N, e.g., *dūsrā lə*ɽ*kā* (second boy)
- Non-Spec Det+N, e.g., *kuch kitābẽ* (some books)
- Det+Adj+N, e.g., *kuch purānī kitābẽ* (some old books)
- Inten+Adj+N, e.g., b*əhut purānī kitābẽ* (very old books)
- Pron, e.g., ve (they), v*əh (he/it/she), tum (you), āp* (you-honorific)
- N or Proper N (postpositions fuse with Hindi Pronouns; they are not written as free words) followed by a simple postposition or a compound postposition, e.g. lə*ɽ*ke **ke lie** (for the boy), kəmre **m*ẽ*** (in the room), mez **p*ə*r** (on the table)
- Part/Discourse marker+N, e.g., lə*ɽ*kī **h**ī (girl only), lə*ɽ*kī *bhī (girl too), pānī tə*k (water even)

The ordering of the constituent elements of a Hindi NG can be captured using morphotactical rules and a few additional constraints. For example, the end of a Noun Group may easily be marked when the group is Oblique, i.e. when a post-position appears immediately after the head noun. NGs where the head is not directly followed by a postposition require deeper analysis to mark the group boundary. In addition to consulting standard Hindi grammar texts like Kachru(2006), we performed a detailed corpus analysis to determine the word grouping rules. Candidate constituents of the Hindi NG may be placed in five sets as shown below. Optional elements are marked by parentheses. If two or more elements always appear together, they are shown as a single unit within parentheses. Curly brackets are used to show optionality between constituents competing for a single position.

Set 1 includes possessive demonstrative pronouns. Both are optional elements of a Hindi NG and may appear in any order with or without the other. For example, both *vo tumhārī mīthī bātẽ* (those your sweet word*s), tumhārī vo mīthī bātẽ* (your those sweet words) are possible. The possessive followed by a demonstrative pronoun is the canonical order, while the reversed order is a stylistic, poetic construction. The optionality and the order of Set 1 elements is shown in 9.

9) ((Demonstrative) (Possessive)) OR ((Possessive) (Demonstrative))

Thus, any of the following outputs are valid:

- Both items are optional – (*vo tumhārī) mīthī bātẽ*
- Both items may appear together - *vo tumhārī mīthī bātẽ* or *tumhārī vo mīthī bātẽ*
- One item appears without the other - *vo mīthī bātẽ* or *tumhārī mīthī bātẽ*

Set 2 includes intensifiers and numerals. A numeral may be of the type - approximate, fractional, universal quantifier, indefinite quantifier, multiplicative, aggregative, ordinal, cardinal and measure word. Kachru (2006:133) provides the ordering among Hindi quantifiers as in 10.

10) approximate-cardinal-collective-ordinal-multiplicative/fractional-measure

We modified this ordering to capture the arrangement of numerals in a more elaborated way (in Figure 1 below).

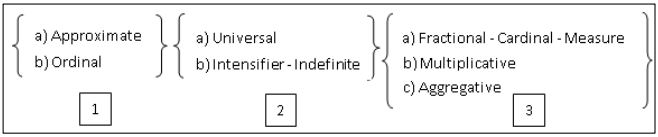


Figure 1: Ordering of quantifiers in a Hindi NG

The categories within curly braces are mutually exclusive while those separated by a hyphen '–' can appear one after another in a sequence. The ordering suggests that:

- Approximate quantifier and ordinal (e.g., *ləgbhəg *dūsrā vyə*kti *'around second man'*), universal and indefinite quantifier (e.g., *s*əbhī kuch log 'all few people'*), cardinal and aggregative (e.g., *do donõ log), aggregative and multiplicative (e.g., **donõ dugunā*), and fractional and aggregative/multiplicative quantifier (e.g., **ādhā donõ, *ādhā dugunā*) are mutually exclusive
- An intensifier may precede an indefinite quantifier but not a universal quantifier (e.g., b*ə*hut k*əm log 'very few people' (intensifier*-indefinite quantifier), b*ə*hut s*əbhī log 'very all people'* (intensifier-universal quantifier))
- Fractionals do not appear with aggregative or multiplicative quantifier (e.g., **ādhā donõ, *ādhā dugunā*)

Set 3 includes adjectives (including imperfective or perfective verbal adjectives) and are optional in an NG. Many adjectives may appear inside an NG recursively. Examples include, *bhāgtā huā kālā* gho*ɽā (running blackhorse), bhāgtā huā ghoɽā (running horse), kālā ghoɽā (black horse),* th*əkā huā kālā ghoɽā (tired black horse), thəkā huā ghoɽā (tired horse*).

11) ((Verbal Adjective) (Adjective))

The adjectives are internally ordered based on the adjective type. Those that denote shape, color, size or the origin of a noun are known as fact adjectives. Those that refer to a noun's quality or those that denote a speaker's opinion appear before fact adjectives. The order followed by different kinds of adjectives is quality-size-age-shape-color-origin material, as in l*əmbī kālī reshmī bənārəsī sā*dī *(long black silk* banarasi saree), n*əyā khushhāl bhārtiyə səmudāyə* (new happy Indian community), etc.

Set 4 includes nouns and pronouns (except the demonstrative) which are obligatory members (Heads) of an NG. They are the right most element of the group with the exception of particles and postpositions that appear after them.

Set 5 includes postpositions that form oblique NGs. Postpositions may be primary (such as ne, ko, ke, etc.) or compound (such as *'ke bā*d*', 'ke sā*t', etc.) and are optional.

Particles (focus, emphatic, etc.) may appear at many places (even at the end) within an NG and we have not put them in any set. The particles to and *bhī* may appear only at the end of an NG.

The complete ordering of constituents within a Hindi NG is given in the expression in 12 where () show optionality, [] represents a set and * stands for zero or more repetition.

12) NG = (Set 1) (Set 2) (Set 3)* Set 4 (Set 5) (Particle)

3.1 Procedure for Noun Group Identification

The Noun Group identification module attempts to isolate the basic non-recursive NG that includes only one head and its specifiers and modifiers. The input to the algorithm is the output of a morphological analyser. For each word, the morphological analyser gives the stem,andthe set of suffixes along with the associated morphological properties. A look-up is performed in a lexicon to retrieve the set of possible POS tags for each stem. The NG is built from right to left in a given sentence. As discussed in the previous section, we formulated five sets of constituents that contain different lexical categories that combine in various ways to form NGs in Hindi. Set 4, the head marks the right end of an NG (neglecting any postpositions and particles).

Sets 1, 2 and 3 contain categories which mark the left end of an NG. Processing from right to left, once the system encounters a Set 4 element, it starts to look for Set 3, Set 2 and Set 1 elements appearing to the left of the head in that order. By 'finding a Set X element', we mean 'finding a stem whose potential POS tag list in the lexicon contains a POS tag belonging to Set X.' The potential candidates are considered to be members of the NG. As soon as any word of a lexical category other than those mentioned in Sets 1, 2 and 3 is encountered, the NG is considered closed. The previous word marks the left end of the NG in such a case. The number, gender and case information for nouns, demonstratives and pronouns are required at each step to select or reject a potential POS tag. This information is extracted from the output of the morphological analyser. The pseudo code for NG identification is given below.

Steps for NG Identification

1. For all tokens, processing goes from right to left
    - 1a. Look for a post-position or a Set 4 element to start an NG
    - 1b. If Set 5 member, i.e., a postposition is found
        - 1b (i) Oblique NG has started
    - 1c. If Set 4 element is found
        - 1c (i) Direct NG has started
    - 1d. If a Demonstrative pronoun is found
        - 1d (i) Consider it as a Pronoun (head)
2. If oblique NG has just started with a Set 5 element, i.e., with a postposition
    - 2a. Look for a Set 4 element
    - 2b. If Set 4 element is not found; find the list of possible POS tags for the current word
    - 2c. If *a POS Tag appears in the possible POS Tags' list and also in Set 4*
        - 2c (i) Assign the tag which is common to both.
    - 2d. If there is no common element in the list and Set 4s
        - 2d (i) Assign the tag other than PP to the next word using the list of possible tags for it.
3. If any NG has started
    - 3a. Look for a Set 3 and/or Set 2 and/or Set 1 element
    - 3b. If Set 3, 2 and 1 elements are found
        - 3b (i) The NG includes the current word
    - 3c. If set 3, 2 and/or 1 elements are not found
        - 3c (i) The NG has already ended with the previous word
4. If any NG is completely identified
    - 4a. Apply rules to check the agreement between modifiers/qualifiers and their head and do corrections if necessary
5. Start looking for the next NG

In what follows, we give an example of how the NGI helps correct a POS Tag error.

13) *ve pūr-e māml*-e ko *suljhā-nā cāh-te haĩ*
They whole-obl matter-obl ACC solve-Inf want be-pres-pl
'they want to solve the whole matter'

For 13 the tagger produces the output as [DEM ADJ NN PP VM VAUX]. *'ve'* is tagged as a DEM instead of PRON. Scanning right to left, the NG identified is (*ve pūre māmle ko*). Now the computational rules are applied to make any POS corrections required. By the first rule for oblique NG,reading the rule from right to left, we find a PP '*ko'* followed by a noun '*māmle'* in the oblique case. '*pūre'* is allowed in the NG as its category and features warrant its being a Set 2 member. '*ve'*may be a Set 1 member and may mark the left end of the NG and may be a demonstrative or a pronoun. The tagger may tag '*ve'*as DEM 'demonstrative' but, as a demonstrative, it does not concord with the head noun for the relevant case feature. Thus, the tag DEM is rejected and PRON is selected.

4 Verb Groups (VGs) in Hindi

A Hindi VG includes a single main verb root followed by a sequence of inflectional suffixes and/or auxiliary verb sequences. The group contains various verbal morphemes that centre on a single event. The verbal morphemes occur in a fixed order and are subject to several grammatical and semantic constraints. Some examples of Hindi VGs are *[khā-yā]* (eat-past) and *[khā-yā gə-yā* hai] (eat-perf passive-perf,sg be-pres).While analysing Hindi VGs, we have not considered complex predicates such as conjunct verbs (as *ārəmbhkər 'start' (literally 'start do')*) or compound verbs (*'kəṛdāl' ('somehow finish')* (Chakrabarti et al.2007, Chakrabarti et al.2008, Begum et al. 2011). Here, a main verb is a single verb root that appears with associated inflectional morphemes.

4.1 Identifying Verb Group Boundaries

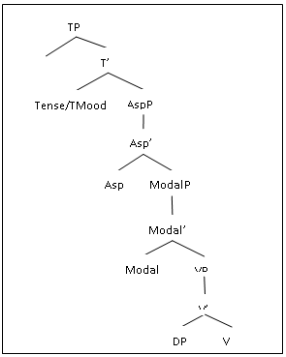


Figure 2: Order of Verbal Elements in a Hindi Verb Group

A VG boundary is marked using the order in which Hindi verbal elementsarrange themselves . The linear order of the major grammatical categories within a Hindi VG is Verb-Aspect-Tense/Mood as shown in Figure 2. The grammatical properties for which Hindi verbs inflect are tense, aspect, mood, modality, gender, number, person, honoroficity, voice and finiteness. These properties are realised analytically or periphrastically (either as suffixes or as auxiliaries). A

Hindi verb group must always begin with a main verb root with or without a suffix. Once the main verb is identified, the verb group is assumed to have begun. Scanning from left to right, the main verb may be followed by a string of intermediate verbal suffixes and auxiliaries until a must-end VG marker is encountered. These elements broadly follow the linear order in 14, though with co-occurrence constraints that are listed towards the end of this section.

14) *Verb Root*−Infinitive/*Passive*−*Modal Auxiliary*−*Aspect*−*Tense*−Mood

The three kinds of morphemes are called Start markers, Intermediate markers and Must-end Markers and are shown in Figure 3. Particles and negation markers are also allowed to appear inside a VG.

Start Marker	Intermediate Markers		Must End Markers
	Possible End Markers	Must-Continue Markers	
Main Verb (Root)	Necessity	Ability/Probability,	Present Tense
	Perfective-gen-num	Obligation/Permission	Past Tense
		Habitual/Progressive	Future+gen-num

Figure 3: Boundary Markers for Hindi VG

a) Start markers

A Hindi VG 'start' marker is always a verb root whether inflected or uninflected. All verbal auxiliaries may also be considered as start markers. Since the identification begins from left to right, the first instance of a free verbal morpheme is always the root or main verb. In rare cases (poetic constructions), verbal auxiliaries appear before the main verb in a Hindi VG, as for examplein 15, where the tense auxiliary precedes the main verb and starts a VG. This scrambling is usually seen only with tense auxiliaries. Such reordering with aspectual auxiliaries is even rarer (see 16). We exclude here the identification of these rare VGs. If such constructions are encountered, the system will identify them as two separate VGs, albeit incorrectly.

15) vo roz mujh-se [hai mil-*tā]*
he everyday I-DAT be-pres meet-hab
'He meets me everyday'

16) vo mujh-se [rəh-*ā hai mil]*
He I-DAT prog-masc,sg be-pres meet
'he is meeting me'

b) Intermediate markers

These markers include two kinds of morphemes, 1) possible-end markers and 2) 'must continue' markers. Possible end markers are those which may end a VG such as the perfective marker or the modal auxiliary for necessity (preceded by an infinitive-gender, number sequence). These morphemes, however, may be followed by other morphemes to further extend the VG. For example, the perfective marker may be followed by the past or the present tense auxiliary as in vo *āyā* 'he came', *vo āyā* hai 'he has come' and vo *āyā thā* 'he had come'. Similarly, the modal auxiliary for necessity may be followed by the past tense auxiliary, such as *usko ānā cāhiye (thā)* 'he should (have) come' and the subjunctive marker may be followed a future-person, number marker, such as *khā-ũ*-g-*ā 'eat*-subjunctive-will-*person, number'.* The 'must-continue' markers,

however, must be followed by other verbal morphemes in order to complete the VG. Details of such markers are given below in Table 1 along with their inflections.

Possible End-Markers	
Modal Auxiliary	*चाहिए (cāhie)* 'should'
Aspect: Perf+gen-num	*-या (-yā), -ा(-ā), -आ(-ā), -ी(-ī), -े(-e), -ए(-e), -ई(-ī), -ीं(-ĩ), -ईं(-ĩ)*
Subjunctive	*-ूँ(-ũ), -ऊँ(-ũ), -ें(-e), -ऍ(-e), -ं(-ŋ), -ेंं(-ẽ), -ऍं(-ẽ), -ों(-o) ,-ओ(-o)*
Must-Continue Markers	
Aspect: Habitual	*-त* (-t), Progressive *रह* (rəh), Completive *चुक* (cuk)
Modal Auxiliaries: Ability/probability	*सक* (sək), ability: *पा (pā), o*bligation: *पड़* (pəɽ), permission: *दे* (de)
Passive	*या* (-yā)/ *यी* (-yī) / *ये* (-ye)/ *जा* (-jā)
Must-End Markers	
Future+gen-num	*-गा (-gā), -गी (-gī),* -*गे* (-ge)
Mood:Imperative	null, -*ो* (-o) ,-*ओ* (-o), *िए* (-ie), *इए* (-ie), *जिए* (-jie), *–ना (-nā)*
Tense Auxiliary: Present	*है* (hai), *हैं (haĩ),* Past: *था (thā), थे* (the), *थी (thī), थीं (thĩ)*
Mood:Conditional	*-त-* (-t-)

Table 1: Intermediate Markers

As shown in 14, the verbal elements appear in a specific order. This ordering issubject to a number of constraints as listed below:

Specific Constraints within a Hindi VG

a) The modal auxiliary *chāhie* must be preceded by an infinitive (with gender-number) marker, such as *khā-nā cāhie (खा-ना चाहिए*). It may be followed neither by an aspect marker (17) nor by a present tense or future tense marker (18). It may only be followed by a past tense auxiliary (19).

17) **chāhie rəh/cuk* (aspect)
18) **chāhie hai/ chāhie-gā* (pres, future)
19) *chāhie thā* (past)

b) In the absence of a modal auxiliary, an infinitive must be followed by a mood or a tense marker (as shown in the examples below in 20 and 21). The expression in 21 where the mood or tense marker is optional is ungrammatical, as in 22.

20) *khā*-ne de-*tā hai*
21) *khā-nā pəɽ-tā hai*
22) **khā-nā (hai/thā/hogā/hotā)*

c) The modal auxiliary *sək* cannot be followed by the perfective marker, the progressive auxiliary *rə*h and the completive auxiliary cuk as shown below in 23-25. It can only be followed by a habitual aspect marker or by a subjunctive marker as in 26 and 27.

23) **khā sək rəhā hai* (progressive)

24) **khā sək-ā hai (perfective)*
25) **khā sək* cuk*ā hai (completive)*
26) *khā sək-tā hai (habitual)*
27) *khā sək*-e (subjunctive)

d) No modal auxiliary may precede the completive auxiliary cuk

28) **khā pā cukā hai*
29) **khānā pəṛ cukā hai*

e) Infinitive marker –*न*-(n), all aspectual markers, past tense auxiliary, conditional mood marker -*त*-(t) and future marker -*ग*-(g) must be followed a gender-number marker as in ता (tā), ती (tī), ते (te), रहा (rəhā), रही (rəhī), रहे (rəhe), चुका (cukā), चुकी (cukī), चुके (cuke), ना (nā), नी (nī), ने (ne), गा (gā), गी (gī), गे (ge).

4.2 Procedure for VG Identification

A VG is identified by scanning the sentence from left to right. The expression given in 30 below is used to detect VGs in a given sentence.

30) Start Marker (Intermediate marker)* Must-end marker

Thus, the start-marker and the must-end markers are obligatory to form a VG while intermediate markers are optional and may recurse (*). The three types of markers were shown in Figure 4 in the previous section. Particles and negation markers may also appear inside a VG. The VG identifier uses the root, suffixes and the morphological features supplied by the morphological analyser and the POS tags assigned by the POS tagger. A VG begins as soon as a verb is scanned and the following morphemes are marked as its suffixes or auxiliaries. The identified verb root may be locally POS ambiguous, i.e., noun or verb (*khānā*'food'and'to eat'), or main verb or auxiliary verb (rəh'live'and *'*progressive auxiliary*'*). The appropriate tag is selected by applying the regular expression on the verbal morphemes. If the sequence of the markers is allowed by the expression, they are included in the VG. The identification continues until a must-end marker is encountered. Once the end of the VG is marked, the group members are assigned fresh, disambiguated POS tags. The head of the VG is assigned VM while the auxiliaries are assigned VAUX along with the TAM features that they express. Some examples are given next.

Types of major POS ambiguity:

a. Main Verb or Auxiliary Verb
b. Main Verb or Noun
c. Main Verb or Postposition

31) [rəh *rəh-ā hai]*
live prog-masc,sg be-pres
'is living'

32) [kər cuk-*ā thā]*
do comp-masc,sg be-past
'had done'

33) kər [cuk-*ā* *de*-g-*ā]*
tax pay-masc,sg give-fut-masc,sg
'will pay the tax'

In 31, *rəh* appears as the progressive aspectual auxiliary as well as a main verb ('live'). Often a POS tagger is unable to resolve this ambiguity in the absence of contextual information. In32, *kər* is ambiguous between being a verb and a noun. As a main verb, it means 'do' and as a noun, it means 'tax'. In order to resolve this POS ambiguity, the system requires the information that when cuk appears as an auxiliary and is followed by a tense auxiliary, it requires preceding main verb. This information rules out the possible tag Noun and leaves Main Verb as the correct one. This information may yield a faulty analysis for the expression in 33. The system will consider *kər*to be a part of the VG and will output the VG as *kərcukādegā*. We require a morphotactical constraint that prevents the completive aspectual auxiliary cuk from being followed by the modal auxiliary de,. We must note that these constraints are ad-hoc and may not always produce correct POS tags.

Secondly, suffixes too may be ambiguous. For example, *'-t'* attached to the stem *ā(*come) may indicate either the habitual aspect or the conditional mood. This ambiguity may be resolved by using the regular expression and by looking at the next morpheme. For example, in 34, the suffix -t- is rejected as being a conditional mood marker as it belongs to the category of must-end markers and cannot be followed by any other verb morpheme (except for the gender-number marker). On the other hand, the habitual -t- may be followed by a tense auxiliary.

34) *bādəl roz [ā*-**te** the]
Clouds everyday come-hab be-past
'Clouds used to form everyday'

During the process of VG identification, feature agreement among elements of the group is also checked. Many invalid sequences are rejected using feature combination rules. For example, *'bhāīthā'* in 36 unlike in 35) cannot be a verb group. It is instead a noun-verb sequence since the masculine gender of the tense auxiliary *thā* does not agree in gender-number with main verb (*bhā 'like'*) marked for feminine gender using *-ī*. On the other hand, *bhāī* (brother) may be a noun with which the gender of the verb (masculine) agrees. The VG identifier thus rejects the Verb tag for the word *bhāī* and retags it as a Noun.

35) *'vo merā bhāī thā'*
he my brother be-past-masc
'He was my brother'

36) **'bhāī thā'*
like-past-fem be-past-masc
'was liked'

Another example where feature checking resolves the POS ambiguity is given in 37 below where the word *'liye'* is POS-ambiguous between a Verb (take-past-pl) and a Postposition (for). If a verb, the form liye should be plural but the tense auxiliary does not agree with it for number(singular, in this case). Thus, it cannot form a VG. By discarding the Verb tag, it is instead assigned the Postposition tag. The VG is formed only with hai 'is'. For the sentence in 38, the word *pāī* may belong to one of two POS categories - Noun (penny) or Verb (found-fem, sg). According to the VG identification rules, negation may appear inside a VG but the perfective must be followed by either a tense marker or a mood marker. The given sequence does not conform to the rule and thus the tag Verb for *pāī*is rejected and a tag Noun is assigned instead.

37) un-*kī* *yojnā shāntipūrnə* uddeshy-õ ke **liye** hai
their plan peaceful aims-obl for be-pres
'Their plan is for peaceful aims'

38) ve ek ***pāī*** nəhĩ *le*-te the
they one penny not take-hab be-past
'They would not take a single penny'

5 Performance Evaluation

We use a CRF based POS Tagger. Without NGI/VGI, the features used for the POS-Tagger include (a) Tag ambiguity scheme from the dictionary, (b) suffix given by the stemmer, (c) prefix and suffix character streams of size one and two, (d) previous word's suffix and (e) tag ambiguity scheme for previous and next word. We tried NG and VG identification at two different places, before and after CRF. When the NGI/VGI module is run before CRF, its output is used as features supplied to CRF and the tags assigned by CRF are considered final. The tag ambiguity scheme of the NG/VG members is simply replaced by the tags given by NGI/VGI modules. On the other hand, when NGI/VGI follows CRF, then NGI/VGI overwrites the tags assigned by CRF.

The Hindi POS Tagger was tested on a corpus of 66,990 words, which is a subset of the BBC Hindi news corpus (downloaded from http://www.bbc.co.uk/hindi) and the IIIT Hyderabad corpus. We partitioned the corpus into four testing folds. The accuracy of the CRF based POS Tag system using Verb Group and Noun Group Identification rules for the four folds are as follows:

Experiment	**Average Accuracy of 4 folds**
CRF	95.18%
CRF + NGI after	95.67%
CRF + VGI after	95.73%
CRF + NGI after + VGI before	95.87%
CRF + NGI after + VGI after	95.26%

Table 2: Experimental Results

We find that while both NGI and VGI help improve accuracy, the best performance is obtained when VGI is applied before CRF and NGI is applied after CRF. It is interesting that applying both NGI and VGI after CRF does not help very much since the errors from one module result in multiple, cascading errors in the second module as the tags given by the first module are considered final. When we apply one module before CRF, then the CRF still gets a chance to overwrite the wrong tags as CRF treats VGI tags as features rather than as final tags.

While a 15% error reduction (from 4.72% to 4.1%) may not appear much numerically large, it should be noted that removing the final 5% of errors is an uphill task with corpus inaccuracies, annotator disagreement, and long distance dependencies dominating. Some of the challenging examples are:

39) mæt∫ 48-48 ovərõ kā kər diyā gəyā hai
match 48-48 overs of do has been be-pres
'Match has been made of 48-48 overs'

Here, the verb group is identified as (diyā gəyā hai). (kər) is marked as a verb whereas in 40, it appears as a noun.

40) mætʃ kā kər diyā gəyā hai
match of tax give-past has been
'Tax has been given/paid for the match'

Another example is that of long distance dependency of the possessive marker:

41) unkā yeh bhī kehnā hai ki
They-ACC this also saying be-pres that
'They also said that'

The verb group is identified as (kehnāhai), whereas (kehnā) is a noun which should co-occur with the preceding possessive. But the possessive pronoun (unkā) is not adjacent to (kehnā).

42) tīm ne spænish līg lā līg kā khitāb jītā
Team-ERG Spanish League Lā Liga of prize win-past
'The team won the Spanish League La Liga title*'*

In 42 (La Liga) is a proper name but as per the morphological analysis (lā) only qualifies to be a verb.

In summary, even with detailed rules for NG and VG identification, there is little improvement in the accuracy of the tagger as (1) our Morphological Analyzer is not able to analyze Compounds (both Verbs and Nouns) and Conjunct verbs as single units unless they are stored in the lexicon,(2) because some of the tags show real ambiguity in a given sentence and 3) because the MA fails to recognize and analyse unknown or foreign words that are not listed in the lexicon.
Our results compare favourably with the 93.45% accuracy reported in Singhet al. (2006) for a CN2 based tagger forthe Hindi BBC news corpus. Guneet al. (2010) report 94% accuracy for CRF on Marathi using a corpus of size 20K. They did not implement NGI but only VGI. They found that use of VGI did not improve the accuracy since not much VM-VAUX ambiguity (their main focus) remained after applying CRF.

6 Conclusions

We have presented algorithms to identify Hindi Noun and Verb Groups by using morphotactical information and the constraints that apply to the constituents of these groups. We also provided the list of grammatical categories and their markers that may appear inside a group and discussed ways in which these markers may be arranged. Group Identification enabled the resolution of major POS ambiguities. The identified groups may also be used at a later stage, i.e., in parsing or in language generation. We cannot handle all the POS ambiguous cases (that involve scrambling or those that are structurally ambiguous) where immediate contextual rules do not help. However, using the ordering among the major categories and their possible combinations, we have tried to present ways that can be applied to other languages equally well. The methods are especially beneficial for languages with meagre corpora or other NLP resources. Since a system will not be able to learn patterns that might be absent in small training corpora, with the useof morphological patterns that govern the ordering of the elements inside a group, a large number of ambiguities and errors may be avoided at a first pass.

Acknowledgements: We would like to thank Nikhilesh Sharma and Neha Gupta for implementing the group identification system discussed here.

References

Abney, S. (1994). “Parsing by Chunks.” In Principle-Based Parsing, eds. B. Berwick, S. Abney, and C. Tenny, 257-278. Dordrecht: Kluwer Academic Publishers.

Baskaran S. (2006). “Hindi POS Tagging and Chunking.” In the Proceedings of NLPAI Machine Learning Contest. Mumbai, India, June.

Begum, R., Jindal K., Jain A., Husain S., Sharma D. (2011). “Identification of Conjunct verbs in Hindi and its effect on Parsing Accuracy.” 12th International Conference on Intelligent Text Processing and Computational Linguistics (CICLing).

Bharati, A., Chaitanya V. and Sangal R. (1995). “Natural Language Processing: A Paninian Perspective.” New Delhi: Prentice-Hall of India.

Chakrabarti, D., Mandalia H., Priya R., Sarma V. and Bhattacharyya P. (2008). “Hindi Compound Verbs and their Automatic Extraction”, In Proc. of Computational Linguistics Conference (COLING), Manchester, UK.

Chakrabarty, D., Sarma V. and Bhattacharyya P. (2007). Complex Predicates in Indian Language Wordnets, Lexical Resources and Evaluation Journal, 40 (3-4).

Dalal, A., Nagaraj K., Sawant U. and Shelke S. (2006). “Hindi Part-of-Speech Tagging and Chunking: A Maximum Entropy Approach.” In the Proceedings of the NLPAI Machine Learning Workshop on Part Of Speech and Chunking for Indian Languages. Mumbai, India.

Grover, C. and Tobin R. (2006). “Rule-based Chunking and Reusability.” In the Proceedings of LREC 2006, 873-878. Genoa, Italy.

Gune H., Bapat M., Khapra M. and Bhattacharyya P. (2010). “Verbs are where all the Action Lies: Experiences of Shallow Parsing of a Morphologically Rich Language”, Computational Linguistics Conference (COLING), Beijing, China.

Halliday, M. A. K. (1977). “Text as Semantic Choice in Social Contexts.” In Grammar and Descriptions (Studies in Text Theory and Text Analysis), eds. T. A. van Dijk and J. Petofi, 176-225. New York: Walter de Gruyter.

Kachru, Y. (2006). Hindi. Amsterdam and Philadelphia: John Benjamins.

Kutlu M. (2010). “Noun Phrase Chunker for Turkish using Dependency Parser”, MS Thesis. Department of Computer Engineering, Bilkent University.

Ramshaw, L. A. and Mitchell, P. M. (1995). “Text Chunking Using Transformation-Based Learning.” In the Proceedings of the Third ACL Workshop on Very Large Corpora, 82-94. Cambridge, MA, USA.

Ray, P. R., Harish V., Basu A. and Sarkar S. (2003). “Part of Speech Tagging and Local Word Grouping Techniques for Natural Language Parsing in Hindi.” In the Proceedings of (ICON). Mysore, India.

Singh, A., Bendre S. M. and Sangal R. (2005). “HMM Based Chunker for Hindi.” In the Proceedings of International Joint Conference on NLP.

Singh, S., Gupta K., Shrivastava M. and Bhattacharyya P. (2006). “Morphological Richness Offsets Resource Demand – Experiences in Constructing a POS Tagger for Hindi.” In the Proceedings of the COLING/ACL-2006, 779-786. Sydney, Australia, July.

Vijay S. and Sobha D. (2010), "Noun Phrase Chunker Using Finite State Automata for an Agglutinative Language", In the Proceedings of the Tamil Internet Conference.

Named Entity Recognition System for Urdu

UmrinderPal Singh[1,1], Vishal Goyal[2,1] Gurpreet Singh Lehal[3,1]
(1) Department of Computer Science, Punjabi University Patiala
umrinderpal@gmail.com[1], vishal.pup@gmail.com[2], gslehal@gmail.com[3]

ABSTRACT

Named Entity Recognition (NER) is a task which helps in finding out Persons name, Location names, Brand names, Abbreviations, Date, Time etc and classifies them into predefined different categories. NER plays a major role in various Natural Language Processing (NLP) fields like Information Extraction, Machine Translations and Question Answering. This paper describes the problems of NER in the context of Urdu Language and provides relevant solutions. The system is developed to tag thirteen different Named Entities (NE), twelve NE proposed by IJCNLP-08 and Izaafats. We have used the Rule Based approach and developed the various rules to extract the Named Entities in the given Urdu text.

Keywords: NER, Urdu NER, Urdu, Rule Based

Proceedings of COLING 2012: Technical Papers, pages 2507–2518,
COLING 2012, Mumbai, December 2012.

1. Introduction

Named Entity Recognition (NER) is a subtask of Information Extraction (IE). NER extracts and classifies the true Named Entities in text. NER system is widely used in different tasks of Natural Language Processing (NLP) and in many commercial applications on internet like Search Engine. Accuracy of NER system is directly reflected in NLP applications. So, accurate working of NER system is very important. NER system can be used for one's personal interest like company manager wants to know all the names involved in specific text document.

S.No:	Tag Name	Example
1	Person Name	ارشد(Arshad)
2	Location	پٹیالا (Patiala)
3	Organizations	رلاینس (Reliance)
4	Terms	سپنڈلائٹس (Spondylitis)
5	Designation	وزیر اعظم (President)
6	Title Person	جناب (Mr.)
7	Title Object	ہندوستان ٹائمز (Hindustan Time)
8	Brand	سیمسنگ (Samsung)
9	Measure	10 سال (10 Years)
10	Number	ایک، دو (One, Two)
11	Date/Time	12 اکتوبر (12 October)
12	Abbreviation	بی بی سی (BBC)

TABLE 1- Different Named Entity Tags

Named Entities mentioned above were proposed at IJCNLP-08 workshop [17]. Named Entities can be domain specifics like NER system to identify entities in scientific data.

The NER system can be developed using three approaches, 'Rule-Based', 'Machine Learning' (HMM, SVM, CRF, Decision Tree) and 'Hybrid' approach. The Rule-Based system is difficult to develop as one should know the language and grammar rules. These kinds of systems are domain specific. Machine learning approach provides different Statistical NLP tools to train NER system. Statistical tools provide fast way to develop NER system but the accuracy of the system is dependent on annotated training data. For greater accuracy we need to train the NER system with large amount of annotated data. Hybrid approach is a combination of both Rule Base and Statistical based.

2. Related Work

NER system came in focus during the sixth Message Understanding Conference (MUC-6) [6]. After that many NER systems were developed. Most of these systems were developed for European languages and all systems were highly accurate. For south Asian languages, NER systems yet in developing phase. IJCNLP-08 workshop played a major role in development of NER Systems for Indian languages. This Workshop focused on five languages i.e. Hindi, Bengali, Oriya, Telugu and Urdu. All the systems were developed using Statistical approaches or Hybrid approach. Hybrid NER system for five languages was developed by (Sujan Kumar saha et al. 2008) [2]. Rules were developed only for Hindi and Bengali. The system was developed

using MaxEnt model. Accuracy for Urdu was Maximal, Nested and lexical were 27.79, 28.59 and 35.47 respectively.

Karhik Gali et al.2008 [18] had developed the system for five languages Telugu, Hindi, Bengali, Urdu and Oriya. The system was developed using CRF based machine learning model. This system also used some heuristic rules. The system was specific for Telugu and Hindi. Accuracy for Urdu was Maximal, Nested and lexical, were 39.86, 39.01 and 43.46 respectively. Asif Ekbal et al 2008 [1] had developed the system using CRF Machine learning approach. The system was trained for Bengali, Hindi, Telugu, Oriya and Urdu. The system also used language dependent and language independent rules. Accuracy of the system for Urdu was Maximal nested and lexical, were 30.35, 28.55 and 35.52 respectively. Praveen Kumar P et al 2008[3] had developed the system for Hindi, Bengali, Oriya, Telugu and Urdu languages. The system was developed using Hybrid approach which was a combination of CRF and HMM models. Accuracy of the system for Urdu by using CRF, Maximal Nested and Lexical were 33.17, 31.78 and 38.25 respectively and by using HMM 34.48,36.83 and 44.73 respectively . Amit Goyal 2008[10] had developed the system using CRF machine learning model for Hindi language. Accuracy of the system was 58.85. Shilpi Srivastava et al. 2011 had developed the NER system for Hindi language based on CRF and MaxEnt models of Machine Learning approach and rules were developed for Hindi language. The system used voting method to improve the accuracy. Accuracy of the system was 82.95. Kashif Riaz et al. 2010[4] had developed the system using the Rule Based approach. Rules were developed for Person name, Location, Date, Numbers, Organizations and Person's designations tags. The system used very small gazetteer for person names and locations. Recall of the system was 90.7%, precision 91.5 and F1-measure was 91.1%. which was better than all the NER systems developed in IJCNLP-2008 workshop for Urdu? We can see that sufficient work was not done for Urdu NER system and work which is available does not show satisfactory results. Only Kashif Riaz's work shows good results.

3. Approaches to NER

3.1 Rule Based approach: Rules are developed to identify NE in text. This approach takes much time in development and one should have good knowledge of target language. Heuristic based rules are used to identify tags and these rules are language specific. Good rules always yield good results. Development of these kinds of systems is always a time consuming task.
3.2 Statistical approach: Statistical approach is also known as Machine Learning approach. This is a fast way to develop a NER system. The system is trained using annotated training data set in specified format. Accuracy of statistical approach is dependent upon the training data. So, we always train the system with a large set of annotated data. Various Machine Learning models like HMM, CRF, MaxEnt, are used for NER system.
3.3 Hybrid system: Hybrid system is combination of Rule Based approach and Statistical approach. To develop the Hybrid system we use Statistical tools as well as linguistic rules. Combinations of both approaches make a system more accurate and efficient.

4. Issues in Urdu NER System

No capitalization: Urdu and other Asian Languages do not have concept of capitalization. In European language like English this feature is widely used to recognize Named Entity in text because all the names in text always begin with capital letter. Absence of capitalization feature makes the NER task hard for Urdu language.

Ambiguous Name: Urdu language has lots of ambiguous names that can be used as proper noun as well as common noun. Main challenge of any NER system is to separate or extract proper noun in place of common noun. Example: برکت (barkat) or سلامت (slaamat) can be the name of a person or it can be used as common noun.

Spelling variations: Lack of standardization in Urdu language can be seen in spelling as well. There are different spellings that can be used for same word. Like word Hospital can be written in two ways in Urdu اسپتال/ہسپتال (hasptaal/asptaal) which makes the task difficult for NER. We are unable to collect the standard spelling of foreign language. Example: انسٹیچیوٹ/انسٹیٹوٹ (institutes/instichutes).

Non-availability of resources: Language Resources are must for any approach whether it is Rule Based or Statistical. There is no large gazetteer and annotated data available for Urdu language.

5. Why Rule Based Approach

Rule Based approach is time consuming task to develop any NER system. Rule based approach is used only when you know the target language well and have sufficient knowledge about the linguistic rules like knowledge of grammar. The system developed using Rule Based approach always yields the good results. On the another hand, Statistical approach which provide us with many Statistical tools, to develop NER system like HMM, CRF, SVM, MaxEnt etc, with the help of these tools development process of the system is rapid as compared to Rule Based approach. We have studied that in IJCNLP-2008 that all the NER systems were developed using different Statistical approaches. But none of the system provides good results for Urdu text because annotated data provided by the workshop is only 36000 Urdu tokens which is not sufficient to train Urdu NER system. New Statistical techniques like CRF not perform well for Urdu. Absence of any large Urdu gazetteers is also one of the reasons for low accuracy. These kinds of gazetteers boost the accuracy of Statistical approaches. Rule Based approach used by (Kashif Riaz 2010[4]) for Urdu language shows good results. Rules are used to identify six tags. Workshop on NER system by IJCNLP 2008 focused on twelve NER tags. By studying various research papers we concluded that we should follow Rule Based approach though it is a time consuming approach but this approach will give us promising results as we have seen in Kashif Riaz's[4] system. We did not try Hybrid Approach because the absence of large annotated corpus for Statistical part. We have tried to develop different rules for all 12 NE tags which are used in IJCNLP-08 workshop.

6. Rule Based Model

Following Rules are used to identify different tags in Urdu text.

1. Rules are applied to identify date and time tags. These kinds of tags are easily identified by Regular Expressions that are created for specific patterns like 01.01.2012 or 01/01/2012 and time is also identified as 11:20 or 01:22. The system is able to identify the date like 01 May 2012 or 01 May and year 2012.

2. Suffix matching is used to identify various locations and types of names and terms. In Urdu language and other south Asian languages, there are many location names that end with 'pur' (Kishanpur, Rampur), 'stan' (Pakistan, Hindustan) 'ghar' (Chandighar, Ramghar), 'nagar'

(Sonagar) and words that end with 'abad' (Fridabad, Hardabad). Suffix matching is also used for persons name, terms and org Like, person name that ends with 'dev' (Ramdev, Shamdev), 'das' (Sumitvadas, Charndas). Terms that ends with 'logy' (biology), person's last name ends with 'brown' or 'wood' then we can identify them as person's name or it may be organization like Hollywood and Bollywood.

3. The system uses gazetteer of most common person names to identify 'Person Names tag'. The system is able to tag words of maximum three lengths as one Named Entity like محمد شفیق تھند (mahmad saphīk thindh). For person names we have collected 4500 Urdu names and 1500 Hindu person names.

4. We have collected the surname of Muslim and Hindu religions. With the help of surnames the system is able to identify his/her first name, like surname (khān)خان helps the system to check one word before the surname which may be the first name of person like (shāhrukh khān) شاہ رُخ خان

5. Title person and Designations helps the system to identify person name like Title Person وزیر اعظم (vajīr-ē-ājam) and مسٹر (miṣtar) that may have proper name next to it. With the help of Title Person and surname, the system is able to detect Person Named Entities easily. The system is also able to identify those person names which are not part of the gazetteer. We have collected 34 Title Persons and 102 Designations.

6. The NER system performs well when it can identify true Named Entities by resolving the ambiguities. Our NER System is able to identity true Person Named Entity based on various rules like if the system encounters any ambiguity in person name it will treat it as a special case and apply different rules to make sure that it is a true person name. For example when system encounter the word کمل (kamal) then system try to resolve the ambiguity of this word with help of postposition as surname or preposition where it may found Title Person or Designation. If there is no clue to identify as Named Entity then at last it check out the post position of ambiguity word likeکمل کو گھر پے کام تھا (kamal kō ghar pe kam tha.) the word (kō) کو give us clue that it may be the person name , So the system tag it as Person Named Entity(PNE).

7. Rule is also used to identify numbers that are non numerals like 'چار'چھے'پانچ' (Four, Five, and Six). The system able to tags three words as one Number Entity likeسو پانچ تین (Three Hundred Five).

8. Person name may have abbreviations in the place of first name of person like کلام اے جے پی اے (A P J A Klam). If the system is able to identify surname alone then it always try to find it as abbreviation name.

9. The system is able to find out and tag abbreviations like (C P U) سی پی زو (B B C) بی بی سی etc.

10. Organizations are tagged during gazetteer look up. We have insufficient data related to organizations so we have used some heuristics to identify ORG tags. For Example if text includes Org (vō pañjābī yūnīvrasiṭī kā ṭālī-ē-ilam hai.)وہ پنجابی یونیورسٹی کا طالب علم ہے and we don't have this Organization in gazetteer then system apply rules to find and tag org as "Punjabi university".

7. Algorithm for Urdu NER

We have developed the system on Windows platform using Dot Net framework 4. System is using other available classes of framework to implement all features of our NER system. Like Tokenizer and Linked List class and its functions. We have developed many other modules for different rules used by the system. The system works in linear complexity.

1. Input text, through file upload or user may type text in given Text Field.
2. Normalization of Input Text
   1.1 Remove Extra spaces to single space.
   1.2 Remove special chars from end of the strings.
3. Gazetteer lookup
   3.1. Gazetteer lookup up for Locations, Terms, Brands, Abbreviations and Organizations Tags.
4. Tokenized and Normalized
   4.1. Tokenized the input Text word by word and search against the Gazetteer
5. Search for Date and Time tags
   5.1 Search for Number (numeral), Date, Time, Email and URL Tags.
6. Rule to tag Person Names
   6.1 Rule to detect Person Name with the help of Title Person Name, Designation and Surname without using Gazetteer.
7. Suffix stripping is used
   7.1 Suffix Striping Rules are used to Detect Location Names, Organizations, Izaafats, and Some types of Person names.
8. Find Person Names and Numbers
   8.1 Find more Person Names through Various rules and Gazetteer lookup.
   8.2 Rules are applied for names up to three words of length.
   8.3 Rule to detect Abbreviation Names.
   8.4 Rule is applied to resolve ambiguity in names.
   8.5 Rule is applied to find Numbers in non numerals form.
9. Rule is applied to find out Abbreviations which were not found during Gazetteer search.
10. Rules are applied to find out Organizations
    10.1 Those Organizations entity which were not found during Gazetteer lookup, rule will try to find out and tag them as organization entities.
11. Show tagged output to user along with untagged data.

Algorithm is self explained; still lightening some of its steps regarding Gazetteer look up. Gazetteer lookup is used to find out various Named Entities in text. We have collected Named Entities related to various fields i.e. Politics, Business etc. In algorithm's step 3 gazetteer look up is used for Locations, Terms, Brands, Abbreviations and Organizations Tags. All these tags are less ambiguous so the system tags them without applying any rule. System have gazetteer list related to these tags which are not ambiguous. In Step 4 system tokenized the input text and normalized the tokens for further processing. Step 6, 7 and 8 are used to tag Person Names. In Step 6, algorithm finds out person names based on Title Person, Designation and Surname. In this step system is able find out Person Named Entities without any gazetteer list. In Step 8, system used gazetteer list to find out person names and apply various rules to resolve their ambiguity. Some person names have patterns in its suffix or in prefix of the word. So, Step 6 of algorithm finds out this kind of person name by using suffix stripping.

8. Evaluation Metrics

Standard evaluation metrics for Information Retrieval includes Precisions, Recall and F-measure.

Recall: Relevant information extracted from text. Recall defined as:

Recall: = No. of correct answers given by system / Total No. of possible correct answers in text.

Precision: Actual correct answers returned by system. Precision defined as:

Precision: = No. of correct answer/No. of answers given

F-Measure: Balances of Recall and Precision by using a parameters **β**. The F-measure is defined as:

F-*measure=(β*^{2}*+1)RP/(β*^{2}P+R)**

β is weighted as **β**=1. When **β**=1 F-measure is called F1-measure. The F1-measure is defined as:-

F1-measure=2*RP/P+R

9. Evaluation and Results

We have constructed two sets of test data. Test data is collected from different websites [19] of Urdu. Test data mainly include News from different fields like Politics, Sports, Business and Science. The reason of collecting News data is that because News data is always full of Named Entities. So, it gives challenging job to our NER system to identify all different kinds of NE tags accurately. Test data also include ambiguous data, our NER system tried to resolve ambiguities and tag only true entities. Mainly ambiguities are in person names and the system resolves them by applying different rules. For evaluating the system we have tested the system on two different test data sets. Both test data sets have news and articles related to different domain. Test data set 1 have data related to political news, some articles and short stories. Test data set 2 mainly have news related to science and business.

Test Case	Number of tokens	Domain
Test set 1	12032 tokens	News and articles related to politics.
Test set 2	150243 tokens	News data related to science topics, business news.

TABLE 2-Test Sets and along with number of tokens and their domain

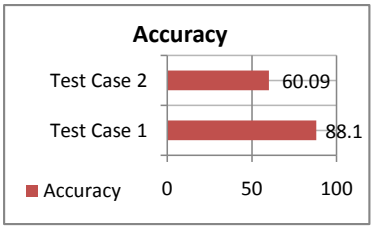


FIGURE 1- Performance of different test cases

Test Sets	Precision	Recall	F1-Measure
Test set 1	58.15	62.05	60.09
Test set 2	86.17	90.40	88.1

TABLE 3- Result of Test cases

Accuracy of Test set 1 is 88.1% and Test set 2 is 60.09%. Note that the frequency of occurrence of tags like Terms, Title Object, Brand Name are very less and we have not sufficient collected data related to these tags. Accuracy without all these four tags is shown below.

NE Tags	Precision	Recall	F1-Measure
NEP(Person Name)	92.85	93.37	93.10
NEL(Location)	85.00	90.00	87.28
NEO(Organizations)	77.70	81.30	80.37
NED(Designation)	86.80	89.21	87.98
NETP(Title Person)	85.33	88.45	86.85
NEM(Measure)	88.24	89.59	88.87
NEN(Number)	92.84	93.50	93.16
NETI(Time)	90.36	91.32	90.83
NEA(Abbreviation)	89.85	91.59	90.71

TABLE 4- Individual results of Nine NE tags

NE Tags	Precision	Recall	F1-Measure
NETE(Terms)	32.20	34.49	33.30
NEB(Brand Name)	43.45	47.28	45.34
NETO(Title Object)	51.49	53.22	52.35
NEIZ(Izaafats)	31.25	35.29	33.14

TABLE 5- Individual results of Four NE tags

Results shown in Table 5 for four tags are not good as compared to other tags because these tags need more time to collect accurate data. If we consider accuracy of all the thirteen tags then accuracy is 74.09%. Accuracy of the system also depends upon the domain of testing data. Test set 2 includes scientific and business terms so that system was not able to perform well. We have considered thirteen tag as compared to twelve tags used in IJCNLP-08. The system tagged Izaafat words because in Urdu, Izaafats are used very frequently and when we translate Urdu to target languages then we need to translate these Izaafats in specific target language words. Like (aab-e-hayat) آب حیات izaafat meaning in English is sacred water. But some time izaafat plays the role as NE for example وراثت خالصہ (vīrāsat-ē-khālsā) name of a place and it should not be translate in target language but transliterate it.

Conclusion and Future Work

We have developed the system to tag different Named Entities and system is able to find out and tag them all. But system has some limitations too.

1. We have developed the rule to tag person's name having length of three words. If person name has longer string of words like four words in a name then it will tag three words as one Named Entity and fourth one as another Named Entity. For Example in محمد أصف علی زرداری (mohmad āsīph alī jardārī) person name, زرداری (jardārī)will be tagged separately and محمد آصف علی (mohmad āsīph alī)will be tagged as another Named Entity. Count of Entities will be two in place of one.

2. Same is the case with Number tag. Like: دو سو چوّن (Two Hundred fifty four) tagged as one named entity but if we have longer string having more than three words like دو ارب چوّن لاکھ (Two Thousand Fifty Four Lakh) will be tagged as two separate entities. Where (Lakh) لاکھ will be tagged as a separately.

3. Partial tagging problem in Org tag like (انڈین انسٹیوٹ آف ٹیکنالوجی) Indian Institute of Technology will tagged partially as (Institute of Technology) word Indian will not be tagged with that Org tag.

4. Problem in Date tag, some time Date is written in word form like: بیس جنوری دویار گیارہ (Twenty January Two Thousand Eleven.) This kind of string will not be tagged as Date tag but will be treated as separate tags like January as Time tag and Twenty as Number tag and Two Thousand Eleven as one Number tag.

5. System is not using any kind of technique to resolve the segmentation problem in given Urdu text. Like other Asian languages Urdu has problem of space omission and space insertion. We will try to improve our NER system by using effective segmentation technique in near future.

6. System is not using POS Tagger or POS tagged data. So system always has to do larger number of comparisons to find out entities in given text. It is difficult to make any decisions for system based on rules because system compares only words but not their part of speech. Due to large numbers of comparisons system is little bit slowly when we gave large amount of input text to system. POS tagged data and stemmer is very essential for NER system and we have not used any POS tagger for our NER. So, we will develop POS tagger to include it in our system.

7. There is a problem to resolve ambiguity of person's name. We have collected the data of ambiguous person names and system treats them as special case to resolve its ambiguity but some time all the rules fail to remove its ambiguity. For example فتح کو بلاوا دو (phatah kō būlāva dō) here word فتح (phatah) can be proper noun or common noun. We do not have any clue to find out whether it is a person name or not. Rule checked Title Person, Designations or surname if all these conditions are not present in given sentence then we need to check out postpositions, but some time these postpositions come along with common name.

We have collected 6000 person names gazetteer. We will collect more number of person names and will try to include surname of other languages. Main problem is gazetteers of terms related to various domains. Collected data is not sufficient. We do not have standard spelling of terms related to various fields. We will try to collect standard spellings of different terms.

To develop the Urdu NER system we have insufficient resources as we discussed earlier but still we are able to get good accuracy. In future we will try to develop other essential tools for Urdu NER.

References

[1] Asif Ekbal, Rejwanul Haque, Amitava Das, Venkateswarlu Poka and Sivaji Bandyopadhyay.(2008). *"Language Independent Named Entity Recognition in Indian Languages"* In Proceedings of the IJCNLP-08 Workshop on NER for South and South East Asian Languages, Hyderabad, India. Asian Federation of Natural Language Processing, pp. 33–40.

[2] Andrew Borthwick, John Sterling, Eugene Agichtein and Ralph Grishman *"Description of* the MENE Named Entity System as Used in MUC-*7"* http://acl.ldc.upenn.edu/muc7/M98-0018.pdf Accessed on December 2011.

[3] Amit Goyal.(2008) *"Named Entity Recognition for South Asian Languages"* In Proceedings of the IJCNLP-08 Workshop on NER for South and South East Asian Languages, , Hyderabad, India. Asian Federation of Natural Language Processing, pp 89–96.

[4] Anil Kumar Singh.(2008) *"Named Entity Recognition for South and South East Asian Languages: Taking Stock"* In Proceedings of the IJCNLP-08 Workshop on NER for South and South East Asian Languages Hyderabad, India. Asian Federation of Natural Language Processing, pp 5–16.

[5] Animesh Nayan, B. Ravi Kiran Rao, Pawandeep Singh, Sudip Sanyal and Ratna Sanyal.(2008). *"Named Entity Recognition for Indian Languages"* In Proceedings of the IJCNLP Workshop on NER for South and South East Asian Languages, Hyderabad, India. Asian Federation of Natural Language Processing, pp. 97–104.

[6] Anil Kumar Singh.(2008) "Named Entity Recognition for South and South East Asian Languages: Taking Stock" in Proceedings of the IJCNLP Workshop on NER for South and South East Asian Languages, pp 5–16.

[7] Ali Elsebai Farid Meziane and Fatma Zohra Belkredim.(2009). *"A Rule Based Persons Names Arabic Extraction System"* In Proceeding Communications of the IBIMA Volume 11.

[8] Bowen Sun *"Named entity recognition Evaluation of Existing Systems" Accessed* November 2011 http://daim.idi.ntnu.no/masteroppgave.pdf

[9] Duangmanee (Pew) Putthividhya.(2011). *"Bootstrapped Named Entity Recognition for Product Attribute Extraction"* In Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing, Edinburgh, Scotland, UK. Association for Computational Linguistics, pp 1557–1567.

[10] David Nadeau, Satoshi Sekine. "A survey of Named Entity Recognition and classification" Accessed on November 2011 http://nlp.cs.nyu.edu/sekine/papers/li07.pdf

[11] Kashif Riaz.(2010). University of Minnesota Department of Computer Science Minneapolis, MN, USA *"Rule-based Named Entity Recognition in Urdu"* In Proceedings of the Named Entities Workshop, Uppsala, Sweden. Association for Computational Linguistics, pp 126–135.

[12] Karthik Gali, Harshit Surana, Ashwini Vaidya, Praneeth Shishtla and Dipti Misra Sharma.(2008). *"Aggregating Machine Learning and Rule Based Heuristics for Named Entity Recognition"* In Proceedings of the IJCNLP Workshop on NER for South and South East Asian Languages, Hyderabad, India. Asian Federation of Natural Language Processing, pp. 25–32.

[13] Karthik Gali, Harshit Surana, Ashwini Vaidya, Praneeth Shishtla and Dipti Misra Sharma.(2008). "Aggregating Machine Learning and Rule Based Heuristics for Named Entity Recognition" In proceedings of the IJCNLP-08 Workshop on NER for South and South East Asian Languages, Hyderabad, India. Asian Federation of Natural Language Processing , pp. 52-32

[14] Praveen Kumar P.(2008). *"A Hybrid Named Entity Recognition System for South Asian Languages"* In Proceedings of the IJCNLP-08 Workshop on NER for South and South East Asian Languages,Hyderabad, India. Asian Federation of Natural Language Processing, pp.83–88.

[15] Pramod Kumar Gupta and Sunita Arora(2009) *"An Approach for Named Entity* Recognition System for H*indi":* An Experimental Study In Proceedings of ASCNT CDAC, Noida, India, pp. 103 – 108.

[16] Sujan Kumar Saha, Sanjay Chatterji ,and Sandipan Dandapat.(2008). *"A Hybrid Approach for Named Entity Recognition in Indian Languages"* In Proceedings of the IJCNLP-08 Workshop on NER for South and South East Asian Languages, Hyderabad, India. Asian Federation of Natural Language Processing , pp. 17–24.

[17] Wenhui Liao and Sriharsha Veeramachaneni.(2009) *"A Simple Semi*-supervised Algorithm *For Named Entity Recognition"* In Proceedings of the NAACL HLT Workshop on Semi-supervised Learning for Natural Language Processing, Boulder, Colorado. Association for Computational Linguistics, pp. 58–65.

[18] http://en.wikipedia.org/wiki/Urdu Accessed on March 2012

[19] www.bbc.co.uk/urdu/ Accessed on March-May 2012

Easy-first Coreference Resolution

Veselin Stoyanov[1] *Jason Eisner*[1,2]
(1) HLT-COE, Johns Hopkins University
(2) CLSP, Johns Hopkins University
{ves,jason}@cs.jhu.edu

Abstract

We describe an approach to coreference resolution that relies on the intuition that easy decisions should be made early, while harder decisions should be left for later when more information is available. We are inspired by the recent success of the rule-based system of Raghunathan et al. (2010), which relies on the same intuition. Our system, however, automatically learns from training data what constitutes an easy decision. Thus, we can utilize more features, learn more precise weights, and adapt to any dataset for which training data is available. Experiments show that our system outperforms recent state-of-the-art coreference systems including Raghunathan et al.'s system as well as a competitive baseline that uses a pairwise classifier.

KEYWORDS: coreference resolution, discourse processing, supervised clustering, greedy approaches.

KEYWORDS IN L_2: resolución de correferencia.

Proceedings of COLING 2012: Technical Papers, pages 2519–2534,
COLING 2012, Mumbai, December 2012.

1 Introduction

Coreference resolution has traditionally benefited from machine learning approaches (Soon et al., 2001; Ng and Cardie, 2002; Bengtson and Roth, 2008; Stoyanov et al., 2009). Surprisingly, however, recent work has shown that simple rule-based coreference systems can compete with the state-of-the-art machine-learning-based systems if provided with rich lexical, syntactic, semantic and discourse information (Haghighi and Klein, 2010; Raghunathan et al., 2010). In fact, the rule-based system of Raghunathan et al. (2010) exhibited the top score in the recent CoNLL evaluation (Pradhan et al., 2011). This system's innovation is to build the coreference clusters incrementally, starting with the most precise rules (dubbed *sieves* by Raghunathan et al. (2010)) and use the available coreference information to guide the less precise sieves.

Coreference resolution is inherently a global task: for example, discovering that *Smith* and *she* corefer makes it more probable that *Smith* corefers with *Jane Smith* (i.e., a female name) rather than *Jason Smith*. Furthermore, grouping *Jane Smith*, *she* and *Jason Smith* in the same cluster should be rejected by the clustering algorithm because it results in poor gender and proper name agreement. Yet incorporating such structured information in coreference has proven challenging. There have been several successful attempts to incorporate structured information through joint inference such as Culotta et al. (2007) and Poon and Domingos (2008), but they do not explicitly learn parameters tuned to the inference algorithm used at test time (and the latter relies on a complicated and expensive inference procedures). The difficulty of inferring globally consistent clusterings lies in the fact that there are exponentially many clusterings and producing an "optimal" clustering according to most measures of global coherence is NP-hard. In this context, the approach of Raghunathan et al. (2010) can be seen as a rule-based greedy search for a globally consistent clustering.

We propose a coreference resolution approach that like Raghunathan et al. (2010) aims to consider global consistency while performing fast and deterministic greedy search. Similar to Raghunathan et al. (2010), our algorithm operates by making the easy (most confident) decisions first. It builds up coreference clusters as it goes and uses the information from these clusters in the form of features to make later decisions. However, while Raghunathan et al. (2010) use hand-written rules for their system, we learn feature weights from training data.

What do the learned weights *mean*? They tell our system how to make the merging decisions as it performs greedy agglomerative clustering. And how do we *learn* the weights? Inspired by Goldberg and Elhadad's (2010) approach to easy-first dependency parsing, we utilize a learning method that performs supervised perceptron-style updates as it carries out clustering. Thus, during training, the learner observes partially completed clusterings similar to those that are likely to be encountered during testing.

Our approach inherits all of the advantages of discriminatively trained systems: tuning of parameters based on the empirical properties of training data (instead of hand-tuning weights); an ability to easily adapt to different datasets for which training data is available; an ability to utilize arbitrary overlapping features of the data. In addition, like Raghunathan et al. (2010), we are able to utilize information from earlier, more certain steps when making later, less certain decisions. Our experimental section compares both to traditional ML-based approaches that do not utilize incremental information and to the approach of Raghunathan et al. (2010), which is incremental but utilizes no ML. Under the same evaluation settings, our algorithm outperforms a competitive machine-learning baseline that uses a pairwise classifier, under 9 out of the 10 evaluation settings. Additionally, our system outperforms that of Raghunathan et al.

by more than 2 points on all 4 evaluation settings in which we are able to compare directly.

2 Coreference Resolution

Noun phrase coreference resolution is the task of determining whether two noun phrases (NPs) refer to the same real-world entity or concept. In this paper we will be concerned with determining coreference only within a document. Following established terminology, we use the term **mention** to refer to a linguistic expression that may participate in the coreference relation, as defined for the particular task. Typically mentions are noun phrases, but definitions vary by dataset and may include nested nouns, gerunds, etc. (see Stoyanov et al. (2009) for details). Our method is agnostic to the particular definition of what constitutes a mention, provided that it can rely on a component that extracts mentions with reasonable accuracy. We will use the terms **coreference chain** (or just **chain**) and **cluster** interchangeably to mean a set of mentions that have been posited to refer to the same entity.

The field of coreference resolution has been dominated by the mention-pair model (Soon et al., 2001; Ng and Cardie, 2002; Bengtson and Roth, 2008; Stoyanov et al., 2009). This approach trains a classifier to decide whether a pair of distinct mentions is coreferent or not. The quadratically many decisions about all pairs are then reconciled using a clustering algorithm to form the predicted coreference chains. Different features, learners, and clustering algorithms have been employed in the literature. Surprisingly, the mention-pair model with a simple clustering algorithm such as single-link clustering (i.e., transitive closure) performs on par with state-of-the-art systems (Bengtson and Roth, 2008; Stoyanov et al., 2009).

The pitfall of the mention-pair model and other algorithms that rely only on "local" information between pairs of mentions is that they cannot consolidate structured information. In fact, it is common in practice that some chains produced by such algorithms exhibit low coherency. (E.g., a cluster may consist of [*Jason Smith*, *Smith*, *Smith*, *she*], which looks good to the clustering algorithm because 5 of its 6 pairs are plausible. See section 4.4 for other examples.) In addition to negatively impacting evaluation scores, such clusters are especially irritating to human users of the system output.[1] Recent work has attempted to overcome the limitation of local models. Culotta et al. (2007) and Poon and Domingos (2008) perform global clustering of all mentions in a document by using first-order probabilistic models in the supervised and unsupervised settings respectively. However, these models do not specifically tune their weights to their respective test-time inference procedures – the method of Poon and Domingos (2008) is unsupervised, while Culotta et al. (2007) learn a scoring function that can judge the goodness of an overall clustering, but it is not trained to judge goodness incrementally as the algorithm progresses.

The model of Raghunathan et al. (2010) uses an approach based on *multiple sieves* of decreasing precision and increasing recall. It begins by creating a coreference chain for each mention in the document. Each sieve consists of deterministic tests that are applied to pairs of chains in the current clustering. When the tests succeed, the coreference chains in the pair are joined together. The sieves are manually designed and tuned manually on development data.

Another approach has considered a set of hand crafted rules — the multi-agent method of Zhou and Su (2004). Like Raghunathan et al., it relies on multiple agents that filter potential antecedent candidates. However, the system of Raghunathan et al. differs in that coreference

[1] Based on our personal experience deploying coreference resolution systems.

decisions are not made sequentially but in order of the precision of the corresponding sieves, with later (less precise) sieves relying on the information from earlier sieves.

3 Easy-first Coreference

The intuition behind the multi-pass sieve coreference system is that some decisions are easier than others. For instance, the first sieve joins together mentions that constitute the exact same string, while the second sieve looks for high-precision constructs such as appositives and predicate-nominal relations. The later sieves that make harder decisions (for instance, pronoun resolution is done by the last sieve) are joining larger clusters and so can exploit more complete information about the entities being referred to. We rely on the same intuition, but we allow the system to automatically learn what constitutes easy decisions based on features of the clusters that are to be joined. We hope that the system will learn based on the statistics of the training data, in addition to the human intuition that goes into designing the features. For example, we hope that our algorithm can learn to link *Smith* and *she* early in certain contexts (e.g., if *Smith* is the only possible antecedent for the pronoun *she*). Additionally, in contrast to a system for which rules are designed manually, our approach can utilize a large number of features, using the learned weights to aggregate evidence from different features.

We will describe our algorithm by first discussing the test-time system and then the learning algorithm that we use. We face a reinforcement learning setting. Our agglomerative clustering "agent" observes a current *state*, which consists of all current partially formed coreference chains. In the *start state*, each mention is a separate, single-element chain. At each step, the agent will select some action of the form JOIN_{ij}, which joins existing chains i and j (changing the state and the set of actions available at the next time step). Alternatively , the agent may select the special action HALT, which stops the computation and returns the current coreference chains.

3.1 Test-time Inference

At test time our agent operates greedily (without lookahead), choosing its action at each step according to a linear model. The main loop of the test-time algorithm without optimizations discussed below is shown in Algorithm 1. For each action JOIN_{ij} that is available in the current state C, we compute a feature vector $\phi(\text{JOIN}_{ij}, C)$. In this work, ϕ ignores most of C and extracts only features that depend only on the two clusters i and j. However, in the future work section we discuss features that express the relative confidence of linking clusters i and j as *compared to* the alternatives (i.e., is i the only reasonable antecedent for cluster j?). Similarly, $\phi(\text{HALT}, C)$ defines a feature vector for the HALT action. At present, there is a single dedicated feature that always fires on this action, but in future work we can use additional features that consider whether it is a good idea to HALT in the current state or to continue merging clusters.

The score of an action a in state s is given by a linear combination of the features, $w \cdot \phi(a, s)$, where w is a weight vector (coefficient) that specifies a weight for each feature. We learn w using a perceptron-style procedure (described in the next section).

Given the feature weights, inference is easy: we keep picking the action a with the highest score. If the action is HALT, we stop and return the current clustering. Otherwise, the action has the form JOIN_{ij} and we merge the clusters i and j. For the sake of efficiency, we maintain a priority queue containing all currently available actions and their current scores. At a state that has n clusters, the priority queue contains $O(n^2)$ actions.[2] When we pop JOIN_{ij} and merge i

[2]In Section 3.2 we describe a method for limiting the number of actions that we consider, so the true number of

Input: document d; weight vector w
Output: clustering C
$C =$ initial clustering with each mention in d in its own cluster ;
$A =$ empty priority queue ;
A.insert(HALT, $w \cdot \phi(\text{HALT}, C)$) ;
foreach *pair of mentions* i, j **in** d **do**
  A.insert(JOIN_{ij}, $w \cdot \phi(\text{JOIN}_{ij}, C)$) ;
end
repeat
  $a_{max} = A$.popMax() ;
  if a_{max} *has the form* JOIN_{ij} **then**
    performJoin(i, j, C, A) ; /* modifies C and A as described in text */
until $a_{max} ==$ HALT ;
return C ;

Algorithm 1: *Inference method: "easy-first" agglomerative clustering.*

and j, we update the queue accordingly, which involves removing the $O(n)$ actions that use the old i and j, and adding $O(n)$ new actions involving the new cluster ij. To quickly identify the actions to remove, each cluster stores pointers to the enqueued actions that involve it.

Additionally, each cluster stores a best-guess description of the *properties* of the entity to which the cluster refers. These properties include the *gender*, *animacy* and *number* of the entity, its *semantic type* (i.e., *person*, *organization*, etc.) as well as the set of *proper names* used to refer to the entity. When joining two clusters we consolidate the property values, which may be in conflict (i.e., the *number* of cluster i is *single*, while the *the number* of cluster j is *plural*). In principle, the cluster should store a probability distribution over the values of its properties, using this distribution both to score the compatibility with other clusters and to compute a consensus distribution when clusters are merged. Our clusters instead use a simpler approach of storing a single value for each property together with a confidence in that value. We employ manually assigned confidences consisting of three tiers. Pronouns induce the most confident properties (confidence=3), followed by proper names (confidence=2) and common nouns (confidence=1). When two clusters are merged, we resolve conflicting values of a property by choosing the more confident value, or in the case a tie, the value contributed by the larger cluster. In case of a further tie, we use the value of the cluster whose first mention is earlier in the document.

If n is the number of mentions in the document, then there are $\leq n$ clusters at any time. Furthermore the algorithm must HALT after at most n JOIN steps—and usually halts much sooner since it would be incorrect to merge all mentions into a single large cluster. Each step of the algorithm involves:

- popping the highest-scoring action JOIN_{ij} from the priority queue (runtime $O(\log n)$),
- performing the merge action (runtime $O(1)$ by having the new cluster ij point back to the old clusters i and j),
- computing the properties of the newly merged cluster (total runtime $O(1)$),
- deleting $O(n)$ old actions (total runtime $O(n \log n)$),
- computing the scores of $O(n)$ new actions (runtime $O(n)$),[3]
- and inserting these (runtime $O(n \log n)$ or $O(n)$ depending on the type of priority queue).

actions is $O(Cn)$, where C is a constant.

[3]One might think that computing the score of each new JOIN_{ij} action would be expensive (because one might have

Thus, the total runtime is $O(n^2 \log n) + O(sn \log n)$, where $s \leq n$ is the number of actions popped before HALT and tends to be small. The initial $O(n^2 \log n)$ term is for initializing the priority queue by pushing all actions JOIN_{ij}. We will speed this up below by pruning the set of actions.

Note that this easy-first algorithm essentially has the same structure as Kruskal's (1956) minimum spanning tree algorithm on a complete graph of $O(n^2)$ edges, except that as the connected components grow and merge, the weights of the edges between them are recomputed. In addition, the easy-first algorithm may HALT early, outputting a forest of several components rather than a single spanning tree. Many agglomerative clustering algorithms have this structure. The key difference here is that we are going to *learn* a linear function that weights the old and new edges—we will learn to define "easiness" such that the easy-first algorithm will achieve good scores on a coreference task.

3.2 Pair Selection

In the name of efficiency, we apply several rules to limit the number of pairs of mentions that our algorithm considers. (That is, we run our Kruskal's-like algorithm on a sparse graph rather than a complete graph with $O(n^2)$ edges.) For each mention, we only consider some of the preceding mentions as possible antecedents. The number of preceding antecedents we consider is based on the type of each mention and relies on linguistic intuitions. The following describes the possible antecedents that we consider for each mention type.[4]

- **Proper Name (Named Entity):** A proper name is compared against all proper names in the current sentence and the 40 preceding sentences. In addition, it is compared to all other mentions in the 3 preceding sentences.
- **Definite noun phrase:** Compared to all mentions in the 4 preceding sentences.
- **Common noun phrase:** Compared to all mentions in the 2 preceding sentences.
- **Pronoun:** Compared to all mentions in the three preceding sentences unless a first person pronoun. First person pronouns are additionally compared to first person pronouns in the preceding 40 sentences.

These simple rules cover almost all positive pairwise links in our corpora while effectively reducing the runtime of our algorithm from $O(n^2 \log n)$ to $O(Cn \log n)$. Technically, C as defined above is not a constant since the k preceding sentences could be arbitrarily long and thus could contain up to $n-1$ mentions. So we set a limit of at most 500 preceding mentions that we consider. This limit is never achieved in our experiments.

3.3 Features

As mentioned before, an advantage of our approach over the pairwise model is that it is not "edge-factored." Each greedy decision can rely on all of the information contained in the coreference chains built so far. In particular, we can rely on features that attempt to capture agreement between entire clusters. We will refer to such features as *cluster features*. Our cluster features include features that fire when the two clusters have the *same* gender, animacy, number,

to score all pairs of mentions in the clusters being joined). However, we explain in the next section how to do it in $O(1)$ time for the particular set of features that we use.

[4]The numbers that appear in our pair selection heuristics could, of course, be automatically tuned on development data to achieve a user's desired balance between speed and accuracy on a particular domain. We do not do this in the present paper, which focuses on a different kind of automatic tuning.

head noun or semantic class. Similarly, we have features that fire when the two clusters have the *same* or *compatible* gender, animacy, number, head noun or semantic class (i.e., one cluster has *gender* value *feminine* while the other has value *unknown*). We also have features that are *true* when the set of heads of the two clusters overlap or the set of proper names overlap. Another highly effective feature that we use detects when there is a conflict between person names—for instance, this feature will fire if we try to match *John Smith* and *Jane Smith*.

Additionally, each cluster stores a bit that indicates whether the cluster's first mention is a pronoun (where "first" refers to the order in which the mentions appear in the text). If it is, additional features indicate whether the other cluster is a potential antecedent for the pronoun (i.e., whether it agrees in gender, number, and/or animacy and has a non-conflicting semantic type and has at least one mention that appears earlier). Additional features indicate whether the noun phrase is the only such potential antecedent in the sentence in which the pronoun appears; the only potential antecedent in the preceding sentence; or the only potential antecedent that is a verbal subject in the preceding three sentences. This special handling is necessary to assure pronouns to get a single antecedent.

In addition to the cluster features, our system utilizes the rich set of features developed for mention-pair systems that capture local configurations indicative of coreference. These *local features* operate on pairs of mentions. We include features that capture when the two mentions are in an appositive relation, a predicate-nominal relation or are aliases of each other (see Ng and Cardie (2002) and Raghunathan et al. (2010) for descriptions of these features). We also introduce in the local feature set some features that capture the textual overlap of two mentions. We use most of the local features utilized by Stoyanov et al. (2009), with the exception of the ones that duplicate our cluster features. Details about the local features that we use can be found in Stoyanov et al. (2009) and Raghunathan et al. (2010). As discussed later in the paper, the code for our system is available upon request; it documents both the feature names as they relate to the corresponding papers, as well as their semantics.

Again, each local feature evaluates a *pair of mentions*. However, what we are evaluating is a JOIN_{ij} action that merges a *pair of clusters*. Thus, when we consider joining two clusters i and j at least one of which contains more than one mention, we have a choice of all local vectors between mentions in i and j to score the local compatibility. After some experimentation on development data, we decided to select the local cluster vector which results in the highest overall local score. In other words, we pick the single local vector that scores highest under w. This resembles single-link clustering. For example, to create a long chain of mentions that runs through the text, it would suffice for each pair of *adjacent* mentions (for example) to be locally compatible (e.g., syntactically related or proximate). Even though other mention pairs in the chain might not receive a high pairwise score (because they are not *directly* related), these lower pairwise scores would not be able to veto the chain. Thus, we effectively have a division of labor where positive evidence is contributed mainly by local pairwise features and negative evidence is given only by cluster features that detect global incompatibilities.

At test time, maintaining the local feature vectors takes time $O(1)$ since both the local features and the weight vector w are fixed. Thus, we can maintain a single local feature vector for each action. When joining two clusters i and j into ij, for each k, the local feature vector of the new action $\text{JOIN}_{(ij)k}$ becomes the maximum of the local feature vectors of actions JOIN_{ik} and JOIN_{jk}. At training time, w is not fixed, so we need to maintain local vectors for all pairs of mentions and recompute the maximum every time w changes.

3.4 Learning

The goal of training is to find a weight vector w that leads our easy-first algorithm to good clusterings. Goodness is defined in terms of a task-specific evaluation function (we consider MUC score and the BCubed evaluation metrics described in Section 4). We face a typical reinforcement learning problem in that in effect, our training set depends on our weight vector w—if we choose our actions differently or choose a different input document, we end up in previously unvisited states, which offer new sets of actions to choose among. During training we need to act in the environment to generate states that present the kinds of choices among actions that we are likely to encounter during testing. We cannot practically enumerate every possible state even for a single training example, because there are exponentially many clusterings; and of course test examples will give rise to entirely new states.

In early experiments we attempted using standard reinforcement learning algorithms such as Q-learning (Watkins and Dayan, 1992) and policy gradient (Sutton et al., 2000), but the learned weights did not lead to accurate predictions. We believe that this is due to the large variation of the values of our actions. Thus, we settled on a variant of the structured perceptron with early updates (Collins and Roark, 2004) with beam width 1. Our algorithm was inspired by the successful use of a similar algorithm by Goldberg and Elhadad (2010) for training an easy-first dependency parser. The nature of our task necessitates certain differences from both the structured perceptron and the "easy-first" algorithm of Goldberg and Elhadad. Those differences are described bellow.

Algorithm 2 is our training algorithm. Overall, the algorithm monitors the agent's behavior, and performs perceptron-style updates when the current weights would result in a mistake. Learning starts from a vector of initial weights $w^{(0)}$ (initialization is discussed below) and iterates over all training documents. For each document d training begins at the start state, where each mention is in its own cluster (we use C to refer to the current set of clusters and A to refer to the current set of available actions). A state s is specified by C and A. At each time step, t, the algorithm finds the highest weighted action a_{max} according to the current weights $w^{(t)}$. If a_{max} is positive (we say that an action is positive if it results in an immediate increase of the evaluation metric for which we train), w is not updated. If a_{max} is negative (i.e., results in an immediate decrease in evaluation score), the algorithm performs a perceptron update by subtracting the features of a_{max} and adding the features of the highest scoring positive action a_{pos} scaled by a learning rate η: $w \mathrel{+}= \eta \cdot (\phi(a_{pos}, C) - \phi(a_{max}, C))$. We repeat this update k times or until a_{max} is a positive action. Note that the highest-scoring positive action a_{pos} may be different in each iteration as the feature weights are changing. Once the weights have been updated, the algorithm selects the (new) highest-scoring action a_{max} (which may be positive or negative at this point) and performs it. The procedure is repeated in the new state, and so on. When the selected action is the HALT action, we stop clustering the document and move on to the start state for the next document. After iterating over the corpus a specified number of times, the learner returns the average of all the weight vectors that were actually used to choose actions.

The above algorithm learns a weight for the HALT action. Nevertheless, in an additional step after running Algorithm 1, we fine-tune the weight of the single identity feature associated with the HALT action. We are essentially learning a *threshold* on when to stop resolution. Stopping earlier will improve precision at the expense of recall, so the ideal time to stop depends on our evaluation metric. We freeze all other feature values and pick the weight of the HALT feature

Input: set D of coreference-annotated documents; initial weight vector w; learning rate η;
number of iterations *numIter*; max number of updates per error, k
Output: new weight vector w
$w_{sum} = \vec{0}$; $wcount = 0$;
for $i = 1$ **to** *numIter* **do**
  foreach *document* $d \in D$ **do**
    initialize C and A from d as in the first 5 lines of Algorithm 1 ;
    repeat
      for *tries* $= 1$ **to** k **do**
        $a_{max} = A$.peekMax() ; /* highest-scoring action */
        if a_{max}.isPositive() **then break** ; /* proceed with it if it's positive */
        $a_{pos} = A$.peekMaxPositive() ; /* highest-scoring pos action (or Halt if none) */
        $w \mathrel{+}= \eta \cdot (\phi(a_{pos}, C) - \phi(a_{max}, C))$; /* update toward a_{pos} */
        recompute scores in A using new w ;
      end
      $w_{sum} \mathrel{+}= w$; $wcount$++;
      $a_{max} = A$.popMax() ;
      if a_{max} *has the form* JOIN_{ij} **then**
        performJoin(i, j, C, A) ;
    until $a_{max} ==$ HALT ;
  end
end
return $w_{sum}/wcount$;

Algorithm 2: *A training method for easy-first clustering.*

that maximizes the desired evaluation metric on the training data. Experiments on development data showed that this step results in rather small but consistent improvements in our domain.

As noted above, our algorithm is a slight modification of the structured perceptron with early updates (Collins and Roark, 2004) for beam width 1. The difference is that upon making a mistake on a training example (in our case a training example is a document), the structured perceptron updates the weights and moves onto the next example. In contrast, we update the weights and continue working on the current example. Our subsequent updates on the example may thus have to update from one incorrect clustering to another—unlike the structured perceptron or violation-fixing perceptrons in general (Huang et al., 2012) (but like DAgger (Ross et al., 2011)). The difference is motivated by the fact that our data is highly non-separable—it is extremely rare to get all of the coreference decisions right in a document. Thus, the algorithm needs to operate in states in which some errors were previously made. Additionally, there is a significant overhead in initializing the necessary data structures for each document. Thus, a straightforward implementation of the structured perceptron with early updates is inefficient.

Goldberg and Elhadad (2010) use an update strategy similar to the structured perceptron, but keep updating until the next move will not result in an error ($k = \infty$).[5] For the reasons discussed above, such an update is not suitable for our problem—we want the algorithm to learn to operate as well as possible in states that contain errors, as such states will inevitably be encountered during testing.

[5] Because it was unclear to them at the time how to supervise the parser beyond its first error (Goldberg, p.c.).

Features from Stoyanov et al. (2009)
ProComp, SoonStr, Modifier, PostModifier, WordsSubstr, Pronoun1, Pronoun2, Definite1, Definite2, Demonstrative2, Embedded1, Embedded2, InQuote1, InQuote2, BothProperNouns, BothEmbedded, BothInQuotes, BothPronouns, BothSubjects, Subject1, Subject2, Appositive, RoleAppositive, MaximalNP, SentNum0, SentNum1, SentNum2, SentNum3, SentNum4plus, ParNum0, ParNum1, ParNum2plus, Alias, Acronym, IAntes, WeAntes, BothYou, Span, Binding, Contraindices, Syntax, ClosestComp, Indefinite, Indefinite1, Prednom, Pronoun, ProperNoun , WordNetClass, WordNetDist, WordNetSense, Subclass, AlwaysCompatible, WNSynonyms, Quantity, PairType
Features from Raghunathan et al. (2010)
IwithinI, Demonyms, ModifierHeadMatch, WhoResolve, WhichResolve
Novel features:
DeterminerHeadMatch, Longer2, LongerPN2, ShorterPN2, Halt

Table 1: Names of the local features used. Global features are described at start of section 3.3.

In the experiments described in the next section, we use for all runs $\eta = .01$ and $numIter = 5$ iterations of training and a single update to the training vector ($k = 1$). These parameters were tuned on additional development data (namely, the corpus described in Hasler et al. (2006)).

For our baseline (described in more detail in the next section) we train a "flat" classifier that implements the traditional mention-pair model. The baseline does not maintain clusters incrementally, but instead, it makes pairwise decisions up front and forms clusters by transitive closure. As in the easy-first case, we utilize a linear classifier trained using the perceptron algorithm on the set of initial clusters C. Weights are trained to recognize whether two clusters are coreferent or not. This training regime is equivalent to using easy-first training on the initial state without ever performing any action, but iterating over all possible actions and updating when a positive action would be classified as negative and vice versa. Our easy-first learning algorithm is initialized with the weights obtained from a single iteration of baseline training.

4 Experiments

We implemented our algorithm in Reconcile, a research platform for coreference resolution (Stoyanov et al., 2010b,a). We used the same set of preprocessing components as Stoyanov et al. (2009) and took a subset of their features for our local features. The names of the features that we use are listed in Table 1 and are documented in our code. The code of our system is available upon request. We can also provide a trained version of our easy-first algorithm that can be used as a resolver.

4.1 Data

We experiment with five of the most commonly used coreference resolution data sets. Those include two corpora from the MUC conferences (MUC-6, 1995; MUC-7, 1997) and three from the Automatic Content Evaluation (ACE) Program (NIST, 2004). We use the training and test splits previously used by Stoyanov et al. (2009).

We first evaluate an end-to-end coreference system that automatically discovers mentions. We measure system performance using two common scoring algorithms—MUC (Vilain et al., 1995) and BCubed (Bagga and Baldwin, 1998). When working with automatically extracted mentions, rather than gold-standard mentions, we use the BCubed variant proposed by Stoyanov et al. (2009), which they label BCubed_{all}.

We also compare to two unsupervised systems that have claimed the best scores on recent evaluations: the systems of Haghighi and Klein (2010) and Raghunathan et al. (2010). In order to compare to those, we also train and test a version of our system that uses gold-standard mentions on the MUC6 and ACE04 corpora. For training on the MUC6 corpus we follow the standard training/test split and report results on the test part. Unsupervised systems (Haghighi and Klein, 2010; Raghunathan et al., 2010) test on the entire newswire portion of the ACE04 corpus. In order to test the supervised methods on the same corpus, we randomly split it in two, and test on each half when training on only the other half (two-fold cross validation).

4.2 Baselines

We compare to a baseline that trains a pairwise classifier and then performs transitive closure. This setting is similar to the state-of-the-art systems of Bengtson and Roth (2008) and Stoyanov et al. (2009). The baseline system uses the same linear scoring function and the same feature set as our easy-first system, including global features. However, it only scores the pairwise joins of the initial single-mention clusters. It then simply performs transitive closure on the pairs whose scores are above the halting threshold, rather than computing new join scores during agglomerative clustering. As noted previously, the baseline is trained using the standard averaged perceptron (Freund and Schapire, 1999) and the positive decision threshold is tuned on training data as in the easy-first system.

Our pairwise baseline uses the same data splits and pre-processing components as Stoyanov et al. (2009). Thus, our results are directly comparable. In fact, our baseline can be considered a re-implementation of Stoyanov et al. (2009), except for the feature set. Our local features are a subset of the features used by Stoyanov et al. (2009), but we have also added some of the features described by Raghunathan et al. (2010) (as discussed in Section 4.3). We will use *Pairwise* to refer to this baseline in the results.

Our *Easy-first*$_{percep}$ setting runs our easy-first inference algorithm (Algorithm 1) but using the same perceptron-trained weights as the baseline system while still utilizing the global features. Finally, *Easy-first*$_{struct}$ also runs easy-first inference, but after training with the full algorithm described in Section 3.4 and Algorithm 2.

4.3 Results

Results of our fully automatic, end-to-end coreference resolution system are shown in Table 2. Results using gold standard mentions are shown in Table 3.

Comparison to Stoyanov et al. (2009). As previously mentioned, our *Pairwise* baseline (the second result line in Table 2) is a reimplementation of Stoyanov et al. (2009) (the first result line) with a different feature set. The results are quite different—*Pairwise* performs slightly worse than Stoyanov et al. on the MUC6 corpus and substantially worse on MUC7, while it shows substantial gains across the three ACE corpora. For instance, we gain 6 points of MUC F1-score on the ACE04 corpus. We attribute these differences to the different features and resources that we used. The main differences are outlined below.

First, *Pairwise* drops several of the features used by Stoyanov et al. (2009). Most notably, we drop the *RuleResolve* and *ProResolve* features that they utilize. Those features rely on an internal rule-based coreference and anaphora resolution systems, respectively, and indicate whether the respective rule based system clustered the noun-phrases together. We chose not to utilize these

	MUC6		MUC7		ACE03		ACE04		ACE05	
	B^3	*MUC*	B^3	*MUC*	B^3	*MUC*	B^3	*MUC*	B^3	*MUC*
Stoyanov et al.	70.9	**68.5**	**65.9**	**62.8**	79.4	67.9	76.5	62.0	73.7	67.4
Pairwise baseline	70.7	67.9	62.2	57.9	81.7	73.5	79.4	68.0	75.7	70.2
Easy-First$_{percep}$	71.5	68.4	64.4	58.1	82.8	74.1	80.3	68.4	**76.0**	70.0
Easy-First$_{struct}$	**72.1**	68.2	64.6	57.7	**83.1**	**74.5**	**80.9**	**68.8**	75.9	**70.9**

Table 2: Results for "end-to-end" coreference resolution, where mentions are automatically extracted. The best result in each column is boldfaced.

	MUC6		ACE04	
	B^3	*MUC*	B^3	*MUC*
Haghighi and Klein	75.0	81.9	76.9	76.5
Raghunathan et al.	73.2	77.7	78.9	78.1
Stoyanov et al.	76.1	**88.2***	*77.8*	*76.2*
Pairwise baseline	72.7	**88.2***	75.9	78.9
Easy-First$_{percep}$	73.3	**88.2***	78.4	**79.8**
Easy-First$_{struct}$	**77.5**	**88.2***	**81.8**	**80.1**

Table 3: Results with gold-standard mentions. In each column, the best result is boldfaced along with all statistically indistinguishable results (paired permutation test, $p < .01$). A star indicates that the result was achieved by a degenerate clustering—the algorithm learns a threshold that groups all mentions in a single cluster.[5] Italics indicate that result are not comparable to the rest, because Stoyanov et al. (2009) use a different test/train split and evaluate on only a subset of the ACE04 documents. Note also that the first two systems are unsupervised.

features because the rule-based systems are computationally expensive. They also appear to be engineered with significant amounts of knowledge and seem to be targeted toward the MUC corpora. We confirmed empirically that the decrease of performance on the MUC7 corpus is due to the absence of these features.

Second, as noted earlier, *Pairwise* introduces several new features inspired by Raghunathan et al. (2010). Below we list the most useful new features as shown through ablation studies:

- **Demonyms** – this feature is true if one of the two mentions is a demonym of the other, e.g. *Holland* and *Dutch*. We utilize the list of demonyms from Raghunathan et al. (2010).
- **WhoResolve** and **WhichResolve** – true for mentions that are animate/inanimate relative pronouns and the preceding mention in the text. This features can help resolve construct such as "*Smith*, *who* was present . . . ".
- **CountryCapital** – true if one mention is a country and the other is the capital of that country. This feature is useful since capitals are often used to refer to the country governments, e.g. "*London* issued a statement." This feature is novel to this work. We used a list of country capitals that we mined from Wikipedia.

Third, following Raghunathan et al. (2010), we use the list of name/gender associations provided by Bergsma and Lin (2006). This list is more comprehensive and more precise than the list that comes with the Reconcile system.

[5] Previous work (Stoyanov et al., 2009) has pointed out that because MUC annotates only anaphoric mentions, such a baseline results in a surprisingly strong performance when evaluating on MUC data with gold-standard mentions and using the MUC score.

Effectiveness of the Easy-First approach. Table 2 shows that easy-first inference, when trained using either our full algorithm or the perceptron algorithm, performs better than the pairwise baseline in 9 out of the 10 end-to-end evaluations. This suggests that it may also work better in other evaluations.[6] In general, the advantage of easy-first is stronger on the more precision-conscious BCubed score, which can be expected since easy-first mainly aims to correct precision errors (see section 4.4 for examples).

The improvements are more pronounced when we evaluate our system on gold-standard mentions as shown in Table 3. Compared to the unsupervised systems of Haghighi and Klein (2010) and Raghunathan et al. (2010), our system shows improvements of at least 2 points on all evaluation metrics on both datasets. When evaluated on BCubed score, easy-first outperforms the pairwise baseline by 5–6 points on both corpora, and Algorithm 2 now helps substantially.

For the gold-standard experiments, we test the statistical significance of these measured improvements, treating each test document as an independent observation. A paired permutation test ($p < .01$) reveals that improvements of Easy-first$_{struct}$ over Raghunathan et al. (2010) are statistically significant in all four settings. Additionally, Easy-first$_{struct}$ improvements over the pairwise baseline are statistically significant in the two settings that use BCubed score for evaluation. For the two settings using MUC score, the pairwise baselines as well as our methods find a degenerate solution that works well for that evaluation measure—the algorithm learns a threshold that groups all mentions in a single cluster.

To the best of our knowledge, the results presented in Table 3 represent a new state-of-the-art for systems using gold-standard mentions. For end-to-end coreference systems, our results in Table 2 represent a new state-of-the-art for the three ACE corpora.

4.4 Qualitative Evaluation

The cold hard numbers behind our evaluation tell only half of the story. Looking at the output of the end-to-end system reveals that the easy-first system produces results that look more consistent with human expectations. Consider, for instance the following sentence from an ACE05 document: *"The court order was requested by Jack Welch's attorney, Daniel K. Webb, who said Welch would likely be asked about his business dealings, his health and entries in his personal diary."*

The pairwise baseline is confused by the evidence that suggests that *his* should be linked to both *Welch* and *Daniel K. Webb* and incorrectly assigns all eight of the underlined mentions to the same cluster. Our easy-first approach, in contrast, does not attempt to join clusters that contain conflicting person names. Thus, it correctly separates {*Jack Welch's attorney, Daniel K. Webb* and *who*} into one cluster and all other mentions into another cluster.

Another example from the ACE04 corpus concerns resolving the following sentence: *"Tamara Maschino, for example, a resident of the Clear Lake area of Houston, criticizes Bush for his lack of attention to pollution problems from chemical plants near her home."*

Here the pairwise baseline resolves the pronoun *her* to *Bush*, grouping together {*George Bush* (an earlier occurrence), *Bush*, *his*, *her*}. The easy-first system, in contrast, first links the mention *Bush* to *George Bush*, and then correctly links *her* not to this male cluster but to *Tamara Maschino*.

[6] Indeed, 9 improvements out of 10 would be significant under a sign test ($p < 0.025$), if the 10 settings were independent. They are not, though, since the BCubed and MUC scores on a given corpus are presumably correlated.

5 Related Work

Related work on coreference resolution is discussed throughout the paper. As previously mentioned, our learning algorithm follows the easy-first approach to dependency parsing proposed by Goldberg and Elhadad (2010). Our learning method is also inspired by the structured perceptron and its application to incremental parsing (Collins and Roark, 2004). Our algorithm is an instance of learning parameters for a fixed approximate inference method (i.e., greedy search). Other approaches that learn for a fixed greedy inference method include SEARN (Daumé et al., 2009) and LaSO (Daumé III and Marcu, 2005). Other work proposes methods for learning parameters for fixed approximate methods beyond greedy search. Stoyanov et al. (2011) propose a back-propagation-based algorithm for learning the weights of Markov Random Fields and Conditional Random Fields for a fixed variational inference algorithm (they experiment with loopy belief propagation). Experiments on three NLP problems show that this regime of learning leads to more accurate system performance than traditional approximate maximum-likelihood training (Stoyanov and Eisner, 2012).

Concurrent to us, Jain et al. (2011) propose a method for learning features to guide a greedy agglomerative clustering algorithm. They apply their algorithm to a very different problem – image segmentation by clustering superpixels. Their learning algorithm is inspired by on-policy reinforcement learning and it differs from ours. Initial experiments with several reinforcement learning approaches on our problem (including Q-learning (Watkins and Dayan, 1992) and policy gradient (Sutton et al., 2000)) did not lead to improvements in our setting.

6 Conclusions and Future Work

We presented a novel algorithm for coreference resolution that capitalizes on the idea that early "easier" decisions can be used to guide later, "harder" decisions. We presented a training algorithm based on the structured perceptron for automatically learning feature weights. When evaluated on coreference data, our algorithm outperformed a competitive baseline as well as previously published state-of-the-art methods under most evaluation conditions.

Our model is still missing the opportunity to compare each action to alternatives. For instance, the algorithm should be able to learn that it is "easy" to join clusters i and j when i is the only reasonable antecedent cluster for j (there are no strong competitors). For instance, if j is headed by a pronoun and cluster i contains the only reasonable antecedent for J, the algorithm can have more confidence that i and j should be joined. Thus, it would learn to perform such actions sooner. Since actions can rely on any features of the current state, we could augment our model with such competition-style features, as used by Yang et al. (2003).[7]

Our easy-first approach is suitable for performing coreference resolution jointly with other tasks. We are particularly interested in jointly performing within-document coreference and cross-document coreference (or entity linking). An easy-first approach will allow us to trade-off decisions on the document and intra-document level. For example, linking *Obama* to the appropriate database entity may help us link it to the mention *president* later in the document.

Acknowledgments

Thanks to the anonymous reviewers, and to Yoav Goldberg, for their thoughtful comments on an earlier version. Also thanks to the JHU NLP group and especially Shane Bergsma for early

[7] We do already use competition-style features to evaluate pronoun antecedents, as described in section 3.3.

discussions of the approach. This work was supported by the the Human Language Technology Center of Excellence at Johns Hopkins University, and by National Science Foundation under Grant No. 0964681 to the second author.

References

Bagga, A. and Baldwin, B. (1998). Algorithms for scoring coreference chains. In *Linguistic Coreference Workshop at LREC 1998*.

Bengtson, E. and Roth, D. (2008). Understanding the value of features for coreference resolution. In *Proceedings of EMNLP*, pages 294–303.

Bergsma, S. and Lin, D. (2006). Bootstrapping path-based pronoun resolution. In *Proc. of ACL*.

Collins, M. and Roark, B. (2004). Incremental parsing with the perceptron algorithm. In *Proceedings of ACL*.

Culotta, A., Wick, M., Hall, R., and McCallum, A. (2007). First-order probabilistic models for coreference resolution. In *Proceedings of HLT/NAACL*, pages 81–88.

Daumé, H., Langford, J., and Marcu, D. (2009). Search-based structured prediction. *Machine learning*, 75(3):297–325.

Daumé III, H. and Marcu, D. (2005). Learning as search optimization: Approximate large margin methods for structured prediction. In *Proceedings of ICML*, pages 169–176.

Freund, Y. and Schapire, R. (1999). Large margin classification using the perceptron algorithm. *Machine learning*, 37(3):277–296.

Goldberg, Y. and Elhadad, M. (2010). An efficient algorithm for easy-first non-directional dependency parsing. In *Proceedings of NAACL*, pages 742–750. Association for Computational Linguistics.

Haghighi, A. and Klein, D. (2010). Coreference resolution in a modular, entity-centered model. In *Proceedings of NAACL*, pages 385–393.

Hasler, L., Orăsan, C., and Naumann, K. (2006). NPs for events: Experiments in coreference annotation. In *Proceedings of LREC2006*.

Huang, L., Fayong, S., and Guo, Y. (2012). Structured perceptron with inexact search. In *Proceedings of NAACL*.

Jain, V., Turaga, S., Briggman, K., Helmstaedter, M., Denk, W., and Seung, H. (2011). Learning to agglomerate superpixel hierarchies. In *Proceedings of NIPS*.

Kruskal, J. (1956). On the shortest spanning subtree of a graph and the traveling salesman problem. *Proceedings of the American Mathematical society*, 7(1):48–50.

MUC-6 (1995). Coreference task definition. In *Proceedings of the Sixth Message Understanding Conference (MUC-6)*, pages 335–344.

MUC-7 (1997). Coreference task definition. In *Proceedings of the Seventh Message Understanding Conference (MUC-7)*.

Ng, V. and Cardie, C. (2002). Improving machine learning approaches to coreference resolution. In *Proceedings of ACL*, pages 104–111.

NIST (2004). *The ACE Evaluation Plan*. National Institute of Standards and Technology, United States of America.

Poon, H. and Domingos, P. (2008). Joint unsupervised coreference resolution with Markov logic. In *Proceedings of EMNLP*, pages 650–659.

Pradhan, S., Ramshaw, L., Marcus, M., Palmer, M., Weischedel, R., and Xue, N. (2011). Conll-2011 shared task: Modeling unrestricted coreference in ontonotes. In *Proceedings of CoNLL 2011*, pages 1–27.

Raghunathan, K., Lee, H., Rangarajan, S., Chambers, N., Surdeanu, M., Jurafsky, D., and Manning, C. (2010). A multi-pass sieve for coreference resolution. In *Proceedings of EMNLP*, pages 492–501.

Ross, S., Gordon, G. J., and Bagnell, J. A. (2011). A reduction of imitation learning and structured prediction to no-regret online learning. In *Proceedings of AISTATS*.

Soon, W., Ng, H., and Lim, D. (2001). A machine learning approach to coreference resolution of noun phrases. *Computational linguistics*, 27(4):521–544.

Stoyanov, V., Cardie, C., Gilbert, N., Riloff, E., Buttler, D., and Hysom, D. (2010a). Coreference resolution with Reconcile. In *Proceedings of the ACL Short Papers*, pages 156–161.

Stoyanov, V., Cardie, C., Gilbert, N., Riloff, E., Buttler, D., and Hysom, D. (2010b). Reconcile: A coreference resolution research platform. Technical report, Cornell University.

Stoyanov, V. and Eisner, J. (2012). Minimum-risk training of approximate CRF-based NLP systems. In *Proceedings of NAACL*.

Stoyanov, V., Gilbert, N., Cardie, C., and Riloff, E. (2009). Conundrums in noun phrase coreference resolution: Making sense of the state-of-the-art. In *Proceedings of ACL/IJCNLP*, pages 656–664.

Stoyanov, V., Ropson, A., and Eisner, J. (2011). Empirical risk minimization of graphical model parameters given approximate inference, decoding, and model structure. In *Proceedings of AISTATS*.

Sutton, R., McAllester, D., Singh, S., and Mansour, Y. (2000). Policy gradient methods for reinforcement learning with function approximation. *NIPS*.

Vilain, M., Burger, J., Aberdeen, J., Connolly, D., and Hirschman, L. (1995). A model-theoretic coreference scoring theme. In *Proceedings of the Sixth Message Understanding Conference (MUC-6)*.

Watkins, C. and Dayan, P. (1992). Q-learning. *Machine Learning*, 8(3):279–292.

Yang, X., Zhou, G., Su, J., and Tan, C. (2003). Coreference resolution using competition learning approach. In *Proceedings of ACL*, pages 176–183.

Zhou, G. and Su, J. (2004). A high-performance coreference resolution system using a constraint-based multi-agent strategy. In *Proceedings of COLING*.

Modeling Leadership and Influence in Multi-party Online Discourse

Tomek STRZALKOWSKI[1,2], Samira SHAIKH[1], Ting LIU[1], George Aaron BROADWELL[1], Jenny STROMER-GALLEY[1], Sarah TAYLOR[3], Veena RAVISHANKAR[1], Umit BOZ[1] and Xiaoai REN[1]

(1) State University of New York – University at Albany, NY 12222 USA
(2) Polish Academy of Sciences
(3) Sarah M. Taylor Consulting, LLC
tomek@albany.edu, samirashaikh@gmail.com

ABSTRACT

In this article, we present a novel approach towards the detection and modeling of complex social phenomena in multi-party discourse, including leadership, influence, pursuit of power and group cohesion. We have developed a two-tier approach that relies on observable and computable linguistic features of conversational text to make predictions about sociolinguistic behaviors such as Topic Control and Disagreement, that speakers deploy in order to achieve and maintain certain positions and roles in a group. These sociolinguistic behaviors are then used to infer higher-level social phenomena such as Leadership and Influence, which is the focus of this paper. We show robust performance results by comparing our automatically computed results to participants' own perceptions and rankings. We use weights learnt from correlations with training examples known leadership and influence rankings of participants to optimize our models and to show performance significantly above baseline for two different languages – English and Mandarin Chinese.

KEYWORDS : computational sociolinguistics, online dialogues, social phenomena, linguistic behavior, influence, multi-disciplinary artificial intelligence, social computing

1 Introduction and Related Work

Our objective is to model high-level sociolinguistic phenomena such as Leadership, Influence, Pursuit of Power and Group Cohesion in discourse. This research project aims to develop a computational approach that uses linguistic features of conversational text to detect and model sociolinguistic behaviors of conversation participants in small group discussions. Given a representative dialogue of multi-party conversation, our prototype system automatically classifies the participants by the degree to which they engage in such sociolinguistic behaviors as Topic Control, Task Control, Disagreement, and several others discussed in this paper. These mid-level sociolinguistic behaviors are deployed by discourse participants in order to assert higher-level social roles such as Leadership. Our approach to this problem combines robust computational linguistics methods and established empirical social science techniques. The focus in this paper is on online multi-party conversations in chat rooms; however, the models we are developing are intended to be universal and are applicable to other conversational situations: informal face-to-face interactions, formal meetings, moderated discussions, asynchronous threaded discussions as well as interactions conducted in languages other than English, e.g., Urdu and Mandarin. We shall discuss the robust detection of Leadership and Influence in discourse in this paper; we defer the discussion of remaining phenomena to a separate, larger publication.

Proceedings of COLING 2012: Technical Papers, pages 2535–2552,
COLING 2012, Mumbai, December 2012.

Social science theory indicates that leadership may be manifested in various ways (Bradford, 1978, Huffaker, 2010). We define leadership in the following terms: A leader is someone who guides group toward an outcome, controls group discussion, manages actions of the group and whom members recognize as the task leader. Such a leader is a skilled *task* leader, which corresponds to the social science theory put forth in Beebe and Masterson (2006). On the other hand, a *thought* leader in the group is someone who has credibility in the group and introduces ideas or thoughts that others pick up on or support. Such a person is a participant but need not be active in the portion of discussion where others credit or support him. This definition corresponds to the Initiator-Contributor type of leadership outlined in Bradford (1978). For ease of presentation and understanding – we shall refer to task or skilled leadership as *Leadership* and thought leadership as *Influence*, henceforth in this paper.

Since leadership and influence are manifested differently and may be deployed by distinct participants in a discussion, it is important for an automatic system to recognize the distinction and make a determination of who is deploying such roles. Consider as an example, a debate with panel of experts hosted by a facilitator. Here, the facilitator will exhibit sociolinguistic behavior consistent with being a task leader, by controlling the agenda, putting forth questions to individual panelists, beginning and ending the discussion and so on. However, she will not be a thought leader, or influencer, as she does not contribute much actual content to the discussion apart from asking questions. Any member of the expert panel may exhibit the sociolinguistic behavior consistent with being an influencer. In a peer-oriented group discussion however, it could occur that the task and thought leader (leader and influencer) are the same person.

Human-human interaction affords a rich resource for research. Much prior work has been done in communication that focuses on the communicative dimension of discourse. For example, the Speech Act theory (Austin, 1962; Searle 1969) provides a generalized framework of multiple levels of discourse analysis; work on dialogue analysis (Blaylock, 2002; Carberry and Lambert, 1999; Stolcke et al., 2000) focuses on information content and structure of dialogues. Somewhat more relevant to social roles is research that models sequences of dialogue acts (Bunt, 1994), in order to predict the next dialogue act (Samuel et al. 1998; Stolcke, et al., 2000; Ji & Bilmes, 2006, inter alia) or to map them onto subsequences or "dialogue games" (Carlson 1983; Levin et al., 1998), from which participants' functional roles in conversation (though not social roles) may be extrapolated (e.g., Linell, 1990; Poesio and Mikheev, 1998; Field et al., 2008). However, the effects of speech acts on social behaviors and roles of conversation participants have not been systematically studied. Research in anthropology and communication has concentrated on how certain social norms and behaviors may be reflected in language (e.g., Scollon and Scollon, 2001; Agar, 1994). But, there are few systematic studies in the current literature that explore the way in which language may be used to make predictions of social roles in groups where (a) these roles are not known a priori, or (b) these roles do not exist prior to the beginning of the discourse and only emerge through interaction.

Internet-enabled conversation is particularly interesting because in this reduced-cue environment, the only means of engaging in and conveying social behaviors is through written language. As such, studying online chat relies on the more explicit linguistic devices necessary to convey social and cultural nuances than is typical in face-to-face or telephonic conversations. The use of language by participants as a feature to determine interpersonal relations has been studied by Bracewell et al. (2011) who developed a learning framework to determine collegiality between discourse participants. Their approach, however, looks at singular instances of linguistic markers or single utterances rather than a sustained demonstration of sociolinguistic behavior over the

course of entire discourse. Freedman et al. (2011) have developed an approach that takes into account the entire discourse to detect behaviors such as persuasion; however their analysis is conducted on and models developed upon online discussion threads where the social phenomena of interest may be rare. By contrast, we build our models based on analysis of a data corpus of online chat discourse, where data collection experiments were specifically designed so that the resulting corpus may be rich in sociolinguistic phenomena.

Our research extends the work of Strzalkowski et al. (2010) and Broadwell et al. (2012), who first proposed the two-tiered approach to sociolinguistic modeling and have demonstrated that a subset of mid-level sociolinguistic behaviors may be accurately inferred by a combination of low-level language features. We have adopted their approach and extended it to modeling of leadership and influence. Furthermore, we enhanced the method by adding the evidence learnt from correlations of indices and measures to compute weights through which sociolinguistic behaviors may be combined appropriately to infer higher-level social phenomena.

In this paper, we describe our approach to model Leadership and Influence in online multi-party task-oriented chat dialogues. We show how our models were developed on evidence from online English and Mandarin chat dialogues. Performance on both languages is very encouraging.

2 Sociolinguistic Behaviors to Model Leadership and Influence

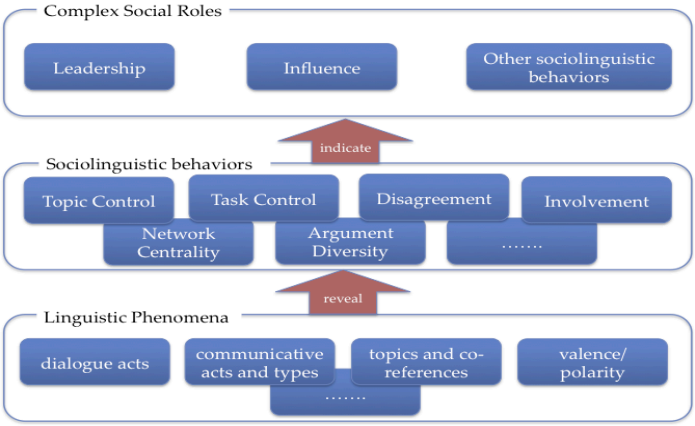


FIGURE 1 – Two-tier approach applied to model social roles in discourse.

In our two-tier approach, we use linguistic elements of discourse to first unravel sociolinguistic behaviors, and then, use the behaviors, in turn, to determine social roles, as shown in Figure 1. It is important to note that, at both levels, our analyses are solidly grounded on sociolinguistic theory. Mid-level behaviors that we shall discuss in this article are Topic Control, Task Control, Disagreement, Involvement, Argument Diversity and Network Centrality that are computed using *indices*. These indices are directly obtained from linguistic elements of discourse, which are described in Section 3. For each participant in the discourse, we compute the degree to which

they engage in sociolinguistic behaviors, using *measures,* which are a linear combination of indices. We describe behaviors, component indices and corresponding measures in this section.

2.1 Topic Control Measure (TCM)

Topic Control is defined as attempts of participants to impose a topic of conversation. This sociolinguistic behaviour is consistent with both Leadership and Influence. In any conversation, whether it is focused on a particular issue or task or is just a social conversation, the participants continuously introduce multiple topics and subtopics. These are called *local topics*. Local topics, following the notion put forth by Givon (1983), may be equated with any substantive noun phrases introduced into discourse that are subsequently mentioned again via repetitions, synonyms, or pronouns. Who introduces local topics, who continues to talk about them, and for how long are some of the indicators of topic control in dialogue. We have developed four indices for Topic Control. Participants who introduce more local topics exert more topic control in dialogue. The first index, called the **Local Topic Introductions Index (LTI)** calculates the proportion of local topics introduced by each participant, by counting the number of first mentions of local topics by each participant as percentage of all local topics in a discourse. The **Subsequent Mentions of Local Topics (SMT)** index calculates the percentage of discourse utterances where the local topics introduced by each participant are being mentioned (by themselves or others) through repetition, synonym, or pronoun. The **Cite Score (CS)** index calculates the percentage of subsequent mentions of local topics first introduced by each participant, but excluding the self-mentions by this participant. The final measure of topic control is the average **Turn Length (TL)** per participant. This index calculates the average utterance length (words) for each participant, relative to other participants.

We shall explain the calculation of one index – Local Topic Introductions (LTI) - in detail. Figure 2 shows a fragment of an actual chat conversation from our corpus with selected local topic references to illustrate how the constructed indices model the concept of Topic Control. In this small excerpt, a few local topics are introduced, including *Carla*, *nanny* and *horses*, as well as possibly others (*record, car,* etc). These local topics are underlined in different ways, with the first mention set in boldface. For example, *Carla* is introduced by speaker JR in turn 1, and is subsequently mentioned by KN (turn 2), LE and KN (via *she*) in turns 3 and 4. Similarly, KN introduces horses in turn 4, and then self-mentions it again in turn 6.

1. **JR**: *wanna go thru **Carlas** resume first ?*
2. **KN**: *i wonder how old carla is*
3. **LE**: *Ha, yeah, when I hear nanny I think she is someone older.*
4. **KN**: *she's got a perfect driving record and rides **horses**! coincidence?*
5. **JR**: *'06 high school grad*
6. **KN**: *i think she rides a horse and not a car!*

FIGURE 2 – Fragment of chat with a few selected local topics highligted.

Once we have computed the scores for each participant on each index, we combine them to compute a single score on the corresponding measure. In Figure 2, we only highlight a few local topics, to illustrate the process. Nouns such as *resume* and *high school*, are not marked for ease of presentation.

In this case, the LTI, SMT, CS and TL indices are combined to get a Topic Control Measure (TCM) for each participant. For instance, suppose we discover the following in a conversation with 7 participants and over 600 total turns:

1. *There are 90 distinct local topics introduced in the conversation.*
2. *The mean rate of local topic introduction is 14.29%.*
3. *Participant LE introduces 23 local topics. Thus, LTI index for LE is 23/90, that is, 23.6%.*
4. *Participant LE scores the highest amongst all on the LTI index in this conversation.*

Using the above information, we can assert that participant LE has the highest topic control in the conversation *based on the LTI index*. This is evidence based on just one index; we compute the index measures for all participants on all component indices to build additional evidence. In our current system prototype, TCM score is computed as the mean of component index scores.

2.2 Task (or Skilled) Control Measure (SCM)

Task Control is an effort by one or more members of a group to define the group's project or goal and/or steer the group towards it. Task Control is gained by telling others to perform certain tasks, or subtasks, or to accept certain decisions about the task. It can also be gained by the speaker offering to perform a task. This sociolinguistic behaviour is primarily consistent with Leadership. One index of Task Control is the number of directives (done as statements or questions) made by each participant as a percentage of all directives in discourse, known as **Directive Index (DI)**. In other words, a participant who tells others what to do (whether overtly or more subtly) is attempting to control the task that the group is performing. Other indices have been developed to support Task Control; these will be explained in future publication.

2.3 Cumulative Disagreement Measure (CDM)

Disagreement has a role to play with regard to leadership and influence in that it is possible that a person in a small group engages in disagreements with others in order to control the topic by way of identifying or correcting what they see as a problem (Ellis and Fisher, 1994; Sanders, Pomerantz and Stromer-Galley, 2010). While each utterance where a participant disagrees with another is a vivid of expression of disagreement, we are interested in a sustained phenomenon where participants repeatedly disagree, thus revealing a social relationship between them. One of the indices we have developed to measure disagreement is the proportion of disagree and/or reject turns produced by a participant that are directed at any other participants in the discourse. This index is called the **Disagree-Reject Index (DRI)**.

2.4 Involvement Measure (INVX)

Involvement is defined as a degree of engagement or participation in the discussion of a group. This behavior is consistent primarily with Leadership. A degree of involvement may be estimated by how much a speaker contributes to the discourse in terms of substantive content. Contributing substantive content to discourse includes introduction of new local topics, taking up the topics introduced by others, as well as taking sides on the topics being discussed. By topics here, we mean the local topics described previously. We have defined five indices in support of Involvement, we shall expand on three of them here. The **Noun Phrase Index (NPI)** is the amount of information content that each speaker contributes to discourse. The NPI measure is calculated by counting the number of content words (e.g., all occurrences of nouns and pronouns referring to people, objects, etc.) in each speaker's utterances as a percentage of all content words

in discourse. The **Turn Index (TI)** is the frequency of turns that different speakers have during a conversation. The **Topic Chain Index (TCI)** is computed by identifying the most frequently mentioned topics in a discourse, i.*e.,* topics chains (i.e., with gaps no longer than 10 turns and then by computing the percentages of mentions of these persistent topics by each participant.

2.5 Network Centrality Measure (NCM)

Another measure is the degree to which a participant is a "center hub" of the communication within the group. In other words, someone whom most others direct their comments to as well as whose topics are most widely cited by others. This behavior is consistent mainly with Influence. Two of the indices used to compute this measure are described. **Communication Links Measure (CLM)** index calculates a degree of Network Centrality for a participant by counting the utterances that are addressed to this participant as a percentage of all utterances in discourse. **Citation Rate Index (CRI)** calculates the degree of Network Centrality for a participant by counting the number of times that the local topics introduced by this participant are cited by other participants. Unlike the Subsequent Mentions measure (SMT) of Topic Control Measure, we calculate CRI by normalizing the citation count by the number of topics introduced by the participant, thus obtaining an average citation rate per topic.

2.6 (Measure of) Argument Diversity (MAD)

Argument Diversity, a behavior consistent with Influence, is displayed by the speakers who deploy a broader range of arguments in conversation. This behavior is signaled by the use of more varied vocabulary, including specialized terms and citations of authoritative sources, among others. A person who uses more varied vocabulary and introduces more unique words into a conversation is considered to have a higher degree of Argument Diversity, which can be measured using the two indices: the **Vocabulary Introduction Measure (VIM)** which is calculated as a proportion of new content words introduced by each participant to all distinct content words in discourse; and the **Vocabulary Range Index (VRI)** which is the number of distinct words used by this participant as a percentage of all distinct words in discourse.

2.7 Combining Indices and Measures

As outlined briefly at the end of Section 2.1, we compute the score of each *measure* by taking linear combination of scores obtained on each *index*. We can thus obtain a full ranking of participants on each sociolinguistic behavior. The measures used to compute Leadership are Topic Control, Task Control, Disagreement, and Involvement. These are indicated in Beebe and Masterson (2006) and borne out in our research. Measures used to compute Influence are Topic Control, Disagreement, Network Centrality and Argument Diversity. Since we have defined Influence as the Initiator-Contributor type of leadership (Bradford, 1978), we shall use those sociolinguistic behaviors that pertain to initiating discussion and contributing substantively in the group. On the other hand, Task Control and Involvement have little or minor role to play in computing Influence, and hence we do not include them while combining behaviors. Similarly, while Task Control and Disagreement are most indicative of Task Leadership, other behaviours such as Network Centrality and Argument Diversity do not correlate with this role. Hence, we do not include them in computation of Leadership. We shall elaborate on this in Section 5, Evaluation and Results.

3 Corpus, Annotation and Computational Modules

The models described in this paper are derived from online chat dialogues. The corpus we use for this analysis is the MPC chat corpus (Shaikh et al., 2010, Liu et al., 2012). This is a corpus of over 90 hours of online chat dialogues in English, Urdu and Mandarin. Participants in these chats are native speakers of these languages. Each chat session is a task-oriented dialogue around 90 minutes in length, with at least 4 participants. This corpus is particularly useful for the type of sociolinguistic analysis we are interested in due to the characteristics of interaction in each chat session – the participants are focused on some task, they form a fairly stable group and the dynamics of conversation unfold naturally through discourse. Other corpora exist such as the ICSI-MRDA corpus (Shriberg et al., 2004) and the AMI meeting corpus (Carletta, 2007), however these are spoken language resources rather than online chat. Where other corpora of online chat do exist, like the NPS Internet chat corpus (Forsyth and Martell, 2007) and StrikeCom corpus (Twitchell et al., 2004), they do not contain any information about the participants themselves or their reactions to the discussion. In order to create a ground truth of assessments of sociolinguistic behavior, we needed certain information to be captured through questionnaires or survey following each data collection session. In the data that comprise MPC corpus, at the conclusion of each chat session, participants were asked to fill out a survey consisting of a series of questions about their perceptions of and reactions to conversation that had freshly participated in. The questions were focused on eliciting responses about sociolinguistic behavior. Questions pertaining to Leadership and Influence are shown in Figure 3. Participants may interpret the notions of socio-linguistic phenomena intuitively, and may rank themselves and other participants accordingly. We refer the reader to the Conclusion section where we address this issue. There are similar questions regarding other behaviors we are interested in modeling and we refer the reader to the cited paper (Shaikh et al., 2010) for a detailed discussion of these.

- *During the discussion, some of the people talking are more influential than others. For the conversation you just took part in, please rate each of the participants in terms of how influential they seemed to you? Scale: Not Influential --- Very Influential.*
- *Below is a list of participants including yourself. Please rank order the participants with regards to leadership.*

FIGURE 3 – Questions regarding sociolinguistic phenomena in post-discussion survey.

We developed a multi-layer annotation process so that automatic modules may be trained to detect and classify social behaviors from discourse. A substantial subset of the MPC corpus was annotated using trained annotators who are native speakers of the respective language. Annotators were trained extensively so that inter-annotator agreement level was sufficiently high (0.8 or higher Krippendorf's alpha). We briefly explain three of the categories that annotation was performed on:

3.1 Communication Links

It is important and very challenging to determine automatically who speaks to whom in multi-party discourse. In our annotation process, we ask annotators to classify each utterance in the chat by marking it as either a) addressed to someone or everyone; b) a response to someone else's specific prior utterance; or c) a continuation of one's own prior utterance. Using annotated data from this layer of annotation; we can train a communication link classification module, which

uses context, inter-utterance similarity and proximity of utterances as some of the features in a Naïve Bayes classifier to automatically classify utterances in one of the above-mentioned three categories. The current performance of this module is 61% accuracy as measured against annotated ground truth data. Indices that are calculated using this automatic module include the Communication Links Measure (CLM).

3.2 Dialogue Acts

We have developed a hierarchy of 15 dialogue acts in order to annotate the functional aspect of an utterance in discourse. The tag set adopted is based on DAMSL (Allen and Core, 1997) and SWBD-DAMSL (Jurafsky et al., 1997), but compressed to 15 tags tuned towards dialogue pragmatics and away from more surface characteristics of utterances. A detailed description of dialogue act tags and annotation procedure has been described in a separate publication. Some dialogue acts that are note-worthy are: Assertion-Opinion, Disagree-Reject, Agree-Accept, Offer-Commit, Information-Request and Action-Directive. Annotated data from this process is used to train a cue-phrase based dialogue act classifier adapted from Webb and Ferguson's (2010) approach, which currently performs at 64% accuracy. Our Cumulative Disagreement Measure (CDM) is calculated using the proportion of disagreement dialogue act utterances detected for each participant by this automatic module. Directive Index (DI) for Task Control is also computed by counting the number of Action-Directive and Offer-Commit types of dialogue acts made by participants.

3.3 Local Topics

Local topics are defined as nouns or noun phrases introduced into discourse that are subsequently mentioned again via repetition, synonym, or pronoun. Annotators were asked to mark all nouns and noun phrases of import from the discussion. We use Stanford part-of-speech tagger (Toutanova et al., 2003) to automatically detect nouns from text. Princeton's Wordnet (Miller et al., 1990) is consulted to identify synonyms commonly used in co-references. Since POS taggers are typically trained on well-formed text, performance of POS tagging on chat text – where grammar may be disorganized, use of abbreviations and symbols etc. may be quite frequent – would affect the accuracy of POS tagging. Our automatic local topic detection module performance is at 70% in the current system prototype. Several indices including Local Topic Introductions (LTI) and Citation Rate Index (CRI) are computed using this module.

We note here that it is not the goal of this research to develop the best POS tagger or the most accurately performing dialogue act classifier. In spite of the shortcomings in the computational modules that support our index calculations, we are able to achieve very robust performance in our intended task of modelling complex social roles. This is because we base our claims of sociolinguistic behaviors on repeated counts of each linguistic phenomenon over the length of entire discourse. When computational modules such as local topic detection fail, such errors are systematic, and would be replicated for each participant in their index scores. If the count for each participant were not fully accurate, nevertheless, the *distribution* of counts for all participants would still hold, thus giving us the desired ranking or the degree of sociolinguistic behaviour for each participant.

Having multiple indices for each behavior helps us account for error introduced from automatic modules. If the predictions on individual indices are not always consistent, we can still combine them into a single output by using different weighting schemes, albeit with lesser confidence. In

order to validate our proposed indices and measures, we analyzed their correlation with each other, both from human annotated data as well as our automatic process, as we shall discuss next.

4 Correlations

4.1 Correlations between proposed indices and behaviors

We compute the scores for each participant of a dialogue for each proposed index. For example, in Table 1, we show the correlation between component indices for Topic Control measure computed for a sample chat session from the MPC corpus – Turn Length (TL), Subsequent Mentions of Local Topics (SMT) and Local Topic Introductions (LTI). If the proposed indices indeed measure the same phenomena, then the correlation between them should be very high. The Cronbach's alpha (1951, 2003) for the scores shown in Table 1 is 0.96, which is extremely high. In Table 2, we show the correlation among selected indices used to compute Involvement measure (INVX). These are Noun Phrase Index (NPI), Turn Index (TI) and Topic Chain Index (TCI). The Cronbach's alpha for this table is also extremely high.

	TL	SMT	LTI	TCM
TL	1.0			
SMT	0.96	1.0		
LTI	0.78	0.80	1.0	
TCM	0.92	0.95	0.88	1.0
α	0.96			

	NPI	TI	TCI	INVX
NPI	1.0			
TI	0.76	1.0		
TCI	0.97	0.83	1.0	
INVX	0.96	0.83	0.98	1.0
α	0.98			

TABLES 1, 2 – Correlation among selected Topic Control and Involvement indices for a sample online chat dialogue

For all sessions we looked at, the correlation among indices of all proposed measures is quite strong, averaging above 0.93 for the Cronbach's alpha reliability statistic. Thus, strong correlations were seen among the index scores both for annotated data as well as automatic computation.

4.2 Correlations between proposed measures and human assessments

In addition to the correlation among indices within a proposed measure, we also computed the correlations among our proposed measures of Leadership and Influence. Figure 4 displays the correlations between the four measures of Leadership – Topic Control Measure (TCM), Skilled Control Measure (SCM), Involvement Measure (INVX) and Cumulative Disagreement Measure (CDM) on an actual 10-participant chat session from MPC corpus. Figure 5 shows the correlations between four measures – Topic Control Measure (TCM), Cumulative Disagreement Measure (CDM), Network Centrality Measure (NCM) and Measure of Argument Diversity (MAD). These measures were calculated for an actual 9-person chat session from the MPC corpus. For both Figures 4 and 5, the x-axes are the anonymized participant names DE, NT and so on and the y-axes are the scores on each of the measures, normalized as percentages.

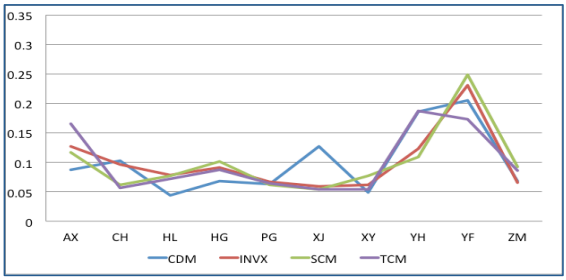


FIGURE 4 – Correlation between selected Leadership measures for a typical chat dialogue

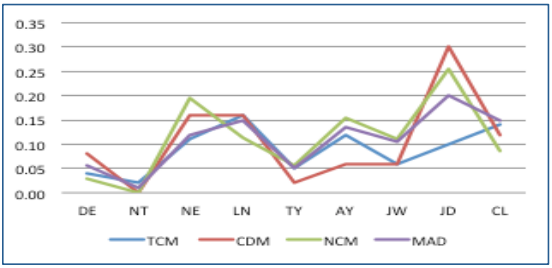


FIGURE 5 – Correlation between selected Influence measures for a typical chat dialogue.

Additionally, we show the correlations between all proposed Leadership and Influencer measures in Tables 3 and 4, to note one important finding. We note that Cumulative Disagreement Measure correlations are lower than the other measures, pointing to evidence of it being the discriminant variable. We have observed similar correlation patterns across the sessions we have looked at.

Leader-ship	SCM	TCM	INVX	CDM
SCM	1.0			
TCM	0.72	1.0		
INVX	0.95	0.77	1.0	
CDM	0.66	0.71	0.76	1.0

Infl-uence	NCM	MAD	TCM	CDM
NCM	1.0			
MAD	0.86	1.0		
TCM	0.98	0.86	1.0	
CDM	0.58	0.59	0.48	1.0

TABLE 3, 4 – Correlation among measures of Leadership and Influence for sample chat dialogue

Computing the correlation against human rankings elicited using survey questionnaire provides us with evidence that indeed the proposed behaviors are measuring the correct phenomena. The correlation between rankings produced by annotated data and ranking induced by participant ratings holds quite strongly across a significant proportion of data sets in our corpus with an average of over 0.80 Cronbach's alpha. Using this evidence of high correlations among indices, among behaviors and their measures, as well as measures against human survey ratings, we can be confident about our approach in measuring and detecting Leadership and Influence.

5 Evaluation and Results

In Table 5, we show how Leadership and Influence are present across different sessions in the MPC corpus. This is determined from participant ratings on post-discussion survey. On average, across both English and Chinese data, in 44.8% of dialogues different participants held the roles of the (task) Leader and the Influencer. Consequently, a significant portion of the MPC corpus has distinct participants exhibiting sociolinguistic behaviors consistent with either Leadership or Influence but not both. In the remaining dialogues, the same person was the leader and the influencer, thus exhibiting both types of behaviors at the same time. An automatic system should be able to distinguish between cases where the leader and influencer are the same person or different; the MPC corpus has a sufficient number of sessions for both cases.

Leader and Influencer are the same participant	Leader and Influencer are different participants
55.2%	44.8%

TABLE 5 – Number of sessions in MPC corpus where Leadership and Influence are exhibited by same or different participants (in percentages)

We compute the scores for each participant for all proposed measures. Although we have a full ranking of participants, both from survey ratings as well as system output, we are only interested in participants who have the highest Leadership and Influence. This means, the top-ranking participant on both rankings should match in order evaluate system performance. In cases where the top two individuals are quite close in the survey scores, we may consider top two participants.

We calculate the Leadership and Influencer score for all participants by taking the mean of our measures and deriving a Leadership and Influencer score for each participant. Using this simple weighting scheme of taking an average across all measures of the corresponding behavior, our accuracy is ~51% in predicting the top Leader and ~70% in predicting the top Influencer across our test data set. However, this scheme does not take into account the evidence found using correlations among measures. We have found that, on average, TCM measure correlates higher than other measures for Influence with survey ratings. We also discovered that SCM (Skilled Control Measure) correlates higher in Chinese dialogues than in English for Leadership. Consequently, we devised a weighting scheme that reflects the evidence found from our analysis of correlations against survey ratings.

So, the weighting scheme for English chat dialogues is:

Leader score = $(\alpha_{TCM} * TCM) + (\alpha_{SCM} * SCM) + (\alpha_{INVX} * INVX) + (\alpha_{CDM} * CDM)$

Where $\alpha_{TCM} > \alpha_{SCM} > \alpha_{CDM} > \alpha_{INVX}$

Influencer score = $(\alpha_{TCM} * TCM) + (\alpha_{CDM} * CDM) + (\alpha_{NCM} * NCM) + (\alpha_{MAD} * MAD)$

Where $\alpha_{TCM} > \alpha_{NCM} > \alpha_{MAD} > \alpha_{CDM}$

Similar combinations are derived for Chinese chat dialogues as well. Using this weighting scheme, we compute the Leadership and Influencer scores again. We illustrate this for Leadership in Table 6; the corresponding analysis is also applied for Influence in Table 7. In Table 6, we see that participant CC has the highest score on leadership from survey rating (4.33), followed by AA (score of 4). If we combine the scores computed automatically by our system on Leadership measures, TCM, SCM, INVX and CDM by taking an average, participant AA scores the highest (0.28). However, using the weights learnt from correlations, these scores can be correctly combined to get a score of 0.59 for CC.

Partici -pant	Survey Score	TCM	SCM	INVX	CDM	Leadership without weights	Leadership with weights
AA	4	0.23	0.25	0.31	0.33	0.28	0.48
BB	1.67	0.15	0.13	0.12	0.09	0.12	0.30
CC	4.33	0.27	0.43	0.21	0.14	0.26	0.59
DD	0.67	0.09	0.06	0.09	0.12	0.09	0.20
EE	1	0.10	0.06	0.13	0.09	0.09	0.22
FF	3.33	0.16	0.08	0.15	0.23	0.16	0.29

TABLE 6 – Leadership score computed using linear combination with weights on a sample English chat dialogue

Participant	Survey Score	TCM	CDM	NCM	MAD	Influencer w/ weights
AA	5.5	0.11	0.15	0.06	0.17	0.20
BB	5	0.13	0.15	0.19	0.11	0.28
CC	6.84	0.26	0.25	0.33	0.13	0.49
DD	4.67	0.23	0.25	0.21	0.28	0.44
EE	5	0.15	0.1	0.16	0.14	0.28

TABLE 7 – Influence score computed using linear combination with weights on a sample English chat dialogue

The accuracy of our predictions on our test data set improves to 80% for both Leadership and Influence after factoring in the weights. For different data types and different languages, we have learnt different weighting schemes where the sociolinguistic behaviors may be combined differently to compute scores. In essence, the higher the correlation, the greater the weight given to the measure. As expected and shown in Table 8, different correlations hold across languages – hence possibly cultures, since the participants are native speakers. We can see from the weighting schemes that different behaviors account for Leadership and Influence in different cultures. The Measure of Argument Diversity (MAD) has a higher correlation with Influence in Chinese chat as compared to English chat dialogues. Where the scores are 0, it signifies that the behavior is found to not correlate well with the other measures that comprise the phenomena being modeled; hence we do not include these behaviors while taking linear combinations. They may be either negatively correlated or demonstrate very low correlation; in both cases, the evidence from those behavior should not be included while predicting that sociolinguistic phenomena.

Weights	TCM	SCM	CDM	INVX	NCM	MAD
English Chat Leadership	0.75	0.6	0.3	0.23	0	0
Chinese Chat Leadership	0.68	0.73	0.36	0.45	0	0
English Chat Influence	0.75	0	0.15	0	0.5	0.4
Chinese Chat Influence	0.75	0	0.1	0	0.34	0.75

TABLE 8 – Weighting schemes for combining behaviors learnt from correlation analyses

Illustrated in Table 9, are the correlations between Measure of Argument Diversity (MAD) and Network Centrality Measure (NCM), which are behaviors that are consistent with Influence; and Involvement (INVX) and Skilled Control Measure (SCM) which are the behaviors consistent

with Leadership. We can see that MAD and NCM correlate quite highly with each other; as do INVX and SCM. By contrast, NCM and INVX correlation is very low, as is SCM and MAD. Topic Control Measure (TCM) and Cumulative Disagreement Measure (CDM) are common to both social phenomena; they exhibit high correlations and are not included in Table 9.

Correlation	MAD	NCM	INVX	SCM
MAD	1			
NCM	0.78	1		
INVX	-0.15	0.14	1	
SCM	-0.25	0.11	0.97	1

Table 9. Correlations between Leadership and Influencer measures on sample chat dialogue

In Tables 10 and 11, we show performance of detecting the top leader and influential participant across languages for the two methods we proposed, when compared to baseline. The baseline we have chosen is to pick a participant at random to be top influential person. In a small group discussion found in our corpus, this is a reasonable baseline given the limited number of participants. We could choose another baseline, such as selecting the participant with the most number of turns as the Leader or Influencer. However, we see similar performance for such baselines as the random one.

Performance Leadership	Baseline	Without Weights	With Weights	Including TFM
English Chat	17.85%	37.5%	80%	80%
Chinese Chat	12.5%	64%	72.7%	90.9%
Average	15.65%	50.75%	76.35%	85.45%

TABLE 10 – Performance of system against random baseline, with and without weighting scheme, for Leadership

Performance Influence	Baseline	Without Weights	With Weights
English Chat	17.85%	71.40%	78.50%
Chinese Chat	12.5%	69%	90%
Average	15.65%	70.2%	84.25%

TABLE 11 – Performance of system against random baseline, with and without weighting scheme, for Influence

In a separate publication (Taylor et al., 2012), we have discussed the development of an additional sociolinguistic behavior to predict Leadership in Chinese dialogues – called the **Tension Focus Measure (TFM)**. This behavior is defined as the degree to which a speaker is someone at whom others direct their disagreement, or with whose topics they disagree the most. Using this additional measure, our performance on detecting the top leader goes up to 85% average for English and Mandarin test data set.

Conclusion and perspectives

We have shown a novel, robust method for modeling social phenomena in multi-party discourse. We have combined established social science theories with computational modeling to create a two-tier approach that can detect high-level sociolinguistic phenomena such as Leadership and Influence in language with a high degree of accuracy. In future work, we have planned for a

larger scale evaluation, testing index stability, and resilience to errors in automated language processing, including topic detection, coreference resolution, and dialogue act classification. Current performance of the system is based on versions of these linguistic modules, which perform at about 70% accuracy, so these need to be improved as well. We could also experiment with training a classifier to learn the weights automatically, which we plan to report in a future publication.

The advantage of applying a two-tier approach is that we can add or remove mid-level sociolinguistic behaviors efficiently when applying our models to different data types and languages. As we have noted, we can insert additional sociolinguistic behaviors such as Tension Focus Measure, if our existing models do not completely account for Leadership in certain data sets. Such sociolinguistic analysis is impractical in a straight-forward machine-learning approach where one can add all features to a learning algorithm to decide how features may best be combined. A machine-learning approach modeled directly on linguistic features would not be easily transferable to other data types and could prove brittle. Some measures turn out to be more predictive in a given data genre, and when applied appropriately, perform well at predicting phenomena as rated and understood by human assessors. We note that there may be some variance as to how humans perceive the concept of Leadership and Influence and rate a participant based on their intuitive notion of the concept. The fact that we have multiple indicators in the form of indices and measures helps us overcome the potential variance in this perception. Another advantage of a two-tier approach is that some of the existing measures can be combined differently as we have demonstrated using the CDM and TCM measures. These behaviors are consistent with both Leadership and Influence. When trying to model additional higher-level sociolinguistic phenomena beyond Leadership and Influence, we can use existing measures in a manner that is substantiated by social science theory as well as revealed in our computational analyses.

Acknowledgments

This research was funded by the Office of the Director of National Intelligence (ODNI), Intelligence Advanced Research Projects Activity (IARPA), and through the U.S. Army Research Lab. All statements of fact, opinion or conclusions contained herein are those of the authors and should not be construed as representing the official views or policies of IARPA, the ODNI or the U.S. Government.

References

Agar, M. (1994). Language Shock, Understanding the Culture of Conversation. Quill, William Morrow, New York.

Allen, J. and M. Core. (1997). Draft of DAMSL: Dialog Act Markup in Several Layers. www.cs. rochester.edu/research/cisd/resources/damsl/

Austin, J. L. (1962). *How to do Things with Words*. Clarendon Press, Oxford.

D. B. Bracewell, M. Tomlinson, Y. Shi, J. Bensley, and M. Draper. (2011). Who's playing well with others: Determining collegiality in text. In *Proceedings of the 5th IEEE International Conference on Semantic Computing (ICSC 2011).*

George Broadwell, Jennifer Stromer-Galley, Tomek Strzalkowski, Samira Shaikh, Sarah Taylor, Umit Boz, Alana Elia, Laura Jiao, Ting Liu and Nick Webb. (2012). Modeling Socio-

Cultural Phenomena in Discourse. *Journal of Natural Language Engineering, Cambridge Press.*

Beebe, S. A., & Masterson, J. T. (2006). *Communicating in small groups: Principles and practices*. Boston, MA: Pearson/Allyn and Bacon

Blaylock, N. (2002). Managing Communicative Intentions in Dialogue Using a Collaborative Problem-Solving Model. Technical Report 774, University of Rochester, CS Dept.

Bradford, L. P. (1978). Group development. La Jolla, Calif., University Associates.

Bunt, H. (1994). Context and Dialogue Control. *Think Quarterly 3(1), 19-31.*

Carberry, S. and L. Lambert. (1999). A Process Model for Recognizing Communicative Acts and Modeling Negotiation Dialogue. *Computational Linguistics*, 25(1), pp. 1-53.

Carletta, J. (2007). Unleashing the killer corpus: experiences in creating the multi-everything AMI Meeting Corpus. *Language Resources and Evaluation Journal* 41(2): 181-190

Carlson, L. (1983). *Dialogue Games: An Approach to Discourse Analysis*. D. Reidel.

Cronbach, L. J. Coefficient alpha and the internal structure of tests. (1951). *Psychometrika, 16(3)*, 297-334.

Cronbach, Lee J., and Richard J. Shavelson. (2004). My Current Thoughts on Coefficient Alpha and Successor Procedures. *Educational and Psychological Measurement* 64, no. 3 (June 1): 391-418. doi:10.1177/0013164404266386.

Ellis, D. G., & Fisher, B. A. (1994). *Small group decision making: Communication and the group process*. New York: McGraw-Hill

Forsyth, E. N. and C. H. Martell. (2007). Lexical and Discourse Analysis of Online Chat Dialog. First IEEE International Conference on Semantic Computing (ICSC 2007), pp. 19-26.

Marjorie Freedman, Alex Baron, Vasin Punyakanok and Ralph Weischedel. (2011). "Language Use: What it can tell us?" In *Proceedings of the Association of Computational Linguistics.* Portland, Oregon.

Givon, T. (1983). *Topic continuity in discourse: A quantitative cross-language study.* Amsterdam: John Benjamins.

Huffaker, D. (2010). Dimensions of leadership and social influence in online communities. *Human Communication Research, 36*, 596-617.

Ji, G. and J. Bilmes. (2006). Backoff Model Training using Partially Observed Data: Application to Dialog Act Tagging in proceedings of the Human Language Technology/ American chapter of the Association for Computational Linguistics (HLT/NAACL'06)

Jurafsky, D., E. Shriberg and D. Biasca. (1997). Switchboard SWBD-DAMSL Shallow-Discourse-Function Annotation Coders Manual. http://stripe.colorado.edu/ ~jurafsky/ manual.august1.html

Levin, L., A. Thyḿe-Gobbel, A. Lavie, K. Ries, and K. Zechner. (1998). A Discourse Coding Scheme for Conversational Spanish. In Proceedings of the International Conference on Speech and Language Processing.

Linell, P. (1990). The power of dialogue dynamics. In Ivana Markov´a and Klaus Foppa,

editors, *The Dynamics of Dialogue*. Harvester, 147–177.

Liu, T., Samira Shaikh, Strzalkowski, T., Broadwell, A., Stromer-Galley, J., Taylor, S., Boz, Umit., Ren, Xiaoai. and Jingsi Wu. (2012) Extending the MPC corpus to Chinese and Urdu - Multi-party Multi-lingual Chat Corpus for Modeling Social Phenomena in Language. In *Proceedings of the Eight International Conference on Language Resources and Evaluation (LREC'12).*

Miller, G. A., Beckwith, R., Fellbaum, C. D., Gross, D., and Miller, K. (1990). WordNet: An online lexical database. *Int. J. Lexicograph*. 3(4): 235--244.

Poesio, M. and A. Mikheev. (1998). The predictive power of game structure in dialogue act recognition. International Conference on Speech and Language Processing *(ICSLP-98)*.

Samuel, K., S. Carberry, and K. Vijay-Shanker. (1998). Dialogue Act Tagging with Transformation-Based Learning. In Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics, Montreal.

Sanders, R. E., Pomerantz, A., Stromer-Galley, J. (2010). Some ways of taking issue with another participant during a group deliberation. Paper presented at the annual meeting of the National Communication Association, San Francisco, CA

Scollon, R. and S. W. Scollon. (2001). *Intercultural Communication, A Discourse Approach*. Blackwell Publishing, Second Edition.

Searle, J. R. (1969). *Speech Acts*. Cambridge University Press, London-New York.

Shaikh, S., T. Strzalkowski, S. Taylor and N. Webb. (2010). MPC: A Multi-Party Chat Corpus for Modeling Social Phenomena in Discourse, in proceedings of the 7th International Conference on Language Resources and Evaluation (LREC2010), Valletta, Malta. 2010.

Shriberg, E., R. Dhillon, S. Bhagat, J. Ang and H. Carvey. (2004). The ICSI Meeting Recorder Dialog Act (MRDA) Corpus, in proceedings of 5th SIGdial Workshop on Discourse and Dialogue, M. Strube and C. Sidner (eds.), April 30-May 1, Cambridge, MA, pp.97-100, 2004

Stolcke, A., K. Ries, N. Coccaro, E. Shriberg, R. Bates, D. Jurafsky, P. Taylor, R. Martin, C. Van Ess-Dykema, and M. Meteer. (2000). Dialogue Act Modeling for Automatic Tagging and Recognition of Conversational Speech. *Computational Linguistics* 26(3), 339–373.

Strzalkowski, T., Broadwell, G. A., Stromer-Galley, J., Shaikh, S., Taylor, S., & Webb, N. (2010) Modeling socio-cultural phenomena in discourse. In *Proceedings of The 23rd International Conference on Computational Linguistics. Beijing, China.*

Taylor S., Ting Liu, Samira Shaikh, Tomek Strzalkowski, Jenny Stromer-Galley, Aaron Broadwell, Umit Boz, Xiaoai Ren, Jingsi Wu and Feifei Zhang. (2012). Chinese and American Leadership Characteristics - Discovery and Comparison in Multi-party On-line Dialogues. In *Proceedings IEEE International Conference on Semantic Computing, Special Session, in Palermo, Italy.*

Twitchell, D. P., J. F. Nunamaker Jr., and J. K. Burgoon. (2004). Using Speech Act Profiling for Deception Detection. *Intelligence and Security Informatics*, LNCS, Vol. 3073

Kristina Toutanova, Dan Klein, Christopher Manning, and Yoram Singer. (2003). Feature-Rich

Part-of-Speech Tagging with a Cyclic Dependency Network. In *Proceedings of HLT-NAACL 2003*, pp. 252-259.

Webb, N. and M. Ferguson. (2010). Automatic Extraction of Cue Phrases for Cross-Corpus Dialogue Act Classification, in the proceedings of the 23rd International Conference on Computational Linguistics (COLING-2010), Beijing, China. 2010.

NEER: An Unsupervised Method for Named Entity Evolution Recognition*

Nina TAHMASEBI *Gerhard GOSSEN* *Nattiya KANHABUA*
Helge HOLZMANN *Thomas RISSE*
L3S Research Center, Appelstr. 9a, 30167 Hannover, Germany
{tahmasebi, gossen, kanhabua, holzmann, risse}@L3S.de

ABSTRACT
High impact events, political changes and new technologies are reflected in our language and lead to constant evolution of terms, expressions and names. Not knowing about names used in the past for referring to a named entity can severely decrease the performance of many computational linguistic algorithms. We propose NEER, an unsupervised method for named entity evolution recognition independent of external knowledge sources. We find time periods with high likelihood of evolution. By analyzing only these time periods using a sliding window co-occurrence method we capture evolving terms in the same context. We thus avoid comparing terms from widely different periods in time and overcome a severe limitation of existing methods for named entity evolution, as shown by the high recall of 90% on the New York Times corpus. We compare several relatedness measures for filtering to improve precision. Furthermore, using machine learning with minimal supervision improves precision to 94%.

TITLE AND ABSTRACT IN GERMAN

NEER: Eine nichtüberwachte Methode zur Erkennung von Namensevolution

Wichtige Ereignisse, politische Veränderungen und neue Technologien spiegeln sich in unserer Sprache wieder und führen zu einer ständigen Evolution von Begriffen, Ausdrücken und Namen. Mangelndes Wissen über frühere Namen einer Entität kann die Leistungsfähigkeit vieler computerlinguistischer Methoden deutlich verringern. In diesem Papier präsentieren wir unsere nichtüberwachte Methode namens NEER zur Erkennung von Namensevolution, die unabhängig von externen Datenquellen arbeitet. Indem wir Zeiträume mit erhöhter Evolutionswahrscheinlichkeit mit Hilfe einer Kookkurrenzmethode basierend auf *Sliding Windows*-Verfahren untersuchen, erfassen wir evolvierende Terme im selben Kontext. Dadurch vermeiden wir es, Terme aus weit auseinander liegenden Zeiträumen zu vergleichen und umgehen damit eine schwerwiegende Beschränkung vorhandener Methoden. Dieses zeigt sich an einer gemessenen Sensitivität von 90% auf dem Korpus der New York Times. Um die Genauigkeit zu erhöhen, vergleichen wir mehrere Ähnlichkeitsmaße zur Filterung. Mit Hilfe von maschinellem Lernen mit minimaler Überwachung verbessern wir die Genauigkeit auf 94%.

KEYWORDS: Temporal Named Entity Evolution, Named Entity Changes, Machine Learning.

GERMAN KEYWORDS: Zeitliche Namensevolution, Namensänderungen, Maschinelles Lernen.

*This work is partly funded by the European Commission under ARCOMEM (ICT 270239)

Proceedings of COLING 2012: Technical Papers, pages 2553–2568,
COLING 2012, Mumbai, December 2012.

1 Introduction

Do you remember the bright yellow Walkman, Joseph Ratzinger or Andersen Consulting? Chances are you do not, because as the world around us changes, new terms are created and old ones are forgotten. High impact events, political changes and new technologies are reflected in our language and lead to constant evolution of terms, expressions and names. Most everyday tasks, like web search, have so far relied on the good memory of users or been restricted only to the current names of entities. As the web and its content grow older than some of its users, new challenges arise for natural language tasks like information retrieval and knowledge consolidation to automatically determine relevant information, even when it is expressed using forgotten terms.

Language evolution is reflected in documents available on the web or in document archives but is not sufficiently considered by current applications. Therefore, not knowing about different names referencing the same entity severely compromises system effectiveness. This can partially be addressed by using external knowledge sources like DBpedia. The limitation of this approach is that these resources do not cover all entities and are not able to capture ephemeral names or jargon used in everyday language or social media.

There are several kinds of language evolution, among others spelling variations, name changes and concept changes. We focus on named entity evolution, the detection of name changes, as it has a high impact, for example in information retrieval. This research field is becoming increasingly important. However, most previous work depend on the availability of external knowledge sources or assume a static context around terms and expect the names to be the only changing factor. We follow a statistical approach to eliminate the dependency on external resources and use a context based method that considers only periods with a high likelihood of name change, thereby capturing evolving names with less computational effort. This independence opens the possibility to apply the method to any corpus, including historical collections or those in different languages, and to identify undocumented named entity evolution.

The main contributions of this paper are:

- We propose the use of change periods (i.e., periods with high likelihood of name change) to capture the evolution of one name into another instead of comparing names and their contexts from vastly different periods in time.
- We propose NEER, a method for named entity evolution recognition that analyzes the context of entities during time periods of evolution. The proposed method is independent from external knowledge sources and is able to find name changes.
- We describe and compare named entity evolution filtering methods, statistical as well as machine learning based, that capture relatedness between different names used to reference the same entity at different points in time in order to increase accuracy.
- We apply NEER on a standard dataset (New York Times corpus) to identify named entity evolution and evaluate using precision and recall. We make our test set publicly available to encourage comparison of results.

The rest of the paper is organized as follows: In the next section we review related work in areas relevant to named entity evolution. In Section 3 we define terminology needed for our work. We describe our approach and highlight the limitations of previous work in Section 4 and present the NEER method in Section 5. We introduce our data and test sets in Section 6 and present our experimental results. We discuss our findings in Section 7 and finally present our conclusions and future work.

2 Related Work

Previous work on automatic detection of term evolution has been very limited and mainly focused on named entity evolution. The interest has largely been from an information retrieval point of view as named entity evolution makes finding relevant documents more challenging.

Berberich et al. (2009) propose reformulating a query into terms prevalent in the past; they measure the degree of relatedness between two terms when used at different times by comparing the contexts as captured by co-occurrence statistics. This approach requires a recurrent computation each time a query is submitted as it requires a target time point for the query reformulations which reduces efficiency and scalability. The results presented in this paper are "anecdotal" (to use the words of the authors) and thus cannot be used for direct comparison. However, because of the promising results we use the same method for defining a context.

Kaluarachchi et al. (2010) propose to discover semantically identical concepts (or named entities) used at different time periods using association rule mining to associate distinct entities to events. Sentences containing any subject, verbs, objects, and nouns are targeted and the verb is interpreted as an event. Two entities are considered semantically related if their associated event is the same and the event occurs multiple times in a document archive. The temporally related term of a given named entity is used for query translation (or reformulation) and results are retrieved appropriately w.r.t. specified time criteria. They present precision and recall for very few queries and evaluate only indirectly on the basis of retrieved documents.

Kanhabua and Nørvåg (2010) define a time-based synonym as a term semantically related to a named entity at a particular time period. They extract synonyms of named entities from link anchor texts in Wikipedia articles using the full history. Unfortunately, link information, such as anchor text, is rarely available and thus limits the method to hypertext collections. They evaluate the precision and recall of the time-based synonyms by measuring increased precision and recall in search results rather than directly evaluating the quality of the found synonyms.

3 Definitions and Terminology

Language evolution is a broad concept which can be divided into several sub-classes including spelling variations (Ernst-Gerlach and Fuhr, 2007; Hauser et al., 2007; Gotscharek et al., 2009) and word sense evolution (Tahmasebi et al., 2011). The general class of **term to term evolution** contains terms of any part of speech that have been used to mean the same thing at different points in time. The shift between terms typically occurs over a long period of time, e.g., one sense of the term *cool* was previously expressed with the term *collected*. In this paper, we focus on a special case of term to term evolution namely **named entity evolution** which considers a given entity and the different lexical names for the same entity. Here, the entity is fixed while the name changes over time.

Named entity changes do not need to have any lexical overlap between two representations, for example *Joseph Ratzinger* was the birth name of *Pope Benedict XVI* and *Hillary Rodham* was *Hillary Clinton*'s maiden name. The latter is an easier case of evolution because of the overlapping first name and can be targeted using entity consolidation or linking techniques (Shen et al., 2012; Ioannou et al., 2010). However, most existing techniques do not take historic changes into account and only focus on merging concurrent representations of the same entity.

We consider a term w_i to be a single or multi-word lexical representation of an entity at time t_i. The **context** C_{w_i} is the set of all terms related to w_i at time t_i. Similar to Berberich et al. (2009)

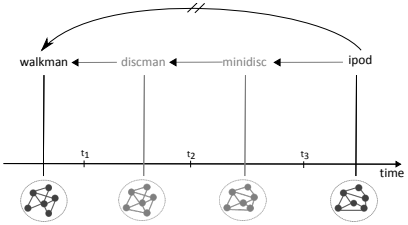


FIGURE 1: Existing methods detect evolution by comparing term contexts from different times. E.g., by directly comparing *walkman* to *ipod*.

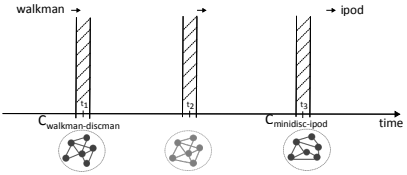


FIGURE 2: NEER detects by creating one context for each change period, thus eliminating the need to compare contexts.

we consider the most frequently co-occurring terms within a distance of k words as the context, however, other contexts can be used. We consider a **change period** to be a period of time in which one term evolves into another.

Co-references are expressions that refer to the same entity. In the sentence "The president said he had discussed the issue" the words *the president* and *he* refer to the same person. In this paper, we consider **temporal co-references** to be different lexical representations that have been used to reference the same concept or entity at the different periods in time.

We have two variations of temporal co-references, direct and indirect. **Direct temporal co-references** are temporal co-references that are variations of each other with some lexical overlap. **Indirect temporal co-references** are temporal co-references that lack lexical overlap on the token level. *Hillary Clinton* and *Hillary Rodham* are examples of direct temporal co-references while *Pope Benedict XVI* and *Joseph Ratzinger* are examples of indirect temporal co-references. All introduced terms will be used with and without *temporal* interchangeably.

A **temporal co-reference class** contains all direct temporal co-references for a given named entity, denoted as $\text{coref}_r\{w_1, w_2, \ldots\}$. Each temporal co-reference class is represented by a **class representative** r which is also a member of the class. For example, *Joseph Ratzinger* is the representative of the co-reference class containing the terms {*Joseph Ratzinger, Cardinal Ratzinger, Cardinal Joseph Ratzinger,* ... }.

4 Approach

Previous work in the area can be generalized as shown in Figure 1. A word w_i (*walkman*) is mapped to its context and compared to word w_j (*ipod*) by comparing contexts. If the corresponding contexts are similar it is concluded that w_i and w_j are temporal co-references. These methods have severe drawbacks because they assume that the queried entity is the only evolving factor and that contexts stay stable over time. This is however not the case. Comparing the term *walkman* and *ipod* (an example given by Berberich et al. (2009)[1]) directly by means of contexts from the New York Times corpus ($C_{walkman}$, C_{ipod}) we find that the majority of the context terms have changed. In Table 1 we can see the contexts of related terms *discman, minidisc* and *mp3 player* during the year when each term was introduced[2].

[1]We do not consider *walkman* and *ipod* to be co-references as they do not correspond to the same named entity. We use this example to illustrate the difficulties that arise with state of the art and the differences to NEER.

[2]The *walkman* was already introduced in 1979, so we chose the first year of the corpus' time span (1987).

$C_{walkman}$	$C_{discman}$	$C_{minidisc}$	$C_{mp3\ player}$	C_{ipod}
cassette	walkman	compact	music	apple
audio	stillvideo	disc	digital	mp3
video	sony	sony	internet	roqit
tape	portable	digital	audio	player
music	cd	cassette	player	music
sony	kodac	phillips	files	geeks
digital	video	walkman	cd	jukebox
stereo	priestly	dcc	computer	portable
earphones	digital	prerecorded	mp3	macintosh
recorders	camera	video	portable	dlink

TABLE 1: Five terms and their contexts in the New York Times corpus.

We find that the only overlap between $C_{walkman}$ and C_{ipod} is the term *music*. By comparing the intermediate contexts pairwise instead we find that there is a much larger overlap between the contexts. For instance, $C_{walkman}$ has a 40% overlap with $C_{discman}$ which in turn has a 30% overlap with $C_{minidisc}$. The same properties hold when we compare the 20 most frequent terms and we find that the overlap between $C_{mp3player}$ and C_{ipod} increases further. From this we deduce that comparing contexts pairwise where the contexts are closer in time is more effective than comparing two contexts far apart in time.

The same observation holds for Kaluarachchi et al. (2010). They consider nouns to have evolved into each other if they point to the same event (represented by a verb) at different points in time. Over long periods of time also the verb undergoes evolution and hence the method is limited only to those terms where the corresponding event has not changed over time.

In this paper we make use of the typical characteristics of named entity evolution. Unlike with other types of evolution, such as word sense evolution, named entity changes typically occur during a short time span. There are few concept shifts where the term slowly changes, instead name changes occur due to special events like being elected pope, getting married or merging/splitting a company. If the named entity is of general interest, these name changes will also be announced to the public repeatedly during the change period with sentences like "The day after Cardinal Joseph Ratzinger became Pope Benedict XVI"[3].

By first identifying candidate change periods and then creating a context around a term, we believe that we can capture both the old and the new co-reference in the same context. We thus eliminate the risk of comparing contexts that are vastly different. Figure 2 illustrates our method. By identifying change periods t_1, t_2, t_3 we can create contexts around a term which contain both co-references and do not have to compare largely different contexts like those of *walkman* and *ipod*.

5 The NEER Method

To find temporal co-references we use the pipeline depicted in Figure 3. We start by detecting change periods for a query term over the entire collection. We make use of the identified change periods to find the subsets in which we look for evolution. We extract single and multi-word nouns and find named entities mentioned in the text.

We create contexts around extracted terms by applying co-occurrence analysis and use the context and the extracted terms to find direct co-references. Finally we apply frequency analysis as well as machine learning to identify direct and indirect co-references and to filter out noise.

[3]The New York Times, April 21, 2005.

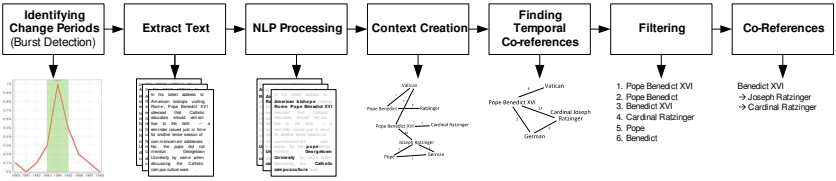


FIGURE 3: Pipeline used to detect temporal co-references.

5.1 Identifying Change Periods

Named entity changes are typically associated with significant events concerning the entities which lead to increased attention. We use this property to pinpoint change periods and detect those using a **burst detection algorithm**.

We use the Kleinberg algorithm (Kleinberg, 2003) to find bursts from the entire document collection $\mathbf{D}$. The algorithm models the frequency of documents $\mathbf{D}_w$ containing term w using a series of probability distributions. Each distribution represents an increasing degree of burstiness. A set of states indicates which distribution is active. By assigning a cost to state transitions the algorithm ensures that an optimal state sequence creates bursts that end only if they are followed by a sufficiently large period of lower activity. This avoids splitting bursts for example around weekends when the number of articles drops.

We detect bursts related to an entity by retrieving all documents in the corpus containing the query term, grouping them into monthly bins and running the burst detection on the relative frequency of the documents in each bin. Each resulting burst corresponds to a significant event involving the entity. However, these bursts do not necessarily correspond to a name change. By choosing the `topB` strongest bursts we expect to find a subset of bursts which also captures change periods. We denote each change period p_i for $i = 1, \ldots, \texttt{topB}$.

5.2 Creating Contexts

After identifying change periods p_i for an entity w we create a context for each period by extracting all documents $\mathbf{D}_{w_i}$ that mention the entity or any part of it and are published in the year corresponding to p_i. We extract nouns, noun phrases and named entities. We use noun phrases to capture more information and create richer contexts around entities. All extracted terms are added to a **dictionary** and used for creating a co-occurrence graph. The co-occurrence graph is an undirected weighted graph which links two dictionary terms if and only if they are present in $\mathbf{D}_{w_i}$ within k terms of each other. The weight of each link is the frequency with which the two terms co-occur in $\mathbf{D}_{w_i}$. The context of entity w is considered as all terms co-occurring with w. The context of a co-reference class is considered to be all terms co-occurring with any of the terms in the co-reference class.

5.3 Finding Temporal Co-references

To find *direct co-reference classes* we need to consolidate the extracted terms by recognizing all variants of each term. As an initial step each term from the dictionary with a frequency above `minFr` is placed in its own co-reference class where the term acts as the representative as well as the only co-reference: $\text{coref}_{\text{Benedict}}\{\text{Benedict}\}$.

Merging: The procedure for merging terms and co-reference classes is shared between all three rules described below; each co-reference class is represented by the term with the highest frequency. A frequency is stored in the co-reference class for the representative r as the sum frequency of all terms in the class. If two co-reference classes have the same representative, they are merged into one. Each co-reference class carries with it all co-occurrences that belong to any of the terms in the co-reference class. These are considered as the context C_r. When terms are merged, the context is updated accordingly.

Next we will describe the main rules used for finding all direct co-reference classes. In the initial iteration the first rule works on the dictionary terms and populates an index with co-reference representatives. In the second and all subsequent iterations the first rule makes use of the terms in the index. This index is passed through all the rules. The rules are iterated until there are no more terms in the index that can be merged.

1. **Prefix/suffix rule:** This rule creates co-reference classes by merging dictionary terms that differ only by a prefix or suffix. For example, the co-reference classes of *Pope Benedict* and *Benedict* as well as *Pope* and *Pope Benedict* are merged. In both cases the co-reference class has *Pope Benedict* as representative and these co-reference classes are therefore merged and result in $\text{coref}_{\text{Pope Benedict}}$*{Pope, Pope Benedict, Benedict}*.
2. **Sub-term rule:** This rule merges classes that are represented by terms that can be considered sub-terms. For a term to be a sub-term of another we require the longer term to contain all terms from the shorter term in the correct order. For example, the co-reference classes represented by *Cardinal Joseph Ratzinger* and *Cardinal Ratzinger* are merged.
3. **Prolong rule:** The third rule is used to create longer terms than might be found in the dictionary. It merges two representatives from the index into one longer term if the terms have an overlapping part and there exists a co-occurrence between the remaining terms. E.g., *Pope John Paul* and *John Paul II* are merged if there is a co-occurrence (*Pope John Paul , II*) or (*Pope , John Paul II*); the representative of the merged co-reference class is *Pope John Paul II*. The third rule also merges terms that differ due to plural of the prefixes assuming that the prefix is not considered a stopword. E.g., *Senator Barack Obama* and *Senators Barack Obama* are merged but *Mr Obama* and *Mrs Obama* are not.

Final merging When the terms in the index cannot be merged further, a final round of merging takes place. In this round we apply a *soft* sub-term rule where we drop the requirement that the terms should be in the same order but require them to be similar in frequency. This way terms like *Illinois Democrat* and *Democrat of Illinois* are merged.

Consolidation When all terms are merged we create a mapping from each term to the co-reference class representative that has the highest frequency. Using this map we consolidate all terms in the context of each class.

An example is shown in Figure 4 (original context in 4a). Using the three rules we find

$$\text{coref}_{\textit{Cardinal Joseph Ratzinger}}\{\textit{Cardinal Joseph Ratzinger, Joseph Ratzinger, Ratzinger}\}$$
$$\text{coref}_{\textit{Pope Benedict XVI}}\{\textit{Pope Benedict XVI, Pope Benedict,Pope}\}$$
$$\text{coref}_{Vatican}\{\textit{Vatican}\}$$
$$\text{coref}_{German}\{\textit{German}\}.$$

Next a mapping is created: [*Joseph Ratzinger* → *Cardinal Joseph Ratzinger*, *Ratzinger* → *Cardinal*

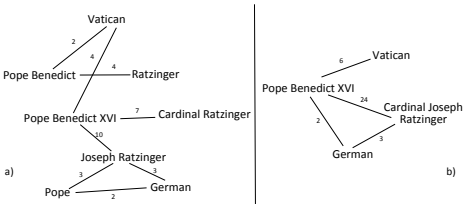


FIGURE 4: a) Example graph after creating context. b) After consolidating and merging all direct co-references.

Joseph Ratzinger, *Pope Benedict* → *Pope Benedict XVI*, *Pope* → *Pope Benedict XVI*]. Additionally, all co-reference class representatives map to themselves. Then each term in the co-reference class context is consolidated and replaced using the map. If two co-occurrences share a term they are merged into one and the frequency of the co-occurrence is updated as shown in Figure 4b.

Ranking The term frequencies and the merging steps offer a natural ranking of co-references. When two terms are merged like *Pope Benedict* and *Pope Benedict XVI* we update the frequency of the class representative by summing up the frequencies. During merging all co-occurrences are updated with the sum of the frequencies of all participating terms. In Figure 4b we see that the co-occurrence frequencies of (*Vatican* , *Pope Benedict XVI*) is 6 because the frequency of (*Vatican , Pope Benedict*) is 2 and (*Vatican , Pope Benedict XVI*) is 4. The term frequencies and co-occurrence frequencies are stored in each co-reference class. The frequency of *Pope Benedict* and *Pope Benedict XVI* is much higher than that of *Benedict XVI* and *Eggs Benedict* and during consolidation the term *Benedict* is replaced with *Pope Benedict XVI* rather than *Eggs Benedict*.

Indirect Co-references Indirect co-references are found implicitly by means of the direct co-references. After consolidation, all terms in the context of a co-reference class are considered candidate indirect co-references. These are a mix between true indirect co-references, highly related co-occurrence phrases as well as noise. The quality of the indirect co-references is dependent on the named entity extraction, co-occurrence graph creation and filtering of the co-occurrence graph. The choice of including single token terms in addition to multi-token terms has a high influence on the quality of the resulting co-occurrences. In Figure 4b *Vatican*, *German* and *Cardinal Joseph Ratzinger* are candidate co-references for *Pope Benedict XVI*.

If NEER does not find any co-references for a term, all direct co-occurrences from the co-occurrence graphs (derived from the union of the change periods) are returned instead.

5.4 Filtering Temporal Co-references

To remove noise and identify the true direct and indirect co-references we make use of the term frequencies as well as the document frequencies for the filtering. We start by describing similarity measures between terms and continue with the filtering techniques.

Similarity measures To keep true co-references we need to measure the temporal relatedness of terms. Unlike previous works that take temporal features into account it is not sufficient to consider relatedness over the entire time span of a collection. In Radinsky et al. (2011) time series of terms are used to capture the relatedness of terms like *war* and *peace* or *stock* and *oil*.

These terms are considered related because they have similar frequencies over time.

For temporal co-references, capturing the overall relatedness is not sufficient. Both direct and indirect co-references can be related only for a certain period in time and then lose their relation. To give an example: Both *Senator Clinton* as well as *Hillary Clinton* have been used to refer to the same person at different periods in time. As the latter name was only valid for a certain period in time, measuring the relatedness between *Hillary Clinton* and *Senator Clinton* using global term frequencies (i.e. term frequency over the entire corpus) will not yield the correct results. However, global measures can help to find direct co-references such as *Barack Obama* and *Barack Hussein Obama*.

Therefore, to fully capture temporal co-references we need, in addition to global relatedness measures, a relatedness measure that captures how related terms are during the time periods where they can be related at all. To this end we allow a relatedness measure to consider only periods where both terms occur. In all cases we use the normalized frequencies.

We consider four relatedness measures: (1) Pearson's Correlation ($corr$) (Weisstein, 2012a), (2) Covariance (cov) (Weisstein, 2012b), (3) Rank correlation (rc) and (4) Normalized rank correlation (nrc).

The two first measures are standard relatedness measures where $corr$ measures linear dependence between random variables while cov measures correlation between two random variables. The two last measures are rank correlation measures and inspired by the Kendall's tau coefficient that considers the number of pairwise disagreements between two lists. Our rank correlation coefficient counts an agreement between the frequencies of two terms for each time period where both terms experience an increase or decrease in frequency without taking into consideration the absolute values. The rank correlation is normalized by the total number of time periods. The normalized rank correlation considers the same agreements but is normalized with the total number of time periods where both terms have a non-zero term frequency.

Filtering Co-references using Pearson's Correlation The first filtering makes use of the correlation measure to determine which co-references are related to the query term and filter out the rest. This measure is used by Radinsky et al. (2011) to measure similarity between terms and serves as a comparison for our filtering mechanisms. We keep a co-reference if its correlation to the query term exceeds the threshold corr_{min}. An increase in the filtering threshold would lead to the same or decreased recall while the precision could be affected either way. A decrease in the threshold would lead to a lower precision. Therefore a low threshold is sufficient to get an upper bound of the recall while maintaining precision.

Filtering Co-references using Document Frequency The second filtering is based on the document frequencies (df) of co-references. We filter out all co-references that differ largely in document frequency from the document frequency of the query term. The filtering depends on the document frequency of the most frequent term in the dictionary corresponding to a change period ($\text{df}_{\texttt{max}}$), the document frequency of the query term ($\text{df}_{\texttt{query}}$) and a *scaling factor* (sc). We filter out all co-references that have a document frequency $\text{df} \geqslant \text{df}_{\texttt{query}} \cdot \text{sc}(\text{df}_{\texttt{max}})$, i.e., which are frequently used in different contexts.

Filtering Co-references using Machine Learning Our third and final filtering method is based on machine learning. We use a random forest classifier (Breiman, 2001) consisting of a combination of decision trees where features are randomly extracted to build each decision

tree. In total ten trees with three features each are constructed. We choose features from the similarity measures presented above. This means that for each term–co-reference pair (w , w_c) found by NEER we calculate the $corr$, cov, rc and nrc measures. We also use the average of all four measures as a fifth feature. Finally we classify the pair as either 1 for w_c being a correct co-reference of w or 0 otherwise.

6 Experiments

The aim of our experiments was to measure how well NEER can detect names used during different time periods to refer to the same entity. We did this by (1) investigating how well burst detection can be used to capture change periods, and (2) measuring precision and recall of the co-references found using NEER. Each experiment in (2) was performed using two settings: (a) the first made use of the known change periods (denoted **known periods**), (b) the second used the detected bursts (denoted **found periods**). We used the known change periods to measure how well the method works assuming that we can find the correct change periods.

As there are no available baselines to compare our methods to, we defined our own baseline and named it **co-occurrence**. This considers all terms that co-occur with the queried named entity within a sliding window for all change periods, for (a) and (b) separately. This provided a baseline that shows what can be achieved with minimal computational effort.

We considered $precision = \frac{\text{\# correctly captured co-references}}{\text{\# all captured co-references}}$ and $recall = \frac{\text{\# captured co-references}}{\text{\# known name changes}}$ for our evaluation. For a term we required all direct co-references and at least one indirect co-reference for each name change to achieve full recall. That means that for *Joseph Ratzinger* we required all direct forms {*Cardinal Joseph Ratzinger, Cardinal Ratzinger*} but only one of the indirect {*Pope Benedict XVI, Pope Benedict*} to achieve a recall of 100%.

6.1 Experimental Setup

Dataset and Test Set For our experiments we used the New York Times Annotated Corpus (NYTimes). The dataset contains around 1.8 million articles published between 1987 and 2007.

We devised a test set of named entities, based on Kanhabua and Nørvåg (2010), with direct as well as indirect co-references and divided them into three categories: People, Locations and Companies. We identified all relevant name changes and the year in which they occur. Each co-reference pair was verified using three judges and kept if at least two judges agreed. If the change occurs in January of a year also the previous year was added. Change periods are available in the released test set. We mirrored all the entities so that *Pope Benedict* → *Cardinal Ratzinger* and *Cardinal Ratzinger* → *Pope Benedict* both exist as separate entries.

The final test set was devised by keeping all terms that (1) exist in the NYTimes, (2) have a change period in the NYTimes time span and (3) occur at least 5 times in at least one change period. The dataset is available in Tahmasebi et al. (2012). We started with 75 distinct names and 294 co-references. After filtering there were 16 distinct entities corresponding to 33 names and 86 co-references (44 indirect and 42 direct).

Setup We used the NYTimes API to extract documents from the NYTimes corpus. To extract terms we used Lingua English Tagger (Coburn, 2008) for finding single and multi-token nouns and the Stanford Named Entity Recognizer (NER, Finkel et al., 2005) to extract named entities. NERs typically consider names but not the role as a part of the name. For example *Barack Obama* is extracted but not *Senator Barack Obama*. Therefore we used the Lingua tagger which

	Found periods			Known periods		
Method	**Precision**	**Recall**	**avr # co-ref**	**Precision**	**Recall**	**avr #co-ref**
co-occurrence	8%	51%	120	20%	59%	16
NEER	8%	90%	128	13%	89%	64
NEER + Corr	20 %	61%	107	17%	74%	43
NEER + DF	33%	86%	28	50%	81%	10
NEER + ML	91 %	81%	6	94%	92%	5

TABLE 2: Precision and recall for the baseline and the different filtering techniques.

recognizes also terms like *Senator Barack Obama*. Named entities recognized by both methods are counted twice and thus receive a higher frequency. This procedure helps to choose good representatives for the co-reference classes.

In order to increase precision we filtered out infrequent terms. During graph creation we required a term to occur at least three times in the collection used for creating the graphs. If the most common terms in the dictionary occured more than 800 times, we required at least five occurrences. For finding direct co-references we required that each term occur at least five times. However, if the most common term in the dictionary occured more than 3000 times we increased the threshold to 10 occurrences. We also filtered out terms containing lowercase tokens. For this reason the term *Union of Myanmar* could not be found by the system.

For the relatedness calculations we used normalized term frequencies that are calculated as the fraction of term occurrences in all documents published per month divided by the total number of tokens in these documents.

To find the bursts we used the Java implementation from CIShell (Alencar, 2012) with 3 burst states, a transition probability γ of 0.8 and a density of 1.9. Using these parameters we detected on average 3.2 bursts for each term in our test set.

6.2 Results

Burst Detection We approximated change periods using burst detection. Considering all found bursts for an entity we were able to capture 73% of all change periods. This indicates that burst detection works well for capturing change periods but that there is room for improvement and parameter tuning. To reduce complexity and false positives, we limited the number of bursts to $\texttt{topB} = 6$. Using the $\texttt{topB}$ bursts we were able to capture 66% of all change periods.

If a name is ambiguous the bursts are less suitable for capturing correct change periods as the burst detection algorithm cannot distinguish between entities. This is the case for *George Bush* where the top bursts are 1988, 1989 and 1990 and correspond to *George Bush Senior*. *George W Bush* is not ambiguous and the found bursts are 1999, 2000 and 2001.

The results for the named entity evolution detection presented next are summarized in Table 2.

Baseline – Co-occurrence For comparison we chose a baseline consisting of all terms that co-occur with the query term in the datasets corresponding to the known and found burst periods. This naive baseline serves as a lower bound on recall. We used precision and recall for direct and indirect co-references found by our method and the corresponding entries from the test set. In Table 2 we find that the recall for the baseline is 59% using known periods and 51% using found periods. When considering the co-occurrences for the query term very few

or no direct co-references were found for the terms. Instead most indirect co-references were found. The precision differs largely between known and found periods; for the latter precision is surprisingly high with 20% compared to the found bursts (8%). This shows that the known periods help to find better co-references without introducing too much noise.

To mimick other methods where the user chooses a target time, we randomly chose three years per query term (corresponding to the average number of bursts per term) and created co-occurrence graphs for these years. We repeated the experiment three times and got an average recall of 36% which is statically significantly lower than the recall for known burst showing the power of using change periods for finding temporal co-references. As a comparison, a baseline method that chooses all terms that have lexical overlaps with the query term, can maximally achieve a recall of 49% ($= 42/86$) because no indirect co-references can be found.

NEER In this experiment we kept all co-references found by NEER without any filtering. This experiment provides an upper bound on the recall for the subsequent experiments. We found that for known periods as well as for found periods recall is high with 89% and 90% respectively. Only very few true co-references were missed and we found at least one correct co-reference for all but two terms. The precision is much lower for known bursts in comparison to the baseline which is a consequence of the higher average number of co-references found. However the recall is statistically significantly higher. The precision of the found bursts is comparable to the baseline and again the recall is significantly higher.

Out of 22 terms with indirect co-references our method was able to find at least one indirect co-reference for 21 terms for both known burst and found bursts. For found bursts no indirect co-reference could be found for *Airtran* because no bursts could be detected. For known bursts no indirect co-references could be found for *Andersen Consulting*.

Some sample queries and their five most frequent direct and indirect co-references for known bursts are shown in Table 3. As we can see the results contain co-references of high quality. For *Vladimir Putin* NEER found four roles *President-elect, Minister, Acting President* and *President*. For *Sean Combs* all but one of his names are present in the top five co-references, missing is only *Puff Daddy* which appears with a lower frequency. *Sean Combs Ruiz* is an error caused by the term extraction. The term *Ruiz* is a name of a movie character that was played by Sean Combs in 2001.

Considering names with only a single token typically decreases the precision for people because it increase the number of co-occurrences with first names. However, for companies and locations it is necessary to keep single token names as otherwise many names would be missed, e.g. *Burma*. To further improve results an extension to NEER could classify names into different categories. The extended NEER can then keep or discard single token names accordingly as well as allow different name patterns such as names with non-capital tokens (e.g., *Union of Myanmar*).

NEER + Correlation filtering Using correlation as a filtering, with $\text{corr}_{\text{min}} = 0.4$, precision increased over the NEER results while recall decreased. For both known and found periods the decrease in recall corresponds to a statistically significant decrease. The recall is higher than that of the baselines but is not competitive to the NEER results and shows that global correlation is not an appropriate similarity measure for temporal co-references.

Barack Obama	Vladimir Putin	Sean Combs
Senator	President-elect Vladimir V Putin	Puffy
State Senator Barack Obama	Minister Vladimir Putin	Sean John
Senator-elect Barack Obama	Acting President Vladimir V Putin	Diddy
Senator Barack Obama	President Vladimir V Putin	Sean Combs Ruiz
Illinois Democrat	Vladimirovich	Sean John Combs

TABLE 3: Terms and their top temporal co-references. Gray cells are considered incorrect.

NEER + Document Frequency filtering In this experiment co-references were removed if they occured in more documents than the query term times a scaling factor (sc), as described in Section 5.4. The filtering provides a good recall for both found and known periods. The decrease in recall compared to the NEER results is not statistically significant for either found nor known periods. With regards to precision both found and known periods show the most competitive performance compared to the baseline and the NEER results.

We used sc $= 10$ for $\mathrm{df}_{\mathtt{max}} \leqslant 300$, 5 for $\mathrm{df}_{\mathtt{max}} \leqslant 800$ and 3 for $\mathrm{df}_{\mathtt{max}} > 800$. These filtering thresholds as well as the scaling factors were found empirically. Learning these could improve the results for document filtering further.

NEER + Machine Learning We showed that unsupervised filtering can perform well for filtering out erroneous co-references found by NEER. In this experiment we investigated if the results could be further improved by using a limited amount of supervision for training a classifier. We used WEKA (Hall et al., 2009) and the random forest classifier. We trained our classifier on the dataset and used 15 fold stratified cross fold validation to determine precision and recall. We classified each term–co-reference pair produced by NEER. For known burst we got in total 2963 instances where 170 were correct co-references (we accepted combinations of correct names, e.g., *Sean Diddy Combs*). Using the classifier we were able to achieve a 94% precision and only 11 false co-references were classified as correct. The recall of the filter is 92%, however, it was only applied to the output of the NEER. For the known bursts this corresponds to a true recall of $92\% \cdot 89\% = 82\%$. This shows the true potential of the machine learning approach and the features chosen for the classification (see relatedness measures, Section 5.4). Assuming that the NEER results can be improved further the classification is a powerful tool.

For the found bursts there were 7048 instances with 204 correct co-references. The precision of 92% is comparable to that of the known bursts and only 16 false co-references were classified as true co-references. The recall is 81% compared to 92%, likely a consequence of the ratio between the two classes with 204 vs 6844 instances. Adding learning instances could help boost the results, e.g., adding the entire test set to the training set could further improve the recall.

7 Discussion

Our experiments show that we are able to find temporal co-references with high recall without depending on external knowledge sources. We found that, even though not all change periods could be found using burst detection (recall 66%), we still managed to get a recall that is comparable to the high recall for the known (correct) change periods. Because every change period captures two co-references, it is possible to capture more co-references than the number of found change periods suggests.

There can be several reasons for a change period not to occur as a burst. In some cases the name change is discussed before the change takes place and thus there is a discrepancy between

Accenture	Comcast	Czech Republic	Myanmar
Accenture Match Play Championship	Comcast Corporation	Hungary	Burma
Andersen Consulting	AT&T Comcast	Slovakia	

TABLE 4: Top co-references from the categories *Location* and *Company* after document frequency filtering for known bursts. For *Myanmar* only *Burma* remains after filtering.

the found burst and the 'true' change period. It is also possible for a name change to correspond to a smaller increase in frequency than other events (possibly such leading up to and causing the name change). Future work is to learn thresholds to find bursts that correspond to change periods or find other methods better suited for change period detection.

We found that co-references cannot be detected in a symmetric way: Finding w_i as a co-reference for w_j does not imply that we can find w_j as a co-reference for w_i. E.g., NEER found *Slovakia* and *Czech Republic* as co-references for *Czechoslovakia*. However, for *Czech Republic* NEER could not find *Czechoslovakia* (using found or known bursts).

Our experiments show that co-references found for terms from the category *People* have a good accuracy among the top co-references also without filtering. However, for the category *Locations* and *Companies* filtering is needed for achieving high accuracy among the top results. Some examples for companies and locations can be found in Table 4. For *Accenture*, *Czech Republic* and *Myanmar* we found an indirect co-reference among the top two terms.

By making use of terms from the dataset we ensure that all found temporal co-references can be used for information retrieval on the dataset. The results of Kanhabua and Nørvåg (2010) contain co-references of high quality like *Senator Barack H. Obama Jr*, but many of these do not appear in the NYTimes and thus cannot be used to retrieve documents. By not relying on external resources we enable a robust method that can be applied on any corpus and finds ephemeral co-references like *President-elect George Bush* or *Senator-elect Barack Obama*. NEER can also be applied to heterogeneous data such as long-term archives as well as web data and can mix content from several sources.

Conclusions and Future Work

In this paper we presented NEER, an unsupervised approach for named entity evolution recognition that overcomes limitations of existing approaches and does not depend on external knowledge sources. We made use of change periods to create term contexts to capture co-references in the same context, thereby avoiding to compare term contexts from vastly different periods in time. Burst detection was used to detect change periods and captured 66% of all change periods. Because term contexts created in change periods capture more than one co-reference, 90% of all name changes were found. We used frequency analysis to find direct and indirect co-references by filtering on document frequency as well as using machine learning to classify correct co-references. Using a random forest classifier we achieved a precision of 94% on known periods and 91% on found periods. All name changes used in our test set are released to encourage further research in this area (Tahmasebi et al., 2012).

In this paper we focused on change periods for one term and searched for temporal co-references. This means that for terms with indirect co-references, we can find at most one change at a time. In future work we will focus on automatically creating chains of evolution to handle terms with many changes and to associate validity period to each co-references, e.g.,

$$\textit{ipod} \xrightarrow[2012-2001]{} \textit{mp3 player} \xrightarrow[2001-1996]{} \textit{minidisc} \xrightarrow[1996-1992]{} \textit{discman} \xrightarrow[1984-1979]{} \textit{walkman}.$$

References

Alencar, A. (2012). CIShell Burst Detection. Retrieved 2012-11-05, from `http://wiki.cns.iu.edu/display/CISHELL/Burst+Detection`.

Berberich, K., Bedathur, S. J., Sozio, M., and Weikum, G. (2009). Bridging the terminology gap in web archive search. In *WebDB*.

Breiman, L. (2001). Random forests. In *Machine Learning*, pages 5–32.

Coburn, A. (2008). Lingua::EN::Tagger – Part-of-speech tagger for English natural language processing. Accessed 2012-11-05, from `http://search.cpan.org/perldoc?Lingua::EN::Tagger`.

Ernst-Gerlach, A. and Fuhr, N. (2007). Retrieval in text collections with historic spelling using linguistic and spelling variants. In *JCDL*, pages 333–341.

Finkel, J. R., Grenager, T., and Manning, C. (2005). Incorporating non-local information into information extraction systems by Gibbs sampling. In *ACL*, pages 363–370.

Gotscharek, A., Neumann, A., Reffle, U., Ringlstetter, C., and Schulz, K. U. (2009). Enabling information retrieval on historical document collections: the role of matching procedures and special lexica. In *Workshop on Analytics for Noisy Unstructured Text Data*, AND '09, pages 69–76.

Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., and Witten, I. H. (2009). The weka data mining software: an update. *SIGKDD Explor. Newsl.*, 11(1):10–18.

Hauser, A., Heller, M., Leiss, E., Schulz, K. U., and Wanzeck, C. (2007). Information access to historical documents from the early new high german period. In *Digital Historical Corpora – Architecture, Annotation, and Retrieval*, number 06491 in Dagstuhl Seminar Proceedings.

Ioannou, E., Nejdl, W., Niederée, C., and Velegrakis, Y. (2010). On-the-fly entity-aware query processing in the presence of linkage. *PVLDB*, 3(1):429–438.

Kaluarachchi, A. C., Varde, A. S., Bedathur, S. J., Weikum, G., Peng, J., and Feldman, A. (2010). Incorporating terminology evolution for query translation in text retrieval with association rules. In *CIKM*, pages 1789–1792.

Kanhabua, N. and Nørvåg, K. (2010). Exploiting time-based synonyms in searching document archives. In *JCDL*, pages 79–88.

Kleinberg, J. (2003). Bursty and hierarchical structure in streams. *Data Mining and Knowledge Discovery*, 7(4):373–397.

Radinsky, K., Agichtein, E., Gabrilovich, E., and Markovitch, S. (2011). A word at a time: computing word relatedness using temporal semantic analysis. In *WWW*, pages 337–346.

Shen, W., Wang, J., Luo, P., and Wang, M. (2012). Linden: linking named entities with knowledge base via semantic knowledge. In *WWW*, pages 449–458.

Tahmasebi, N., Gossen, G., Kanhabua, N., Holzmann, H., and Risse, T. (2012). Named entitiy evolution dataset. Available online at `http://l3s.de/neer-dataset/`.

Tahmasebi, N., Risse, T., and Dietze, S. (2011). Towards automatic language evolution tracking, a study on word sense tracking. In *Workshop on Knowledge Evolution and Ontology Dynamics (EvoDyn 2011), co-located with ISWC 2011*.

Weisstein, E. W. (2012a). Correlation coefficient. Retrieved 2012-11-05, from `http://mathworld.wolfram.com/CorrelationCoefficient.html`.

Weisstein, E. W. (2012b). Covariance. Retrieved 2012-11-05, from `http://mathworld.wolfram.com/Covariance.html`.

Evaluating the translation accuracy of a novel language-independent MT methodology

George TAMBOURATZIS, Sokratis SOFIANOPOULOS, Marina VASSILIOU
Institute for Language & Speech Processing, Athena R.C.
6 Artemidos & Epidavrou Str., Paradissos Amaroussiou, 151 25 Athens, Greece
giorg_t@ilsp.gr,s_sofian@ilsp.gr,mvas@ilsp.gr

ABSTRACT

The current paper evaluates the performance of the PRESEMT methodology, which facilitates the creation of machine translation (MT) systems for different language pairs. This methodology aims to develop a hybrid MT system that extracts translation information from large, predominantly monolingual corpora, using pattern recognition techniques. PRESEMT has been designed to have the lowest possible requirements on specialised resources and tools, given that for many languages (especially less widely used ones) only limited linguistic resources are available. In PRESEMT, the main translation process is divided into two phases, the first determining the overall structure of a target language (TL) sentence, and the second disambiguating between alternative translations for words or phrases and establishing local word order. This paper describes the latest version of the system and evaluates its translation accuracy, while also benchmarking the PRESEMT performance by comparing it with other established MT systems using objective measures.

KEYWORDS: hybrid machine translation; language-independent methodology; MT evaluation

Proceedings of COLING 2012: Technical Papers, pages 2569–2584,
COLING 2012, Mumbai, December 2012.

1 Introduction

The Machine Translation (MT) task has been studied for a number of decades, but still remains to a large extent an issue unresolved, as the performance delivered by the best current systems still falls short of the required quality. Since the number of texts available over the World Wide Web is ever increasing, and these texts may be written in one of several hundred languages, the requirement for automatically performing translation of an acceptable quality remains a prime objective. A number of MT paradigms have been proposed, the main ones including Rule-Based MT (RBMT), Statistical MT (SMT) and Example-Based MT (EBMT). Furthermore, the requirement for covering an ever increasing combination of Source to Target language (SL to TL) combinations necessitates the development of language-independent methodologies.

Currently most MT approaches are based on the SMT paradigm (Koehn, 2010). SMT uses dedicated algorithms that do not employ language-specific rules and is thus portable to new language pairs, provided the necessary training data are available. The main SMT constraint is the need for SL-TL bilingual corpora of a sufficient size (of the order of a million parallel sentences) to allow the building of accurate translation models. Such corpora are hard to obtain, particularly when less widely-used languages are involved. Besides, the process of compiling and verifying such corpora is expensive in terms of both manpower and time.

In EBMT, translations are generated by analogy, where the system has available a set of known pairs of input sentence (in SL) and corresponding translation (in TL). Then, each new input sentence is broken down to non-overlapping phrases, which are translated using the translation examples as a reference. The translated sentence is finally composed by combining the translated phrases.

Another paradigm is hybrid MT, which combines ideas and techniques from more than one approaches, like for example EBMT and SMT techniques (cf. Groves & Way, 2005 and Phillips, 2011). Such approaches have been proposed for creating MT systems using more limited but easily obtainable resources. Even if these methods do not achieve accuracy as high as that of SMT systems, their ability to develop MT systems with limited resources is an advantage in the case of less-widely used languages. The PRESEMT system is based on such a methodology, as detailed below, its main characteristics being the use of only very small bilingual corpora and the employment of large monolingual corpora for extracting most of the necessary linguistic information.

A number of methods for the automatic inference of templates for the structural transfer from SL to TL have been proposed. For instance, Sanchez-Martinez et al. (2009) suggest using small parallel corpora only to extract transfer rules, assuming that a sufficient bilingual dictionary is already available. Carbonell et al. (2006) propose an MT method that requires no parallel corpora, but relies on a translation model utilising a full-form bilingual dictionary and a decoder using long-range context via large n-grams.

Another family of systems are METIS (Dologlou et al., 2003) and METIS-II (Markantonatou et al., 2009), both of which rely solely on extensive monolingual resources in order to generate translations employing pattern recognition-based algorithms. The METIS family represents the ancestor of PRESEMT, which has built

upon the past experience, by (i) adding a small bilingual corpus to improve translation accuracy and (ii) using more advanced algorithms for pattern matching to provide a measurable increase in both speed and accuracy of the generated translations.

2 The principles of the PRESEMT system

In terms of resources, similarly to METIS-II, PRESEMT uses a bilingual dictionary providing SL – TL lexical correspondences and an extensive TL monolingual corpus collected automatically from the web. A small bilingual corpus containing parallel sentences is added in PRESEMT, in order to (a) reduce the number of possible translations that need to be evaluated by the system and (b) define examples of SL – TL structural modifications, thus improving the translation quality. The bilingual corpus need not cover a particular domain and only numbers a few hundred sentences (typically ~200) for determining structural equivalences between sentences in the source and target languages. Hence, in comparison to SMT systems, the size of the parallel corpus required is reduced by more than three orders of magnitude. Evidently, for a bilingual corpus of only a few hundred sentences, not all linguistic phenomena are likely to occur. However, it is expected that the most frequent ones will be covered and thus a sufficient coverage of the structure transformations from SL to TL can be achieved.

Both the bilingual and the monolingual corpora are annotated[1] with lemma and Part-of-Speech (PoS) information and, depending on the language, with additional morphological features (e.g. case, number, tense etc.). Furthermore, they are segmented into non-recursive syntactic phrases (e.g. noun phrase, verb phrase etc.). The next subsections describe the kind of information extracted.

2.1 Processing the bilingual corpus

The processing of the bilingual corpus involves the use of a pair of modules, namely the Phrase aligner module (PAM) and the Phrasing model generator (PMG). PAM operates on the bilingual corpus to achieve the establishment of matching phrasing schemes in the SL and TL sides. This is achieved by aligning the bilingual sentences initially at a word level and then porting these alignments at a phrase level. PAM aims at identifying how the SL structure is modified towards the TL one, allowing the deduction of a phrasing model for the source language. During initialisation, PAM takes as input a parsed text in the TL-side of the parallel corpus, via a chosen TL parser. It is assumed that in this corpus there is a high level of fidelity between the SL-side and TL-side, which extends to the phrasing schemes of the two languages. Then, PAM algorithmically segments the SL-side sentence into phrases in accordance to the TL side. To achieve that, PAM takes into account alignment information in the form of (a) lexicon-based correspondences, (b) alignment on the basis of grammatical feature similarity and PoS tag correspondence and (c) alignment information provided by already aligned neighbouring words in the SL and TL sides. Within this sequence, in each consecutive step additional SL words are aligned to TL words, the aim being for all words to be assigned to SL phrases that correspond to the TL phrasing, these phrases then being mapped to the TL phrases.

[1] For the annotation task readily available tools are employed, including statistical taggers and (to some extent) chunkers that provide shallow parsing. This alleviates the need for developing new linguistic tools.

The SL side of the aligned corpus is subsequently processed by PMG, with a two-fold purpose, namely to (i) deduce a phrasing model based on conditional random fields (CRF) (Lafferty et al., 2001) and (ii) employ this model for parsing any SL text submitted for translation. During the derivation of a phrasing model, the SL side of the aligned bilingual corpus is used to train a CRF model via a standard iterative process. During operation, this model is used to segment new sentences to be translated into their constituent phrases. Details on the algorithmic design and individual accuracy of PAM and PMG are provided in Tambouratzis et al. (2012a).

2.2 Extracting information from the monolingual corpus

The TL monolingual corpus is processed to create two distinct models, which are employed during the translation process. The first model is used solely for disambiguation purposes, when two or more translations are proposed for a word or set of words. In this account, different models have been studied, including a SOM-based model (Tsimboukakis et al., 2011 and Tambouratzis et al., 2012b) and an n-gram-based model. The second one is a phrase model that provides the micro-structural information on the translation output, to determine intra-phrasal word order. The model is stored in a file structure, where a separate file is created for phrases according to their (i) phrase type, (ii) phrase head and (iii) phrase head PoS tag. As most of the progress has involved developments in the second model, this is the one discussed in more detail in the remainder of the present section.

The number of files created as a result of this process is very large (of the order of millions of files), as for each combination of the three aforementioned criteria, a different file needs to be created; yet each of the files is of a small size and thus can be retrieved and loaded quickly. Due to the very large number of files, the actual data structure and implementation becomes very important. Currently PRESEMT uses a simple string representation to store the phrases in each file, ordered by their frequencies of occurrence. Initially the phrases were stored as serializable objects in hash tables, based on their order of appearance in the corpus. This redesigned model occupies substantially less disk space and provides faster retrieval. Also the ordered storage of phrases provides the algorithm a way to stop the search as soon as a relevant phrase has been retrieved. On the whole, the aforementioned revisions in the modelling of the phrases have led in a reduction in the translation time of approximately 40%, when averaged over a set of 200 sentences being translated (to avoid bias due to sentence-specific phenomena. Regarding the disk requirements, the use of the revised mapping has resulted in a substantial drop in the required space for storing the model (for a corpus of 80 Gbytes, the model size has been reduced by approximately 58%, from 22 Gbytes to 9.3 Gbytes).

2.3 Main translation engine

The translation process is split into two phases. Phase 1 (**Structure selection**) uses the bilingual corpus to determine, for a given input SL sentence, the appropriate TL structure in terms of the sequence of phrases and their order. The output of the Structure selection phase is the SL sentence with a TL structure, created by reordering the phrases according to the archetypes contained in the parallel corpus, and all words replaced by the TL lemmas and tag information as retrieved from the bilingual dictionary.

Phase 2 (**Translation equivalent selection**) uses the models extracted from the TL monolingual corpus as described in section 2 so as to specify the most likely word order within phrases, to handle functional words such as articles and prepositions and to resolve lexical ambiguities emerging from the possible translations provided by the bilingual dictionary. Finally, a token generator component generates tokens out of lemmas. Therefore, the first PRESEMT translation phase is closely related to EBMT, while the second phase is reliant upon information of a statistical nature (but extracted from monolingual corpora), resulting in a hybrid nature.

3 Phase 1: Structure selection

The task of Structure selection is to determine for each input sentence the type of TL phrases to which the SL ones translate and to order them in the TL sentence. To this end it consults the patterns of SL – TL structural modifications to be found in the parallel corpus, thus resembling EBMT (Hutchins, 2005).

Translation phase 1 receives as input an SL sentence (termed **ISS** – Input Source Sentence), bearing lexical translations from the dictionary, annotated with tag and lemma information and segmented into phrases by PMG. A dynamic programming algorithm is applied to determine for each ISS the most similar, in terms of phrase structure, SL sentence found in the bilingual corpus (termed **ACS** – Aligned Corpus Sentence)[2].

The similarity is determined on the basis of structural information such as phrase type, phrase head PoS tag, phrase functional head info and phrase head case. The phrases within ISS are reordered in accordance to the TL side of the chosen ACS by replicating the SL-TL phrase alignment mapping. The dynamic programming algorithm evaluates the similarity in the SL language. The most similar SL structure of the bilingual corpus, that determines the TL structure of the sentence to be translated, is thus selected purely on SL properties. The implemented method is based on the Smith-Waterman algorithm (Smith and Waterman, 1981), initially proposed for aligning DNA and RNA sequences. This algorithm is guaranteed to find the optimal local alignment between two input sequences.

The structural similarity between ISS and ACS is reflected on the similarity score, for the calculation of which a two-dimensional matrix is created with the ISS phrases along the top row and the ACS along the left side. As is standard practice in Dynamic Time Warping, movement across this matrix is from the top left corner towards the bottom right-hand side. The similarity for cell (i,j) is determined by examining the predecessor cells located directly to the left $(i, j-1)$, directly above $(i-1, j)$ and above-left $(i-1, j-1)$,, and is calculated iteratively as the maximum of the three similarities. The similarity of two phrases results by the weighted sum of the similarities of (a) the phrase type, (b) the phrase head PoS tag, (c) the phrase head case and (d) the functional phrase head PoS tag.

The similarity score ranges from *100* to *0*, these limits denoting respectively exact match and total dissimilarity between elements of ISS and ACS. In case of a zero similarity score,

[2] If the most similar ACS retrieved from the parallel corpus is very dissimilar, then ISS does not undergo any reordering. It is notable that in our experiments never did such an occasion appear, the similarity always reaching a high percentage (above 70%). The fact that comparisons involve sentences of the same language (SL) contributes to a high similarity score.

a penalty weight (-50) is employed, to further penalise the establishment of a mapping between dissimilar items.

After calculating the final similarity score between sentences, the comparison matrix indicates the optimal phrase alignment between the two SL sentences. By combining the SL sentence alignment from the algorithm with the alignment information between the ACS and the attached TL sentence, ISS phrases are reordered accordingly.

4 Phase 2: Translation equivalent selection

Following the completion of Phase 1, remaining translation issues include (i) establishing word order within phrases, (ii) handling functional words and (iii) resolving translation ambiguities. To establish the correct word order, the monolingual TL corpus is searched to determine the most similar phrase to each phrase in the SL sentence. The similarity measure takes into account the phrase type and the words contained in terms of lemma, PoS tag and morphological features. These factors enter the comparison with different weights, the relative magnitudes of which are the subject of an optimisation process.

The main issue at this stage is word reordering within each phrase. This entails that the words of a given phrase of the input sentence (denoted as **ISP** – Input Sentence Phrase) and the words of a retrieved TL phrase (denoted as **MCP** – Monolingual Corpus Phrase) are close to each other in terms of number of words and type.

When initiating Phase 2 of the translation process, the matching algorithm accesses the indexed TL phrase corpus (created as described in section 2) to retrieve similar phrases and select the most similar one through a comparison process, which is viewed as an assignment problem. This problem can be solved via both exact algorithms that guarantee the identification of the optimal solution and sub-optimal ones. Experiments when developing METIS-II have shown that the solution of the assignment problem is computationally-intensive. Consequently, to conform to the strict translation time constraints set for PRESEMT, the Gale-Shapley algorithm is used (Gale and Shapley, 1962 and Mairson, 1992), which solves the assignment problem in a reduced time. This process is possibly non-optimal but allows a substantial reduction in the computation time.

After the completion of this comparison process, the selected phrase from the monolingual corpus serves as a basis for resolving other issues such as the handling of functional words (e.g. insertion / deletion of articles). In this process, the TL information prevails over the SL entries, based on initial experiments performed, to provide a translation closer to the TL-provided information.

Translation equivalent selection receives as input the output of Structure selection, which contains sets of candidate translations for each SL lemma. One translation needs to be chosen from each set, thus disambiguating amongst the possible translations. The disambiguation process uses the semantic similarities between words as evidenced by the monolingual corpus. Different approaches are evaluated within PRESEMT for selecting the most appropriate translation, including Vector Space Modelling (Marsi et al., 2010) and Self-Organising Maps, following the work by Tsimboukakis et al. (2011).

Rather than employing these disambiguation processes, a simpler, corpus-based approach is proposed in the PRESEMT configuration discussed here, which relies on the

extraction of statistical information with only limited pre-processing. This method reuses and enhances the indexed sets of the monolingual corpus phrases, by exploiting information on the frequency of occurrence of each TL phrase. When searching for the best matching TL phrase for each combination of lexical alternatives, the frequency of the TL phrase is taken into account. Notably, not all combinations are examined for lexical disambiguation; instead only the phrase mapped to the most frequent TL phrase is retained. A formula is used for selecting the most appropriate phrase based on both the similarity score and the frequency of the TL phrase. This formula ensures that even though one TL might achieve a higher comparison score than another, if its frequency is significantly lower, then the second phrase - which has a lower absolute score - will be selected, due to its substantially higher frequency of occurrence. This enables the algorithm to delete or add a word such as an article in the final translation of the phrase. This scoring mechanism can be easier to understand with a Greek to English translation example where the article in the Greek phrase needs to be removed from the English translation, for instance the Greek noun phrase '*Η Γαλλία*', which translates to "*France*" in English. When searching for relevant phrases in the TL model, the phrase "*the France*" scores *100* and appears in the corpus *34* times, while the phrase "*France*" scores *85* and appears *5,030* times. Using the aforementioned method and a threshold value of the score ratio being equal to 90% in this case, the ratio between the two scores is not high enough, so the selection will be based on the frequency of occurrence ratio between the two phrases, where the correct phrase (the one without the article) has a substantially higher number of occurrences in the corpus.

5 Example of the PRESEMT translation process

In this section a simple example is used to illustrate the translation process of the PRESEMT system in a step-wise manner. An SL sentence as the one in (1) is being input for translation:

(1) Εδραιώνονται σχέσεις καλής γειτονίας στις χώρες των Βαλκανίων
"Good neighbourhood relations are established in the Balkan countries"

Annotation at various levels [tagging & lemmatising; PMG-based segmentation to phrases (VC: verb chunk, PC: prepositional chunk); output of the lexicon look-up]

SL sentence annotated after being input for translation			
Phrase	**VC**	**PC**	**PC**
Word	εδραιώνονται	σχέσεις καλής γειτονίας	στις χώρες των Βαλκανίων
Lemma	εδραιώνω	σχέση, καλός, γειτονία	στου, χώρα, ο, Βαλκάνια
Tag	**vb**o3pl	**no**feplnm, **aj**fesgge, **no**fesgge	**as**feplac, **no**feplac, **at**neplge, **no**neplge
Lexicon	{consolidate; establish}	{relation; relationship} {nice; decent; good} {adjacency; neighbourhood}	{on; at; to; into; in; upon} {country} {the} {Balkan}

1st translation phase: Search the bilingual corpus for the most similar SL sentence in structural terms, find the corresponding TL one and reorder the input SL sentence on the basis of TL; output an intermediate result (2).

Most similar SL sentence of the bilingual parallel corpus			
Phrase	**VC**	**PC**	**PC**
Word	σημειώνονται	διαμαρτυρίες φοιτητών	σε άλλες χώρες της ΕΕ
Lemma	σημειώνω	διαμαρτυρία, φοιτητής	σε, άλλος, χώρα, ο, ΕΕ
Tag	**vb**o3pl	**no**feplnm, **no**maplge	**asppsp**, **pn**fe03plac, **no**feplac, **at**fesgge, **abbr**
Corresponding TL sentence of the bilingual parallel corpus			
Phrase	**VC**	**PC**	**PC**
Word	student protests	occur	in other EU countries
Lemma	student, protest	occur	in, other, EU, country
Tag	nn, nns	vv	in, jj, np, nns

(2) Output of the 1st translation phase (expressed as list of phrases and lemmas):

[PC {relation; relationship}; {nice; decent; good}; {adjacency; neighbourhood}]
[VC {consolidate; establish}]
[PC {on; at; to; into; in; upon}; {country}; {the}; {Balkan}]

2nd translation phase: Identify the correct word order within each phrase (3); disambiguate the translations (4); generate tokens out of lemmas (5); produce final translation (6).

(3) **Word reordering results:**

[PC {nice; decent; good}; {adjacency; neighbourhood}; {relation; relationship}]
[VC {consolidate; establish}]
[PC {on; at; to; into; in; upon}; {the}; {Balkan}; {country}]

(4) **Disambiguation results:**

[PC {good}; {neighbourhood}; {relation}]
[VC {establish}]
[PC {in}; {the}; {Balkan}; {country}]

(5) **Token generation:**

[PC {good}; { neighbourhood}; {relations }]
[VC {are established}]
[PC {in}; {the}; {Balkan}; {countries}]

(6) **Final translation:** Good neighbourhood relations are established in the Balkan countries

6 Experimental Results

The evaluation results reported here concern the Greek – English language pair [3] and are based on the development datasets used in PRESEMT for studying the system performance. For each SL, these datasets contain 1,000 sentences, collected via web-crawling. Sentence length ranges from 7 to 40 words.

[3] PRESEMT currently handles 8 language pairs: SL {Czech, English, German, Greek, Norwegian} – TL {English, German}.

From these datasets, 200 sentences were randomly chosen, and manually translated into each of the target languages. The correctness of these reference translations was checked independently by native speakers. For the current evaluation phase four automatic evaluation metrics have been employed, i.e. BLEU (Papineni et al., 2002), NIST (NIST 2002), Meteor (Denkowski and Lavie, 2011) and TER (Snover et al., 2006).

For the bilingual corpus, 200 sentences were used. The Greek-English dictionary contained a total of just over 40,000 entries. For PRESEMT, two versions are evaluated. The first one (PRESEMT-1) indicates the state of the system on April 2012 (i.e. after almost 28 months of development of the system, including the system specifications definition). PRESEMT-1 includes the basic configuration of the system as described in Sofianopoulos et al. (2012). PRESEMT-2 encompasses a number of improvements in comparison to PRESEMT-1, including refined algorithms for the two translation phases, an improved method of using the indexed monolingual corpus and later enhanced versions of the PAM/PMG modules (reflecting the current state in October 2012). Table 1 summarises indicative scores obtained together with scores achieved by four MT systems available online for the same set of data.

	BLEU	NIST	Meteor	TER
Google	0.5544	8.8051	0.4665	29.791
Systran	0.2930	6.4664	0.3830	49.721
WorldLingo	0.2659	5.9978	0.3666	50.627
Bing	0.4600	7.9409	0.4281	37.631
METIS-II	0.1222	3.1655	0.2698	82.8780
PRESEMT-1	0.1683	5.7389	0.3203	68.4670
PRESEMT-2	0.3011	6.6878	0.3733	54.5990

TABLE 1 – Comparison to other MT systems for the Greek-to-English language pair

In comparison to METIS-II[4], the latest version of PRESEMT offers a substantial improvement for all metrics, with for instance BLEU and NIST scores both being increased by more than 145%. This illustrates the improvements conferred by the new translation methodology as compared to the METIS-II family.

It is noteworthy that PRESEMT outperforms two of the other MT systems, Systran and WorldLingo, with scores increased by 2.7% and 13% respectively. As noted, PRESEMT is still under development and it is anticipated that more extensive experiments involving additional language pairs will provide improvements in the translation quality.

6.1 Detailed analysis of the evaluation results

In the present section, the aim is to visualise the evaluation results for the development set. In Figure 1 the BLEU results of the earlier PRESEMT prototype are indicated in a

[4] http://www.ilsp.gr/metis2/

scatter plot, as a function of the sentence size for the language pair Greek-to-English. It can be seen that, as the input sentence size increases in terms of words, the score shows a trend of reducing. Also, it can be noted that for most sentences, the BLUE score is less than 0.2, indicating a less than satisfactory translation.

The highest BLEU score of PRESEMT-1 is equal to 0.56 and is obtained for a relatively short sentence of 12 words, while for only a few medium to long sentences (of 15 words or more) is a BLEU score of 0.4 or more achieved. Finally, for sentences with length of 20 words or more the BLEU score rarely exceeds 0.2.

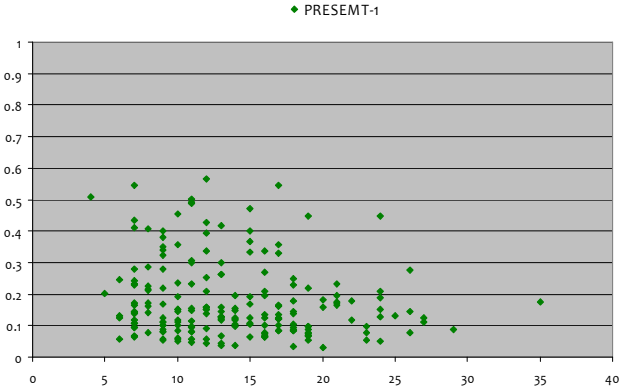


FIGURE 1 – Scatter plot of BLEU results for the EL-EN language pair (PRESEMT-1)

In Figure 2, the BLEU results for the PRESEMT-2 are indicated in a scatter plot, as a function of the sentence size for the language pair Greek-to-English. It is evident that the translation quality is improved, with BLEU scores exceeding 0.5 for a number of sentences. In addition, even for large input sentence sizes, relatively high BLEU scores are achieved (for instance, for the largest sentence of 35 words, a score of almost 0.6 is achieved). Furthermore, even for sentences of more than 25 words, the majority of translations approximate or exceed a score of 0.5, whilst when using PRESEMT-1 (cf. Figure 1) no sentences of this length manage a BLEU score exceeding 0.3.

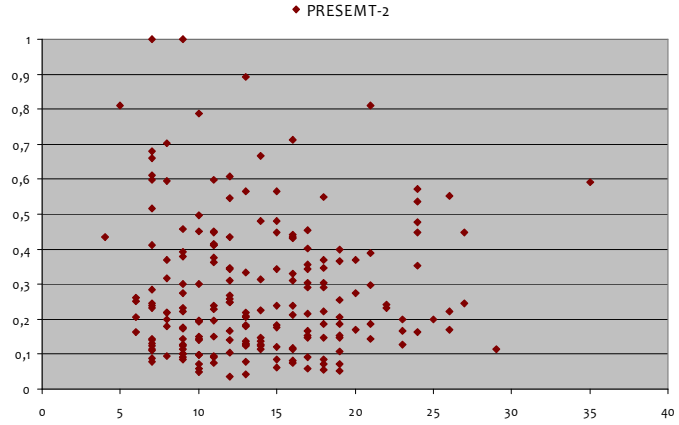


FIGURE 2 – Scatter plot of BLEU results for the EL-EN language pair (PRESEMT-2)

To perform a more systematic analysis, the different sentence sizes have been organised by defining bins, each of which spans 8 sentence sizes (i.e. the first bin concerns sentences of between 4 and 7 words, bin 2 comprises sentences between 8 and 10 words etc.). A boxplot diagram is used to indicate for each of the aforementioned bins the characteristics of BLEU scores, as shown in Figure 3 for PRESEMT-1 and in Figure 4 for PRESEMT-2.

By comparing the boxplots of the two PRESEMT versions for BLEU, it can be seen that boxplots for PRESEMT-1 occupy similar ranges of the score range to those of PRESEMT-2. However, the range for PRESEMT-1 is displaced towards lower values of BLEU in comparison to PRESEMT-2, while also a larger number of outliers exist for PRESEMT-1. Thus, most median values of PRESEMT-1 for different sentence sizes are placed at lower BLEU levels, below the 0.15 mark, with only a few outliers exceeding the limited range of the boxplots.

On the contrary, when turning to PRESEMT-2, the median values are higher, exceeding 0.200 in most cases and even reaching 0.400 in some of the cases. Besides, when comparing the median values, these are increased by 50% or more for most sentence sizes for PRESEMT-2 in comparison to PRESEMT-1. Also, for longer sentences (for instance bin6, which comprises sentences of 24 to 27 words), the improvement in BLEU score is substantial, increasing by a factor of approximately 2.5. This applies to the value corresponding to the 50% level (i.e. the median value) as well as to the levels of 25% and 75%.

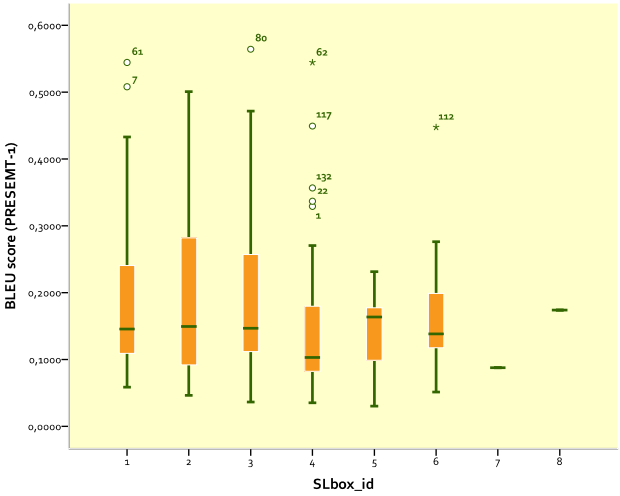


FIGURE 3 – Box plot of BLEU results for the EL-EN language pair (PRESEMT-1)

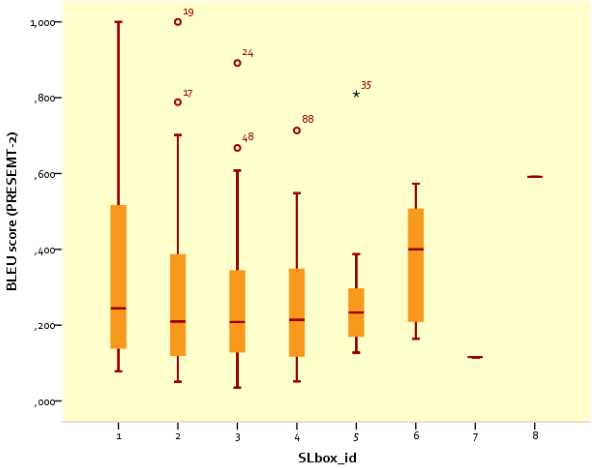


FIGURE 4 – Box plot of BLEU results for the EL-EN language pair (PRESEMT-2)

Furthermore, even though the y-axis scale is larger in Figure 4 than that of Figure 3 the population of solutions covers a much wider range and in several cases translations of a substantially higher quality are achieved. Though the variances are substantially higher for Figure 4 as compared to Figure 3, this is due to several sentences being translated much more accurately, thus reflecting a better translation performance. In addition, the boxplot outliers are fewer in the case of PRESEMT-2, while the variance does not appear to increase as the sentence size increases.

Finally, the BLEU score does not appear to reduce substantially as the sentence size increases, promising scalability of the PRESEMT system for more complex sentences (though this would need to be confirmed via more extensive experiments), with a dependable level of performance. This indicates that the algorithmic improvements integrated when transitioning from PRESEMT-1 to PRESEMT-2 result in a higher translation quality and also contribute to a more predictable performance.

Conclusions

In the present article the principles and the implementation of a novel language-independent methodology have been presented. The PRESEMT methodology draws on information residing in a large monolingual corpus and a small bilingual one for creating MT systems readily portable to new language pairs. Most of this information is extracted in an automated manner using pattern recognition techniques.

First experimental results and comparisons to established systems have been reported. These results are promising, especially taking into account the fact that several PRESEMT modules are still under development and the translation process is being refined, in particular with respect to the handling of internal phrasal structure. Initial studies of the PRESEMT translations have indicated that the handling of the bilingual corpus and the structure selection phase possess the greatest potential for further improvements. The outcome of these efforts will be reported in future articles.

References

Carbonell, J., Klein, S., Miller, D., Steinbaum, M., Grassiany, T. and Frey, J. (2006). Context-Based Machine Translation. In *Proceedings of the 7th Conference of the Association for Machine Translation in the Americas*, Cambridge, Massachusetts, USA, pp. 19-28.

Denkowski, M. and Lavie, A. (2011). Meteor 1.3: Automatic Metric for Reliable Optimization and Evaluation of Machine Translation Systems. *EMNLP 2011 Workshop on Statistical Machine Translation*, Edinburgh, Scotland, pp. 85-91.

Dologlou, I., Markantonatou, S., Tambouratzis, G., Yannoutsou, O., Fourla, A. and Ioannou, N. (2003). Using Monolingual Corpora for Statistical Machine Translation: The METIS System. In Proceedings of the EAMT- CLAW'03 Workshop, Dublin, Ireland, pp. 61-68.

Gale, D. and Shapley, L.S. (1962). College Admissions and the Stability of Marriage. *American Mathematical Monthly*, Vol. 69, pp. 9-14.

Groves, D. and Way, A. (2005). Hybrid data-driven Models of Machine Translation. *Machine Translation*, Vol. 19, pp.301-323.

Hutchins, J. (2005). Example-Based Machine Translation: a Review and Commentary. *Machine Translation*, Vol. 19, pp.197-211.

Koehn, P. (2010). *Statistical Machine Translation*. Cambridge University Press, Cambridge.

Lafferty, J., McCallum, A. and Pereira, F. (2001). Conditional Random Fields: Probabilistic Models for Segmenting and Labelling Sequence Data. *28th International Conference on Machine Learning, ICML 2011*, Bellevue, Washington, USA, pp. 282-289.

Markantonatou, S., Sofianopoulos, S., Giannoutsou, O. and Vassiliou, M. (2009). Hybrid Machine Translation for Low- and Middle- Density Languages. *Language Engineering for Lesser-Studied Languages, S. Nirenburg (ed.),* IOS Press, pp. 243-274.

Marsi, E., Lynum, A., Bungum, L. and Gambäck, B. (2011). Word Translation Disambiguation without Parallel Texts. *International Workshop on Using Linguistic Information for Hybrid Machine Translation,* Barcelona, Spain, pp. 66-74.

Mairson, H. (1992). The Stable Marriage Problem. *The Brandeis Review*, 12:1. Available at: www.cs.columbia.edu/~evs/intro/stable/writeup.html

NIST 2002. Automatic Evaluation of Machine Translation Quality Using n-gram Co-occurrences Statistics.

Papineni, K., Roukos, S., Ward, T. and Zhu, W.-J. (2002). BLEU: A Method for Automatic Evaluation of Ma-chine Translation. *40th Annual Meeting of the Association for Computational Linguistics*, Philadelphia, USA, pp. 311-318.

Phillips, A. (2011). CUNEI: Open-source Machine Translation with Relevance-based models of each translation instance. *Machine Translation*, Vol. 25, pp. 161-177.

Sanchez-Martinez, F. and Forcada, M.L. (2009). Inferring Shallow-transfer Machine translation Rules from Small Parallel Corpora. *Journal of Artificial Intelligence Research*, Vol. 34, pp. 605-635.

Schmid, H. (1994). Probabilistic Part-of-Speech Tagging Using Decision Trees. In *Proceedings of International Conference on New Methods in Language Processing*, Manchester, UK, pp. 44-49.

Smith, T.F. and Waterman, M.S. (1981). Identification of Common Molecular Subsequences. *Journal of Molecular Biology*, Vol. 147, pp. 195-197.

Snover, M., Dorr, B., Schwartz, R., Micciulla, L. and Makhoul, J. (2006). A Study of Translation Edit Rate with Targeted Human Annotation. In *Proceedings of the 7th Conference of the Association for Machine Translation in the Americas*, Cambridge, Massachusetts, USA, pp. 223-231.

S. Sofianopoulos, M. Vassiliou & G. Tambouratzis (2012) Implementing a language-independent MT methodology. Proceedings of the First Workshop on Multilingual Modeling, held within the ACL-2012 Conference, Jeju, Republic of Korea, July 13, pp.1-10.

Tambouratzis, G., Troullinos, M., Sofianopoulos, S. and Vassiliou, M. (2012a). Accurate phrase alignment in a bilingual corpus for EBMT systems. Proceedings of the 5th BUCC

Workshop, held within the LREC2012 Conference, May 26, Istanbul, Turkey, pp. 104-111.

Tambouratzis, G., Tsatsanifos, G., Dologlou, I. and Tsimboukakis, N. (2012b). SOM-based corpus modeling for disambiguation purposes in MT. In Proceedings of the Hybrid Machine Translation Workshop, held within the TSD2012 Conference (MTW-2012), Brno, Czech Republic, September 3, pp. 29-36 (ISBN 978-80-263-0266-7).

Tsimboukakis, N. and Tambouratzis, G. (2011). Word map systems for content-based document classification. *IEEE Transactions on Systems, Man & Cybernetics – Part C*, Vol. 41(5), pp. 662-673.

Native Tongues, Lost and Found: Resources and Empirical Evaluations in Native Language Identification

Joel TETREAULT[1] *Daniel BLANCHARD*[1]
Aoife CAHILL[1] *Martin CHODOROW*[2]
(1) Educational Testing Service, Rosedale Road, Princeton, NJ 08541, USA
(2) Hunter College and the Graduate Center, City University of New York, New York, NY 10065, USA
tetreaul@gmail.com, dblanchard@ets.org, acahill@ets.org,
martin.chodorow@hunter.cuny.edu

ABSTRACT
In this paper we present work on the task of Native Language Identification (NLI). We present an alternative corpus to the ICLE which has been used in most work up until now. We believe that our corpus, TOEFL11, is more suitable for the task of NLI and will allow researchers to better compare systems and results. We show that many of the features that have been commonly used in this task generalize to new and larger corpora. In addition, we examine possible ways of increasing current system performance (e.g., additional features and feature combination methods), and achieve overall state-of-the-art results (accuracy of 90.1%) on the ICLE corpus using an ensemble classifier that includes previously examined features and a novel feature (n-gram language models). We also show that training on a large corpus and testing on a smaller one works well, but not vice versa. Finally, we show that system performance varies across proficiency scores.

KEYWORDS: Native Language Identification, Text Classification, Corpora.

Proceedings of COLING 2012: Technical Papers, pages 2585–2602,
COLING 2012, Mumbai, December 2012.

1 Introduction

One growing NLP field is that of Native Language Identification (NLI), which is the task of automatically identifying a speaker's first language based solely on the speaker's writing in another language. NLI can be useful for a number of applications. Native language is often used as a feature in machine learning approaches to authorship profiling (Estival et al., 2007), which is frequently used in forensic linguistics. NLI can also be used in educational settings to provide more targeted feedback to language learners about their errors (Chang et al., 2008; Dahlmeier and Ng, 2011; Rozovskaya and Roth, 2011). It is well known that speakers of different languages make different kinds of errors when learning a language (Swan and Smith, 2001). For example, a French speaker learning English might write sentence (1), which contains a verb tense error. On the other hand, a Japanese speaker learning English might make the verb tense error shown in (2). A writing tutor system which can detect the native language of the learner will be able to tailor the feedback about the error and contrast it with common properties of the learner's language.

(1) She knows that she hasn't achieve it completely.

(2) They also said to have great curiosity.

There has been a great deal of work on NLI in recent years. The methods employed have ranged from some combination of lexical, part-of-speech and n-gram features (Koppel et al., 2005), to syntactic features (Wong and Dras, 2011) including Tree Substitution Grammars (TSGs) (Swanson and Charniak, 2012), to topic models (Wong et al., 2011). Despite these research efforts, it has been somewhat hard to compare different approaches for a number of reasons.

The first difficulty is with the evaluation data set. Evaluating an NLI system requires a corpus containing texts in a language other than the native language of the writer. Because of a scarcity of such corpora, most work[1] has used the ICLEv2[2] for training and evaluation since it contains several hundred essays written by college-level English language learners. However, this corpus is quite small for training and testing statistical systems which makes it difficult to tell whether the systems that are developed can scale well to larger data sets or to different domains. The usability of the corpus is further compromised by idiosyncrasies in the data such as topic bias (as shown by Brooke and Hirst, 2011) and the occurrence of characters which only appear in essays written by speakers of certain languages. As a result, it is hard to draw conclusions about which features actually perform best.

A second problem is that there is no consensus on the scope of the evaluation. The ICLE contains English essays written by native speakers of 16 languages. Typically a subset of 7 languages is used in the evaluations, although more recently some work has reported results for a larger set. Moreover, when researchers report results for 7 languages, they are not always reporting on the same 7 languages. For example, in the work of Wong and Dras (2011) the 7 native languages (L1s) are Bulgarian, Chinese, Czech, French, Japanese, Russian, and Spanish. Whereas in Brooke and Hirst (2012), Italian and Polish are used instead of Bulgarian and Czech. In addition, different researchers have split the corpus in different ways when training and evaluating their systems, making it even more difficult to compare results across experiments.

[1] Note that Kochmar (2011) used a subsection of the Cambridge Learner Corpus.
[2] Throughout this paper, we will refer to ICLE version 2 as ICLE.

In this paper, we first provide an automatic method for extracting data from the ICLE corpus to remove some of the corpus-specific idiosyncracies that automatic Native Language Identification classifiers currently learn from. We call this modified version of the ICLEv2, ICLE-NLI. Second, we introduce a new data set, TOEFL11 (Blanchard et al., to appear), which is roughly twice the size of the ICLE and has more essays per L1 than the ICLE. We argue for the use of TOEFL11, which will be made publicly available, as a common evaluation resource for this task. Next, we use these two new resources (ICLE-NLI and TOEFL11) to address the following research questions:

1. Are there methods previously unexplored in the literature that can be used to improve performance? (Section 5.1)
2. Do features commonly used in prior work generalize to different corpora? (Section 5.1)
3. What is the effect of training on one corpus and then testing on another? (Section 5.2)
4. What is the effect of larger training data on the performance of these features? (Section 5.3)
5. How widely do results vary across levels of writing proficiency? (Section 5.4)

In Section 2, we discuss previous approaches to NLI and also how there is a need for greater standardization of corpora used and evaluation practices. We present the corpora used in this study in Section 3, and our system and the features we investigate in Section 4. In Section 5 we discuss the results and the five research questions above. Our best system achieves state-of-the-art accuracy (90.1%) on the ICLE-NLI corpus, surpassing the previously reported best accuracy of 81.7% (Wong and Dras, 2011) on ICLE.

2 Related Work

The work of Koppel et al. (2005) set the stage (and probably a high bar as well in terms of performance) for much of the NLI research in the past few years. Their work investigated features from NLP and Second Language Acquisition including character and POS n-grams, content and function words, and spelling and grammatical errors. The features were evaluated on a subsection of the ICLE corpus consisting of essays sampled from 5 L1s (Russian, Czech, Bulgarian, French and Spanish) with 10-fold cross validation. The researchers found that by combining all of the features using a SVM, they could achieve an accuracy of 80.2%. Tsur and Rappoport (2007) continued this work by investigating why character n-grams alone performed so well (66%). As in the work of Wong and Dras (2011), we use an approximation of the Koppel et al. (2005) features both as a baseline system and as a base feature set to which we add our own features.

The notion that different learners tend to exhibit different grammatical error patterns was further explored by Kochmar (2011). Instead of the ICLE corpus, English learner essays from the Cambridge Learner Corpus[3] were used, specifically essays written by test-takers with Romance and Germanic native languages. In this work, a SVM was used to classify these essays on the basis of lexical and parse features as well as manually marked grammatical and spelling errors.

[3] http://www.cup.cam.ac.uk/gb/elt/catalogue/subject/custom/item3646603/Cambridge-International-Corpus-Cambridge-Learner-Corpus

One of the main conclusions was that character and POS n-grams were the most powerful features. Since this work used a different evaluation corpus and different L1s, it is hard to compare to other studies.

Wong and Dras (2009) based their NLI system on the contrastive analysis hypothesis: that differences between a writer's L1 and the language they are trying to write in are caused by differences between the two languages. Often, characteristics of the writer's L1 are carried over into the target language. They investigated the impact of three common ESL error types: subject-verb agreement, noun-number agreement and determiner errors and used 7 languages from ICLE: the 5 used in the Koppel et al. (2005) study in addition to Chinese and Japanese. While the determiner error feature did seem informative, performance did not improve when it was combined with a baseline model of lexical features.

Recently, more complex syntactic features have been proposed to better model the structural differences in learner writing. Wong and Dras (2011) extended the Koppel et al. work by incorporating production rules from two parsers as well as reranking features into the classification framework. In line with their prior work, they address the 7-way NLI task (Bulgarian, Chinese, Czech, French, Japanese, Russian and Spanish) by selecting 95 essays for each L1 and doing 5-fold cross validation with 70 essays selected for training and 25 for testing. This particular evaluation framework has been adopted by others in the field, such as Swanson and Charniak (2012). The combination of parse production rules with the Koppel et al. lexical features achieved a performance of 81.71% on the 7-way NLI task, currently the highest reported in the literature for that set of 7 in the ICLE. Swanson and Charniak (2012) experiment with various Tree Substitution Grammars (TSGs) on the 7-way NLI task and achieve an accuracy of 78.4%. In our work, we also experiment with the use of the Post and Gildea (2009) TSG fragments for the NLI task. Bergsma et al. (2012) tackled the related problem of classifying text as either written by a native speaker or a non native speaker of English. They used TSGs in conjunction with other stylometric features to achieve an F-score of 91.6.

Wong et al. (2011) also investigated the use of Latent Dirichlet Analysis (LDA) to cluster features and then adaptor grammars (Wong et al., 2012) to better capture arbitrary length n-gram sequences over tags and words. While the LDA approach did not improve performance over a baseline model of lexical features, the adaptor grammar approach scored close to state-of-the-art, around 75% accuracy.

Finally, Brooke and Hirst (2012) explored the effect of adding more training data on NLI classifier performance. In one experiment, English sentences written by Chinese and Japanese native speakers were scraped from a language-exchange social networking website (lang8)[4] and used to augment a classifier which used different n-gram features and function words. The effect was that for those two languages, increasing the training data from 200 texts to 5,000 improved overall performance by 30% (from 59.8% to 89.8%). This work sets the tone for the resource contribution of this paper: the creation of a publicly available corpus of ESL essays.

3 Data

In this work, we evaluated our NLI systems on four corpora: a subset of the ICLE (Granger et al., 2009) and three samples from a collection of essays written by non native English speakers as part of a high stakes college-entrance test (TOEFL®). Each corpus is described in detail below.

[4] http://www.lang-8.com/

3.1 ICLE Corpus Description

Currently, the only widely available corpus of non native English that is annotated for native language is the International Corpus of Learner English (ICLE) (Granger et al., 2009). It is a large collection of 6,085 essays written by university undergraduates of advanced proficiency.[5] The intent of the project was to produce for corpus linguistics a relatively large corpus that "shared a large number of task variables, notably in terms of medium (writing), genre (academic essay), field (general English rather than English for Specific Purposes) and length (between 500 and 1,000 words)" (Granger et al., 2009). Because these were the only factors the designers controlled for, there is substantial variation of other factors such as the number of essays per L1 and the number of languages covering each topic (see Table 1). The current version of the corpus, ICLEv2, consists of 6,085 essays for 1,302 prompts, which we have manually clustered into 736 topics.

Language	Unique Topics	Total Topics	Essays on Unique Topics	Total Essays
Bulgarian (Bul.)	0	4	0	302
Chinese (Chi.)	16	27	542	982
Czech (Cze.)	29	49	46	243
French (Fre.)	6	21	33	347
Japanese (Jap.)	119	132	336	366
Russian (Rus.)	2	17	2	276
Spanish (Spa.)	14	32	60	251

Table 1: ICLE counts of topics and essays for seven L1s commonly used in the NLI task.

3.2 Using the ICLE for NLI

Because techniques for NLI have looked for patterns in the data not only at the lexical level but also at the character level, any unintended lexical or character patterns correlated with L1 are likely to be weighted heavily by a statistical classifier. Therefore, it is important that the corpus is free of such patterns.[6] After examining the ICLE in detail, we discovered that there are two classes of confounding patterns for the task of NLI:

1. A variety of character encoding errors and annotations (both erroneous and correct) occur predominantly in certain languages and not in others. Because of the techniques and features commonly used for NLI, these issues can impact system evaluation.

2. The topics the authors write about are not evenly distributed across languages. In fact, there are many topics for which all the authors are native speakers of a single L1 (see Table 1). This topic bias is also observed and discussed in Brooke and Hirst (2011). The bias is problematic for machine learning approaches to NLI because it could cause classifiers to conflate the tasks of topic identification and NLI. For example, only Chinese

[5] http://www.uclouvain.be/en-cecl-icle.html

[6] We do acknowledge that there are two ways of evaluating NLI systems. The first one, and the approach we are taking here, is that all characters are encoded in the same way. The second way of evaluating is to actually include any patterns that may arise from using a certain keyboard or encoding scheme that writers with a certain L1 tend to use. The first method essentially focuses on solely on linguistic patterns, while the second is more application-driven and focuses on both linguistic and extra-linguistic patterns.

authors responded to the prompt "Discuss the advantages and disadvantages of using credit cards," and consequently all of those essays contain the $ character, whereas almost no other languages' essays do.

To address the first set of problems we (a) removed the header information from all documents, (b) removed all instances of the codes used by the annotators to indicate deleted references, quotations, or illegible words, (c) converted all non-ASCII characters to their closest ASCII equivalent using the Unidecode Python module,[7] (d) fixed characters that resulted from encoding errors by replacing them with the appropriate character, (e) deleted long quotes that were surrounded by <</>> instead of the usual English quote character ", and (f) removed the duplicate essay from `RUMO7057.txt`

To remove as much topic bias as possible from our ICLE dataset, we used a specific sample of the corpus that did not contain any topics that were found in one and only one native language group. The only exception to this was the sample for Japanese, where most of the topics were unique to that language group. We chose to leave the unique Japanese prompts in the sample because the alternative would have been to remove all of the Japanese essays from consideration. For the evaluations with ICLE discussed later in this paper (Section 5), we use this modified version of the ICLE. We provide a script which will automatically modify ICLE to address the issues above. The transformed version of this corpus is called ICLE-NLI.

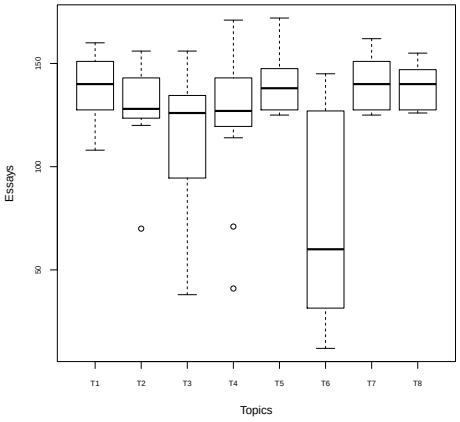


Figure 1: TOEFL11: Language distribution for each topic

3.3 TOEFL11: A New Corpus for NLI

While the modifications do address many of the issues with the ICLE corpus, it still has problems of small data size and some remaining topic bias. We constructed a new corpus of non native English writing called TOEFL11 which will be released through the LDC in 2013.

[7] http://code.zemanta.com/tsolc/unidecode/

This corpus consists of essays written by non native English speakers during a high stakes college-entrance test. It contains 1,000 essays per language sampled as evenly as possible from 8 topics along with author proficiency scores (low/medium/high) for each essay determined by assessment specialists. The distributions of the number of essays from each L1 group for each topic are shown in Figure 1. The 11 native languages covered by our corpus are: Arabic (Ara.), Chinese (Chi.), French (Fre.), German (Ger.), Hindi (Hin.), Italian (Ita.), Japanese (Jap.), Korean (Kor.), Spanish (Spa.), Telugu (Tel.), and Turkish (Tur.).

Corpus	**Languages (bold = common)**	**Essays per L1**	**Total Essays**	**Topics**	**Avg. Langs per Topic**	**Avg. Words per Essay**
ICLE-NLI	Bul., **Chi.**, Cze., **Fre.**, **Jap.**, Rus., and **Spa.**	110	770	76	1.4	666
TOEFL7	Bul., **Chi.**, Cze., **Fre.**, **Jap.**, Rus., and **Spa.**	659	4,613	79	6.5	252
TOEFL11	Ara., **Chi.**, **Fre.**, Ger., Hin., Ita., **Jap.**, Kor., **Spa.**, Tel., Tur.	1,000	11,000	8	11	315
TOEFL11-Big	Ara., **Chi.**, **Fre.**, Ger., Hin., Ita., **Jap.**, Kor., **Spa.**, Tel., Tur.	7,954	87,502	76	11	256

Table 2: Properties of NLI Corpora

We also compiled a larger dataset to investigate the effects of large amounts of training data on NLI accuracy. This corpus, which we will refer to as TOEFL11-Big, contains 87,502 essays from the same 11 languages as our public corpus, with between 7,900 and 7,983 essays per language. Again we sampled as evenly as possible across topics, but this time from a larger pool of 76 topics. There is no overlap in data between TOEFL11 and TOEFL11-Big.

To allow for a more direct comparison with previous ICLE results, we compiled a third corpus, henceforth TOEFL7, that uses the same 7 native languages that are most frequently used for NLI with the ICLE: Bulgarian, Chinese, Czech, French, Japanese, Russian, and Spanish. We used 659 essays per L1, sampled from a pool of 79 topics. In comparison, ICLE work typically uses 70 or 110 essays per L1.

Table 2 summarizes the four corpora to be used for training and testing in this paper. Of the three new corpora introduced in this paper, TOEFL11 will be made public.

4 System

Following previous work, we treat the problem of native language identification as a classification task. We train a native language identification system using a logistic regression model with L1-regularization.[8] We carry out 10-fold cross validation on all of the TOEFL® corpora and 5-fold cross-validation on the ICLE-NLI corpus (following Wong and Dras (2011)). In addition to building individual classifiers for each of our features, we also experiment with two ways of combining them. **Simple Combination** involves combining the features into one large set. In

[8]For all of our experiments we use the liblinear implementation (Fan et al., 2008) available from `http://www.csie.ntu.edu.tw/~cjlin/liblinear/` with the L1-regularized logistic regression solver and default parameters.

Character n-grams	All unigrams and bigrams present in each essay.
Function words	Function word counts based on the same list used by Koppel et al. (2005).
Part-Of-Speech (POS) bigrams	All POS bigrams present in each essay. Tags were obtained using the Stanford Tagger (Toutanova et al., 2003).
Spelling errors	Spelling errors returned by MS Word 2007 based on the list of error types used by Koppel et al. (2005). In the Koppel et al. (2005) work, spelling errors were derived using a pre-2003 version of MS Word.
Word n-grams	For unigrams, we restrict the list to correctly-spelled content words to prevent overlap with other features. For word bigrams, we do not filter the list.
Writing quality features	Counts and binary features quantifying different aspects of writing quality such as grammatical errors, style, discourse and vocabulary level. These features were derived using a proprietary automatic essay scoring engine (Attali and Burstein, 2006). Koppel et al. (2005) used a pre-2003 version of MS Word to find grammatical errors.
Tree Substitution Grammar Fragments	Counts and binary features corresponding to TSG fragments. These were extracted using the software described by Post and Gildea (2009) available from `https://github.com/mjpost/dptsg`.
Stanford Dependencies	Counts of basic dependencies extracted using the Stanford Parser (de Marneffe et al., 2006). Also included are variations of dependencies where lemmas were replaced by POS tags.
Language Model	Perplexity scores from 5-gram language models, one for each language in the corpus. Language Models were trained using the IRSTLM toolkit (Federico et al., 2008) .

Figure 2: A summary of all features used in our classifier

Ensemble Combination, which has not been explored for this task, each individual feature is trained as its own classification system and the predictions from that system, along with the scores (either probabilities or perplexity scores) are used in the final ensemble model to predict the native language.[9]

4.1 Features Used

The features used in our system are summarized in Figure 2. In order to be able to examine the performance of features across corpora, we implemented many of the features commonly used in the literature, as well as a novel language-model-based feature.

To begin with, we implement the features described by Koppel et al. (2005). The main classes of features in that paper were: character n-grams, function words, parts of speech, spelling errors, and writing quality features. We implement these features, roughly corresponding to the original work. One key difference is that we do not restrict our n-gram features to only include n-grams that occurred a certain number of times; we include all n-gram features as input to the classifier unless otherwise specified. For each feature, we implement both the binary and frequency variants. This set of binary and frequency features is combined to form the "Koppel Baseline."

We also combine Koppel's features with content words and word bigrams to produce a model

[9]Note that the splits used for all the individual classifier training and testing were identical.

that includes most of the simple features previous researchers have used for NLI. We only include counts of correctly spelled content words for the unigrams to avoid overlap with the other Koppel Baseline features.

Tree Substitution Grammar features have been used by previous researchers for this task (e.g., Swanson and Charniak, 2012). We take the fragments as generated by the Post and Gildea (2009) system and use those as fragments in our classification system. The Stanford dependency features are a variation on the syntactic features proposed previously (e.g., Kochmar, 2011; Wong and Dras, 2011; Post, 2011). We automatically extract all basic dependencies for each essay and consider each dependency to be a feature. This yields features of the form `nsubj(saw, dog)`, `dobj(saw, cat)`. We also carry out a backoff transformation based on part-of-speech, and for the two dependencies previously listed, would also consider the following features: `nsubj(VBD, dog)`, `dobj(VBD, cat)`, `nsubj(saw, NN)`, `dobj(saw, NN)`, `nsubj(VBD, NN)`, `dobj(VBD, NN)`.[10] We also consider the corresponding unlabeled dependencies as features. For almost all features we use both count-based (relative frequency for all but the writing quality features) and binary (presence/absence) features. We do not use binary features for the Stanford dependencies.

In addition to the previously described features, we also propose the use of language model perplexity scores. Surprisingly, language models, to our knowledge, have not been used for native language identification. We hypothesize that previous researchers may have avoided them because of the topic bias inherent in the ICLE or because there may not be enough data to build reliable models. Jarvis et al. (2012) showed that using higher-order *n*-gram features did not help for the task of L1 identification in the ICLE with 12 L1s and 10-fold cross-validation. We take a slightly different approach and train a 5-gram language model (with Witten-Bell smoothing) for *each* language in the corpus. We then apply each language model to each essay in the test data and choose as the prediction, the language model with the lowest perplexity.

5 Results

In this section, we discuss evaluations of different NLI systems under a variety of conditions to best answer the original research questions. In Section 5.1, we address the first two research questions by investigating the effectiveness of typical NLI features, as well as a novel feature (n-gram language models) across different corpora. In addition, we show that different feature combination methods can further improve system performance. In Section 5.2, we discuss the impact of training on one corpora and testing another. In Section 5.3, we discuss the impact of increasing the amount of training data on system performance. Finally, in Section 5.4 we show that the proficiency level of a writer should be taken into account when designing features and reporting results.

5.1 General Feature Discussion

This section reports the results of our experiments on the four corpora described in Section 3 to address the first two research questions posed. For each experiment we carried out cross validation, training on one portion of the data and testing on the remainder. We measure the performance of the classifier in terms of accuracy, i.e. the percentage of correctly classified languages in the corpus. Table 3 gives the results for all experiments. As a baseline, we take

[10] We ignore the index information provided by the Stanford Parser when storing the dependencies, however they are used when applying the part-of-speech backoff transformation.

our approximation of the Koppel et al. (2005) features. We also report results for features derived from Tree Substitution Grammar fragments, Stanford dependencies, and our novel language model feature. Finally the table shows that the ensemble method of combining features performs best, often significantly better than the simple combination of all features into one model.

On the ICLE-NLI corpus, our best ensemble model achieves an accuracy of 90.1% for a 7-way classification task. This is substantially higher than any previously reported results on the ICLE. Note that our experiments were carried out on a cleaned up version of the ICLE corpus (in an attempt to remove some of the biases, making the task more difficult). As noted in Section 2, it is very difficult to compare results for this task on the ICLE corpus because of the many different experimental procedures used by different researchers. However, despite these caveats, we believe that our system achieves state-of-the-art performance on this corpus for this task. Looking at the results for individual features, it is clear that the Koppel features alone do a very good job at predicting. Interestingly, the language model-based classifier also performs very well. The syntax-based features alone are also quite predictive, and our experiments show that the slightly more abstract representation of the syntactic structures within essays that can be captured by the Stanford dependencies is also a powerful predictor.

The results for TOEFL7, which contains the same languages as the ICLE-NLI corpus with a similar number of total topics but is around 6 times larger, show that the features and combinations that worked extremely well for the ICLE-NLI corpus, do not perform as well on this new corpus. The best result is achieved by our ensemble model with an accuracy of 70.9%. Again we see that the baseline Koppel features are very strong, although the language model-based classifier does not outperform the syntax-based models on this corpus. We believe that this indicates that while we may be reaching the upper bounds of performance on the ICLE-NLI corpus, there still remains plenty of room to improve current models and develop them to be able to perform well across corpora.

Interestingly, all of the combined models (and most of the individual ones) perform substantially worse on TOEFL7 than on any of the other corpora. There are two properties of the TOEFL7 corpus that are likely the cause for this performance disparity: (a) essay length and (b) topic distribution. The essays in the ICLE-NLI are on average 666 words long (see Table 2), whereas the TOEFL7 essays are an average of 252 words long. Therefore, while there are more total essays in the TOEFL7 corpus to use for training than in the ICLE-NLI sample, there are fewer potential occurrences of each useful feature per essay, which means the binary versions of the features are less helpful. This length disparity does not exist between the TOEFL7 and TOEFL11 data sets, but the distribution of topics is very different, which may contribute to the difference in performance. The topic distributions for TOEFL11 and TOEFL11-Big were made as even as possible, but we did not have access to enough of some of the ICLE-NLI languages to be able to sample TOEFL7 as evenly.

Our performance on the TOEFL11 corpus, developed specifically for the task of native language identification, are encouraging. Our best ensemble model achieves 80.9% accuracy, and the performance of the individual features follows a pattern similar to the ICLE-NLI and TOEFL7 evaluations. These results show that the features and their combinations do scale to larger corpora (1000 essays per language, 11 native languages)

Finally, on our largest corpus, TOEFL11-Big we see the same patterns as on the other corpora. Here, the increased data size does seem to lead to some small improvements in performance

Features	ICLE-NLI (7-way)	TOEFL7 (7-way)	TOEFL11 (11-way)	TOEFL11-Big (11-way)
Random baseline	14.3	14.3	9.1	9.1
Koppel baseline	80.0	58.0	65.9	67.3
character unigrams	20.1	23.2	23.8	28.1
character unigrams (binary)	46.1	29.2	26.6	24.3
character bigrams	14.5	26.0	25.0	39.2
character bigrams (binary)	70.3	44.4	52.2	51.9
function words	23.1	25.5	27.2	42.3
function words (binary)	57.7	38.8	45.9	43.4
POS bigrams	25.6	31.3	26.8	41.4
POS bigrams (binary)	54.8	42.0	38.5	46.0
spelling	29.9	31.8	31.1	30.7
spelling (binary)	29.5	31.3	29.6	30.1
writing quality	57.4	35.6	37.2	32.8
writing quality (binary)	40.6	27.8	26.0	24.3
Koppel + word n-grams	82.9	67.5	76.6	81.4
word unigrams	22.7	18.6	21.3	44.9
word unigrams (binary)	76.5	52.7	64.9	63.5
word bigrams	14.3	19.0	22.2	41.7
word bigrams (binary)	72.7	53.9	67.9	77.1
5-gram language models	80.8	53.4	73.9	74.4
tree subst. grammar frags.	74.4	55.6	62.6	64.3
stanford dependencies	77.1	59.3	70.9	76.7
simple combination	82.6	70.5	76.0	80.9
ensemble	**90.1**	**70.9**	**80.9**	**84.6**

Table 3: Cross-validation results for all systems on each corpus; accuracy in %

for some features, but not all. Particularly, some of the binary features perform worse on the TOEFL11-Big corpus compared to the smaller TOEFL11 corpus. The syntax-based features do seem to be one of the stronger features given the extra data.

We notice that the performance of the language-model feature is inconsistent across corpora. It remains unclear why this is the case, but we hypothesize that it is partly due to topic distributions. We will continue to explore the cause of this inconsistency. Noteworthy, is the fact that word unigrams/bigrams are generally one of the strongest baselines. Our language-model feature is a more complex version of these features. These simple features alone perform about as well as the entire set of features that goes into the Koppel baseline. One reason previous researchers had given for avoiding these word-level features was because it was felt that they were unfairly advantaged because of the topic bias in ICLE. Our experiments show, however, that even in corpora where there is no topic bias (TOEFL11), these word-level features remain predictive.

5.2 Cross-Corpus Evaluation

The experiments in Section 5.1 all involved cross-validation on one corpus. A remaining issue to address is the question of whether a system trained on one corpus can generalize to another. Brooke and Hirst (2011) reject cross-validation as a means of evaluating NLI systems on corpora that are heavily topic biased and instead train their system on a large amount of web-scraped data.

In our study of the effect of training on one corpus and testing on another, we carry out experiments on pairs of corpora that consist of the same sets of languages. We evaluate first on the ICLE-NLI vs TOEFL7 corpora (7 languages) and second on the TOEFL11 vs TOEFL11-Big corpora (11 languages). The main argument for carrying out this evaluation as proposed by Brooke and Hirst (2011) is to circumvent the issue of topic bias. We believe that these pairs of corpora are composed of sufficiently different topics that there should not be significant overlap.

For the ICLE-NLI vs TOEFL7 experiment, we train a classifier using the combined set of features listed in Figure 2 apart from the language model features. The results are reported in Table 4. The results generally show lower performance than the cross-validation experiments. In particular, the system trained on the ICLE-NLI data set does not generalize at all to the TOEFL7 data set. Training on TOEFL-Big and testing on TOEFL11 shows that the large amount of training data available in the TOEFL11-Big corpus leads to similar performance as the cross-validation result on TOEFL11. In general it seems that training on a larger corpus and testing on a smaller one works reasonably well, however training on a small corpus and testing on a larger one does not yield good results with our feature set. It remains for future work to determine which individual features can generalize well in this scenario.

Train	**Test**	**Accuracy(%)**
ICLE-NLI	TOEFL7	26.6
TOEFL7	ICLE-NLI	67.4
TOEFL11	TOEFL11-Big	35.4
TOEFL11-Big	TOEFL11	79.2

Table 4: Results for Cross-corpus evaluation

5.3 Corpus size

The relatively small size of the portion of the ICLE corpus traditionally used for the task of NLI has been a major criticism (Brooke and Hirst, 2011). Since our TOEFL11-Big corpus is several orders of magnitude larger than this, we are in a favorable position to be able to examine the impact of adding additional training data to NLI classifiers. In Brooke and Hirst (2011), they automatically gleaned additional training data by scraping online blogs. Our large data set is composed of essays written by learners of English and the native language self-reported by the learners.

Figure 3 shows the learning curve for an ensemble classifier on our TOEFL11-Big data set. When increasing the size of the data set, we add the same number of essays for each language, where possible. The graph shows a steep rise at the beginning, but as more data is added, the increase in performance is low and appears to be leveling off. The graph shows that the 11,000 essays in our publicly available data set should be a large enough number to be able to train classifiers with high accuracy, but that there could also be performance improvements with a somewhat larger dataset.

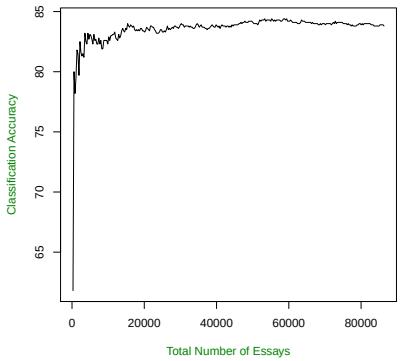


Figure 3: 11-way Classifier accuracy when trained on increasing amounts of data

5.4 Proficiency-based Evaluation

One issue that has not been thoroughly addressed in the field is the effect of writer proficiency on system performance. If a corpus is composed of highly proficient writers, then one would expect the effects of features such as spelling and grammatical errors to be less useful since they would be less common. Conversely, if the writer is of moderate or low proficiency, error features may be a larger contributor to classifier performance while more complex features may be less so due to data sparsity. So, when designing an NLI system and evaluating it, it is important to take into account the proficiency of the writer. In a pilot study, Jarvis and Paquot (2012) found that it is necessary to control for proficiency since it impacts the final discriminative analysis. They used trained assessment specialists to manually rate 223 ICLE essays written by learners of 3 L1s: Spanish, German and French. They found that the Spanish essays tended to have lower proficiency scores than the other two and were thus more easily distinguishable using the standard error techniques.

To investigate the influence of author proficiency on classification accuracy, we examined the performance of our best ensemble model at the three proficiency scores. As can be seen in Table 5, accuracy is highest for essays with a medium proficiency score. One possible reason for this is that medium proficiency essays feature a significant number of errors that many of the features (e.g. spelling errors, POS bigrams, and writing quality) can be used for prediction, but not so many that the sentences are seriously ungrammatical and difficult to interpret, as is the case with low-scoring essays. It is not surprising that high-score essays are difficult for NLI because their near-native writing would contain substantially fewer predictive errors. An alternative explanation for the results in Table 5 is that they simply reflect the score distribution in the corpus. We will continue to examine these hypotheses in future work.

We believe that proficiency reporting should be a necessary “best practice” in the NLI field. As more learner corpora become available, such as TOEFL11, Cambridge Learner Corpus and lang8, including a breakdown of classifier performance by proficiency score will make it easier to compare and discuss results across corpora.

Proficiency Score	Accuracy (%)	Number of Essays
Low	82.8	1,201
Medium	86.1	5,964
High	79.8	3,835

Table 5: TOEFL11 Accuracy for Best Model by Proficiency Score

Conclusion

To date, it has been hard to interpret results or compare systems because of the reliance on the ICLE corpus. In this paper, we addressed these issues by providing two resources: a modified version of the ICLEv2 (ICLE-NLI) and a larger corpus of essays that is more balanced across topics (TOEFL11). It is our hope that making these resources publicly available will help foster better standardization in the growing field of NLI.

Using these two corpora, we were able to draw the following conclusions:

1. Many of the trends found in previous work on the ICLE do generalize to other corpora, such as the power of the Koppel et al. (2005) features and word n-gram frequencies.
2. N-gram language models, which had been heretofore unexplored in this work, perform comparatively well for all corpora we examine.
3. Training on a large corpus and testing on a smaller corpus works well, but classifiers trained on smaller corpora do not appear to generalize well.
4. Combining multiple features in an ensemble classifier yields significantly greater classification accuracy than simply using including all of the features in one large classifier for all corpora we examine.
5. For the ICLE, the ensemble method has an accuracy of 90.1%, which is higher than the previously reported best accuracy of 81.7%.
6. Classification accuracy varies across proficiency levels, however further research is required in order to be able to explain the reasons for this.

Acknowledgments

The authors wish to thank Ben Swanson, Julian Brooke, Matt Post, Jojo Wong and Sylviane Granger for stimulating discussions related to this work. We would also like to thank the anonymous COLING reviewers, Jill Burstein and Nitin Madnani for their helpful comments and suggestions for improving earlier versions of this paper. We are grateful to Derrick Higgins, Anthony Ostrander, Xiaoming Xi and Robbie Kantor for their help in making the public release of the TOEFL11 dataset possible.

References

Attali, Y. and Burstein, J. (2006). Automated Essay Scoring with e-rater V.2. *Journal of Technology, Learning, and Assessment*, 4(3). Available from http://www.jtla.org.

Bergsma, S., Post, M., and Yarowsky, D. (2012). Stylometric analysis of scientific articles. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 327–337, Montréal, Canada. Association for Computational Linguistics.

Blanchard, D., Tetreault, J., Higgins, D., Cahill, A., and Chodorow, M. TOEFL11: A Corpus of Non-Native English. Research report (to appear), Educational Testing Service.

Brooke, J. and Hirst, G. (2011). Native language detection with 'cheap' learner corpora. In *Proceedings, Conference on Learner Corpus Research*, Louvain-la-Neuve. Presses universitaires de Louvain.

Brooke, J. and Hirst, G. (2012). Measuring interlanguage: Native language identification with l1-influence metrics. In *Proceedings, 8th ELRA Conference on Language Resources and Evaluation (LREC 2012)*, Istanbul.

Chang, Y.-C., Chang, J. S., Chen, H.-J., and Liou, H.-C. (2008). An automatic collocation writing assistant for Taiwanese EFL learners: A case of corpusbased NLP technology. *Computer Assisted Language Learning*, 21(3):283–299.

Dahlmeier, D. and Ng, H. T. (2011). Correcting Semantic Collocation Errors with L1-induced Paraphrases. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 107–117, Edinburgh, Scotland, UK. Association for Computational Linguistics.

de Marneffe, M.-C., MacCartney, B., and Manning, C. D. (2006). Generating Typed Dependency Parses from Phrase Structure Parses. In *Proceedings of the Fifth International Conference on Language Resources and Evaluation (LREC'06)*, pages 449–454, Genoa, Italy.

Estival, D., Gaustad, T., Pham, S.-B., Radford, W., and Hutchinson, B. (2007). Author profiling for English emails. In *Proceedings of the 10th Conference of the Pacific Association for Computational Linguistics (PACLING)*, pages 263–272.

Fan, R.-E., Chang, K.-W., Hsieh, C.-J., Wang, X.-R., and Lin, C.-J. (2008). LIBLINEAR: A library for large linear classification. *Journal of Machine Learning Research*, 9:1871–1874.

Federico, M., Bertoldi, N., and Cettolo, M. (2008). IRSTLM: an Open Source Toolkit for Handling Large Scale Language Models. In *Proceedings of Interspeech*, pages 1618–1621, Brisbane, Australia.

Granger, S., Dagneaux, E., and Meunier, F. (2009). *The International Corpus of Learner English: Handbook and CD-ROM, version 2*. Presses Universitaires de Louvain, Louvain-la-Neuve, Belgium.

Jarvis, S., Castañeda-Jiménez, G., and Nielsen, R. (2012). *Approaching Language Transfer through Text Classification*, chapter Detecting L2 Writers' L1s on the Basis of Their Lexical Styles, pages 34–70. Multilingual Matters.

Jarvis, S. and Paquot, M. (2012). *Approaching Language Transfer through Text Classification*, chapter Error Patterns and Automatic L1 Identification, pages 127–153. Multilingual Matters.

Kochmar, E. (2011). Identification of a writer's native language by error analysis. Master's thesis, University of Cambridge.

Koppel, M., Schler, J., and Zigdon, K. (2005). Automatically determining an anonymous author's native language. In *ISI*, pages 209–217.

Post, M. (2011). Judging grammaticality with tree substitution grammar derivations. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 217–222, Portland, Oregon, USA. Association for Computational Linguistics.

Post, M. and Gildea, D. (2009). Bayesian Learning of a Tree Substitution Grammar. In *Proceedings of the ACL-IJCNLP 2009 Conference Short Papers*, pages 45–48, Suntec, Singapore. Association for Computational Linguistics.

Rozovskaya, A. and Roth, D. (2011). Algorithm Selection and Model Adaptation for ESL Correction Tasks. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 924–933, Portland, Oregon, USA. Association for Computational Linguistics.

Swan, M. and Smith, B., editors (2001). *Learner English: A teacher's guide to interference and other problems*. Cambridge University Press, 2 edition.

Swanson, B. and Charniak, E. (2012). Native language detection with tree substitution grammars. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 193–197, Jeju Island, Korea. Association for Computational Linguistics.

Tetreault, J. R. and Chodorow, M. (2008). The Ups and Downs of Preposition Error Detection in ESL Writing. In *Proceedings of the 22nd International Conference on Computational Linguistics (Coling 2008)*, pages 865–872, Manchester, UK. Coling 2008 Organizing Committee.

Toutanova, K., Klein, D., Manning, C., and Singer, Y. (2003). Feature-Rich Part-of-Speech Tagging with a Cyclic Dependency Network. In *Proceedings of HLT-NAACL 2003*, pages 252–259, Edmonton, Canada.

Tsur, O. and Rappoport, A. (2007). Using classifier features for studying the effect of native language on the choice of written second language words. In *Proceedings of the Workshop on Cognitive Aspects of Computational Language Acquisition*, pages 9–16, Prague, Czech Republic. Association for Computational Linguistics.

Wong, S.-M. J. and Dras, M. (2009). Contrastive analysis and native language identification. In *Proceedings of the Australasian Language Technology Association Workshop 2009*, pages 53–61, Sydney, Australia.

Wong, S.-M. J. and Dras, M. (2011). Exploiting parse structures for native language identification. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 1600–1610, Edinburgh, Scotland, UK. Association for Computational Linguistics.

Wong, S.-M. J., Dras, M., and Johnson, M. (2011). Topic modeling for native language identification. In *Proceedings of the Australasian Language Technology Association Workshop 2011*, pages 115–124, Canberra, Australia.

Wong, S.-M. J., Dras, M., and Johnson, M. (2012). Exploring adaptor grammars for native language identification. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 699–709, Jeju Island, Korea. Association for Computational Linguistics.

Inverse Document Density: A Smooth Measure for Location-Dependent Term Irregularities

Dennis THOM Harald BOSCH Thomas ERTL
Institute for Visualization and Interactive Systems, University of Stuttgart, Germany
Dennis.Thom@vis.uni-stuttgart.de, Harald.Bosch@vis.uni-stuttgart.de,
Thomas.Ertl@vis.uni-stuttgart.de

ABSTRACT
The advent and recent popularity of location-enabled social media services like Twitter and Foursquare has brought a dataset of immense value to researchers in several domains ranging from theory validation in computational sociology, over market analysis, to situation awareness in disaster management. Many of these applications, however, require evaluating the a priori relevance of trends, topics and terms in given regions of interest. Inspired by the well-known notion of the *tf-idf* weight combined with kernel density methods we present a smooth measure that utilizes large corpora of social media data to facilitate scalable, real-time and highly interactive analysis of geolocated text. We describe the implementation specifics of our measure, which are grounded in aggregation and preprocessing strategies, and we demonstrate its practical usefulness with two case studies within a sophisticated visual analysis system.

TITLE AND ABSTRACT IN ANOTHER LANGUAGE, L_2 (OPTIONAL, AND ON SAME PAGE)

Inverse Dokumentdichte: Ein glattes Maß für ortsbezogene Termnutzungsunregelmäßigkeiten

Das Aufkommen und die derzeitige Beliebtheit von ortsbezogenen Diensten der Sozialen Medien wie Twitter und Foursquare haben einen Datensatz von immensem Wert für Forscher verschiedener Domänen, von der Aussagenvalidierung in der Soziologie, über Marktanalysen, bis zur Situationseinschätzung im Katastrophenschutz, geschaffen. Viele dieser Anwendungen erfordern jedoch eine Einschätzung der a priori Relevanz von Trends, Themen und Termen für gegebene geographische Regionen. Basierend auf der Idee hinter dem bekannten Tf-idf-Maß, präsentieren wir eine geglättetes Maß, welches, durch die Ausnutzung großer Korpora bestehend aus Daten der Sozialen Medien, die skalierbare, echtzeitfähige und voll interaktive Analyse von geokodierten Texten ermöglicht. Wir beschreiben die Details der Umsetzung unseres Maßes, welche auf Aggregations- und Vorverarbeitungsstrategien beruht, und wir zeigen seine praktische Anwendbarkeit durch zwei Fallbeispiele mit Hilfe eines elaborierten Systems zur visuellen Analyse.

KEYWORDS: tf-idf, term density, geolocated corpora, Visual Analytics, Twitter, social media.

KEYWORDS IN L_2: TD-IDF, Termdichte, geokodierte Korpora, Visual Analytics, Twitter, Soziale Medien.

Proceedings of COLING 2012: Technical Papers, pages 2603–2618,
COLING 2012, Mumbai, December 2012.

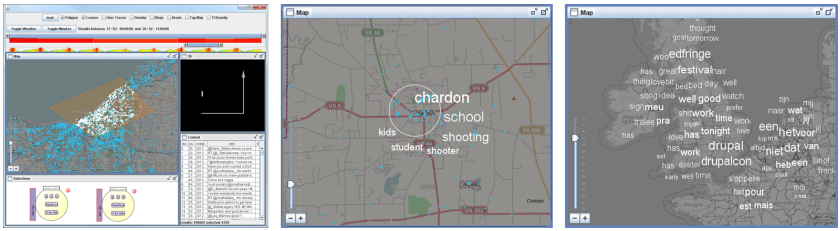


Figure 1: Left: The ScatterBlogs system for visual social media analysis. Interactive exploration techniques are used to examine aggregated message contents. Middle: *content lens* — a circle which can be interactively moved over the map to show the most frequent terms within a region and timeframe. Right: Cluster analysis is used to detect spatiotemporal clusters of similar topic usage to display them on a map.

1 Introduction

Every time Twitter users write a message on their GPS-enabled mobile device, they can attach precise location information. Each day more than 3 million documents are produced this way. Within one year, people from all over the world have generated a corpus comprising more than 1.3 billion[1] geolocated messages. Such location-enriched text data has tremendous value for researchers and analysts in several fields ranging from theory validation in computer sociology to location aware market analysis. Most notably, this data source has opened important application domains for research in situation awareness and disaster management, since the community of social media users can serve as a global 'sensor network' of potential incident reporters of critical events (Sakaki et al., 2010).

Due to the high complexity and volume of data, automated means for language processing and information mining are often complemented by highly interactive visualization tools (MacEachren et al., 2011; Marcus et al., 2011) that help analysts to explore and filter relevant messages and detect important localized events like river floods, wildfires, typhoons, hurricanes, infectious diseases or shooting incidents (Starbird and Palen, 2010; Mendoza et al., 2010; Sakaki et al., 2010; Hughes and Palen, 2009; Chew and Eysenbach, 2010; Palen et al., 2009; Heverin and Zach, 2010). Several of these tools have shown that aggregated representations of keywords and topics of large message volumes from selected or automatically determined spatiotemporal ranges can be of great value for precise situation assessment and large scale anomaly indication. For example, the ScatterBlogs Visual Analytics system (Bosch et al., 2011; Thom et al., 2012), depicted in Figure 1, provides means to aggregate large volumes of textual data in selected areas and indicates spatiotemporal topic clusters with geolocated Term Clouds.

Relevant keywords in such systems are often detected through their high term-occurrence counts compared to the counts of other terms in the vicinity. It has been a challenge, however, to evaluate whether documents or even individual terms are actually anomalous outliers within the specified area and timespan or just perennially prominent terms like stopwords or certain keywords that are frequently used in the examined region or throughout the whole Twitter

[1]Tweets can either have a user defined geotag (e.g. `London`) or precise latitude/longitude coordinates. In this work we only consider the latter ones, comprising a subset of 0.7 billion messages.

network. As a result, relevant and anomalous documents and keywords are often obscured by falsely highlighted keywords resulting from regular day-to-day chatter. Conventional solutions for this problem rely on the use of *tf-idf* and similar measures, which indicate whether a keyword is specifically important within a selected document set or whether it is also frequent within the corpus of all documents. However, when looking at geolocated documents the situation is quite different. Besides globally prominent terms, which can easily be detected by *tf-idf*, there are also many terms that are only prominent within a certain region. Thus, when examining a certain geospatial area and timeframe of documents, the importance of certain prominent and unusual terms can still be easily obscured by numerous and ordinary terms for the area.

To address this problem we present a smooth and scalable technique for geospatial term relevance normalization. Based on the combination of *tf-idf* with kernel density methods, a complete one-year corpus of geolocated Twitter messages is evaluated to determine the a priori probability that a given term is contained in a document composed at a given location. The intuition behind our measure is quite simple: In the *idf* part of the *tf-idf* measure, the number of documents in which a term appears is counted in order to put the term frequency in relation to the sum of documents in the corpus. In contrast, our measure sums for a given location the derived probabilities that a document containing the term could have appeared at this point and puts it in relation to the sum of derived probabilities that any document could have appeared at this point. The outcome of this is the (im)probability that a term is contained in a message appearing at any given point, allowing to assess the abnormality of observed term occurrences in examined document sets.

The details and a formal definition of this concept will be given in Section 3. An important contribution of our paper is a scalable implementation to calculate, store and quickly retrieve the normalization values based on adaptive grid aggregation techniques, described in Section 4. Since the proposed technique was particularly developed for large scale interactive document exploration, the performance details are discussed in Section 5 and its practical applicability is evaluated in Section 6. We conclude with a discussion of the results, final remarks and an outlook on future work.

2 Related Work

Work related to the described approach can be found mainly in two areas, which will be addressed in this section. First, geolocated resources can be used to create meta-documents for specific locations in order to geographically tag new resources according to their similarity to these meta-documents. Second, the geographic information of resources can be exploited to establish a 'geo-ranking' of search results lists.

Several approaches propose a prediction of a resource's geolocation inferred from the textual content or associated tags by features such as toponyms, geographic and cultural features, and stylistic and dialectic differences. Wing & Baldridge (2011) discretize the world with a regular grid of equal degree and calculate the term probabilities per document, grid cell and corpus using different geolocated document collections, i.e. Wikipedia articles and Twitter messages. Given a new and unlocated document, they can calculate the similarity between the document term-distribution and the cells' term-probabilities and choose the closest cell as the probable document location. Roller et al. (2012) follow the same approach but construct an adaptive grid from a space-partitioning k-d-tree trained on the same document collections for efficiency and higher resolutions in densely populated regions. For each grid cell, a pseudo-document is assembled from all training documents that lie within the cell and again their similarities

to documents of unknown origin are calculated. This can lead to a decreasing quality of the measure when the resolution is increasing, due to an overfitting of the data when the pseudo-documents will become small and specific for the high resolution cells. In contrast, our approach uses a smoothing kernel, such that the measure's quality is increasing with higher resolutions. Serdyukov et al. (2009) present a similar approach related to the tags associated with user uploaded photos. They consider a bag-of-tags model for each grid cell and combine it with a smoothing strategy and additional codified knowledge about geolocations from services such as GeoNames[2]. However, all these approaches examine the situation where an unknown document is given and the most probable location of its origin has to be found. Therefore it is reasonable to estimate the probability that a certain term or document *appears* at a certain location. In contrast, our approach estimates the probability that a document written at a certain location *contains* a certain term. What seems to be just a slight difference, leads to a completely new application domain, as will be demonstrated in our case studies.

Zhang et al. (2010) also present a geospatial extension of the *tf-idf* measure in the context of tag centric query processing for geolocated web 2.0 resources. The approach tries to identify regions where each of the query tags is covered by at least one nearby web 2.0 resource. The purpose of the measure is to rank regions within the result set according to how characteristic the tags are for each region. Due to the focus on querying, their use case and therefore the measure itself and its implementation differs from ours in the following aspects. The geospatial extension can be seen as replacing documents by regions and the corpus by the set of all regions. A term is therefore characteristic for a region when it is *frequently* observed in the region but *infrequently* used globally. As our work focuses on exploring large data sets with potentially huge amounts of frequent but irrelevant chatter, we need a measure for normalization with respect to long-term keyword densities. Our approach contrasts term densities with document densities for each individual point of the world. Therefore, a term can still achieve a high score for a specific region even if it is a globally common word. Furthermore, the approach of Zhan et al. utilizes a fixed grid in order to approximate a smooth measure and a tagged resource can only influence directly neighboring cells by a two-step degradation function. This limits the spatial scalability and allows only rough resolutions when dealing with global data.

The World Explorer visualization tool (Ahern et al., 2007) identifies representative labels for geolocated photo collections. The available photos are geospatially clustered, independent of their associated tags, to form potential map label locations for each zoom level and map tile individually. Afterwards, a representative tag is selected for each cluster by evaluating an adapted *tf-idf* measure. Similarly to Zhang et al., the tag frequency within a cluster is related to its overall frequency within one tile. With this approach, tags like *San Francisco* are relevant on the scale of the state of California, but insignificant on a detailed tile of San Francisco itself. Eisenstein et al. (2010) present a latent geographic topic model to identify words with high regional affinity, geographically coherent linguistic regions, and regional variations of general topics. The model can be seen as a latent Dirichlet allocation with regional corruptions of the base topics. The model was applied to 380k twitter messages and achieved a good performance in locating users by their allocation to a regional topic.

3 Model

In information retrieval and language processing, the *tf-idf* measure is used to evaluate the relevance of a term for a given document within a given corpus (Jones, 1972; Manning et al.,

[2] `http://www.geonames.org`

2008). Given a document d and a term t, the number of occurrences $tc_{t,d}$ of t in d is determined and normalized by document length to compute the term frequency $tf_{t,d} = \frac{1}{|d|} tc_{t,d}$. This results in the relative prominence of t within d. A high term prominence, however, is no indicator that the term is also important to the document, since common terms like `the`, `she` or `like` are prominent within most documents. Therefore, one also has to compute the inverse document frequency based on the corpus D of all documents, from which d was taken:

$$idf_{t,D} = \log \frac{|D|}{|\{d|d \in D \wedge t \in d\}|} \tag{1}$$

The *idf* can be seen as an indicator for the a priori probability that t appears in documents drawn from D - the higher the probability, the lower the value. The *tf-idf* is computed by simply multiplying the values:

$$tfidf_{t,d,D} = tf_{t,d} * idf_{t,D} \tag{2}$$

Moving from regular document sets to the domain of geolocated messages, we find a very diverse corpus with contents ranging over several different regional topics, characteristics and languages. For example, in most major cities around the globe the city's own name as well as the names of individual districts are usually mentioned in hundreds of Twitter messages every day. The same is true for regional points of interest or words from languages and dialects which are only used in distinct parts of the world. In order to estimate, whether a term is actually important or anomalous for a particular map area and time span, it is not sufficient to compare its local term frequency against the global *idf* value. Globally, the term `Denver` could have a similar or even lower *idf* value as `shooter`. What is needed instead is a measure that compares the term count of the examined message set with the estimated inverse document frequency of all messages that have been written in the appropriate region and in a larger timeframe than that of the message set.

To estimate this inverse document density for arbitrary map locations from the sparsely distributed message corpus, we utilize kernel based density estimation (KDE), which approximates probability distributions from discrete point data, and integrate it with the traditional *tf-idf* measure. Characteristics of the KDE and the formal definition of the new measure will be detailed in the following two subsections.

3.1 Kernel Density Estimation

It is a well-known problem to derive a continuous probability density f from a finite sample data set $X = \{x_1 \dots x_n\}$. For example, X could be a list of locations of crime reports for a major city. In KDE techniques (Rosenblatt, 1956; Parzen, 1962), the so-called density estimator for the data set is constructed as follows:

$$\hat{f}(x) = \frac{1}{nh} \sum_{i=1}^{n} K\left(\frac{d(x, x_i)}{h}\right) \tag{3}$$

In this equation, $d(x, y)$ is a distance metric, e.g. the Euclidean distance in case of samples from $\mathbb{R}^n$. The function K is the kernel and it is used together with the bandwidth h to assign a weighted value to each x_i depending on its distance from x. For the sake of simplicity one can also write the equation using subscript notation $K_h(u) = \frac{1}{h} K(\frac{u}{h})$. Selected kernel functions

usually have their maximum at $u = 0$, are rapidly decreasing with higher u, and integrate to 1. A Gaussian kernel is used in our implementation but other choices are also conceivable:

$$K(u) = \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{u^2}{2}\right) \tag{4}$$

The purpose of using these functions is to reflect the probability that a given sample could deviate a given distance u from its actual location. For example, if a crime was committed somewhere along a street at point x, without prior knowledge, it could as well have happened 5, 10 or 100 meters farther away with decreasing probabilities. By summing and normalizing all these spatially decreasing probabilities from all samples, the constructed function $\hat{f}(x)$ gives us a relative estimate of the probability that a crime can happen at an arbitrary location.

3.2 Measure Definition

Similar to the crime example above, the KDE principle can be applied to a sample of term occurrences from Twitter messages m_t containing term t at location x to inspire the estimation of term occurrence densities at arbitrary map locations. In the presented measure, KDE is employed to define the localized version of the inverse document frequency *idf* as basis for the localized *tf-idf* measure. It is assumed, that a given corpus G of geolocated messages has been collected over a large temporal range, e.g., one year, to stabilize the corpus against seasonal characteristics. We first define a measure for local term density and normalize it by the corresponding local document density to calculate the local inverse document density:

For a given term t let $G_t = \{m \in G : t \in m\}$ be the subset of messages from G that contain t and let $loc(m) \in \mathbb{R}$ be the location of message m in the coordinate space. Furthermore, let K_h be a kernel with fixed bandwidth h. For a given location $x \in \mathbb{R}$ we call

$$td_t(x) = \sum_{m \in G_t} K_h(d(x, loc(m))) \tag{5}$$

the term density of t at x. Note that these are absolute and not relative densities as they are not normalized with $\frac{1}{|G_t|}$. In order to allow a cross-comparison of different terms, the terms' densities are normalized at each location using the term independent document density at the same location:

$$dd(x) = \sum_{m \in G} K_h(d(x, loc(m))) \tag{6}$$

Finally, and analogous to the *idf*, we call

$$idd_t(x) = \log \frac{dd(x)}{td_t(x)} \tag{7}$$

the inverse document density of t at x.

In these equations the distance function must be matched to the coordinate space that has been chosen to represent message locations. For the example in Section 3.1, we assumed a uniform grid coordinate system and thus the Euclidean metric was an appropriate choice. However, since Twitter messages are usually given in graticule coordinates (latitude, longitude), one can convert them to a uniform grid or use the haversine formula (Sinnott, 1984) to approximate the distance.

For the term frequency value *tf* there is a natural analogue in most application cases. However, compared to the *idd* these values cannot be precomputed, since we want to apply the measure to new messages as they arrive from a continuous real-time data stream. Furthermore, in the case of Twitter messages, which are bound to 140 characters, there will rarely be more than one occurrence of a specific term and it is thus not meaningful to calculate the term frequency for a single message. However, if a set of messages M is examined - e.g. within a user selected region and timeframe - one can build a localized *tf-idf* value for any term t by calculating the sum

$$\sum_{m \in M} tc_{t,m} * idd_t(loc(m)) \tag{8}$$

This equation properly reflects the relation of current prominence of the term versus its commonness at the message locations. For the sake of simplicity and computational cost, the value can be approximated by calculating the term frequency for a pseudo-document, generated by concatenating all documents in M, and multiplying it with the $idd_t(mean)$ at the centroid location $mean = \frac{1}{|M|} \sum_{m \in M} loc(m)$ of the documents.

4 Implementation

For the traditional *tf-idf* measure, it is expensive to compute the *idf*-part, as the whole corpus has to be analyzed. Therefore, the values are usually precomputed at once by iterating through the complete set of documents and terms within the corpus. Once this is done, one can quickly compute a *tf-idf*-vector for any given document by computing a term frequency value for each term $t \in d$ and multiplying it with its precomputed $idf_{t,D}$ value.

The computation of the $idd_t(x)$ values is even more expensive, because for any point x the sum of kernel-weighted distances between x and $loc(m)$ for all messages $m \in G$, and for the messages $m \in G_t$ respectively, has to be calculated. Furthermore, as we are looking at a continuous and thus infinite coordinate space, it is not feasible to precompute and store an $idd_t(x)$ value for every possible t and x.

For practical applications, however, there is no need to have an infinitely high spatial resolution of values. Therefore, a high resolution regular grid can be laid over the globe and the $td_t(x)$ and $dd_t(x)$ values can be compute at every cell center or vertex. Missing values between these points could then be calculated at runtime through interpolation. Nevertheless, to achieve cell-sizes below 0.5 kilometer at the equator line, a grid resolution of at least 80 000 × 40 000 cells would be needed for global coverage. If we assume that each value takes 4 Bytes of storage, this amounts to approximately 12 Gigabytes of data for every single term in the corpus. To counter these problems, we adhere to a grid construction strategy that is adaptive to regional requirements resulting from population density. This will be detailed in the next subsection. Section 4.2 explains how the *idd* values are actually computed for each generated grid cell using a technique called Splatting. Finally, we explain in Section 4.3 how the values are stored and quickly retrieved once they are needed by the application.

4.1 Grid Creation

In many regions of the world, like oceans, deserts or large rural areas, the occurrence of Twitter messages is sparse. At the same time a high Tweet frequency in major cities can be observed. Instead of using a regular grid with a fixed resolution, it is reasonable to use an adaptive grid with high resolution in densely populated areas and lower resolution elsewhere. This grid is

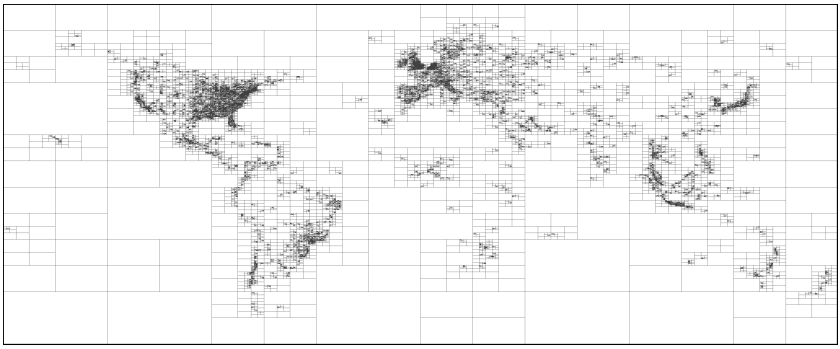


Figure 2: The result of adaptive grid creation, generated with a maximum depth of 16 and cell splitting as long as more than 50 messages were contained within a cell.

constructed by a recursive splitting mechanism generating a quadtree data structure (Finkel and Bentley, 1974). Since only Twitter users are of interest for this data set, the relevant 'Twitter population' density can be directly derived from message densities in the observed corpus. A data sample of 10 days taken uniformly distributed from one year of recorded Twitter messages was used for the construction of the grid. The initial, single cell comprises the complete coordinate space with $lat \in [-90, 90]$ and $lon \in [-180, 180]$. As long as more messages than a fixed threshold fall within one cell, we split it into four equally sized sub-cells and recursively apply the algorithm to each. A recursive path is terminated as soon as a predefined minimum cell-size of 0.5 kilometers is reached. The result of this method can be seen in Figure 2. For further computation, the complete recursive tree structure with the leaves representing the grid cells is stored. This way we can quickly ($O(\log|leaves|)$) find cells containing a given location x by recursively searching through the tree. Furthermore, a unique cell ID c_i is assigned to each cell and we use $loc(c_i)$ to denote the center-location of the cell.

4.2 Fast Value Computation

In a Gaussian kernel K_h, more than 99% of the area beneath the curve is within a $3h$-radius from the center. During the computation of the $td_t(x)$ and $dd(x)$ values, messages which are further away from x than this radius can be ignored, as they add little to nothing to the sum. Based on this observation, we chose a strategy called Splatting to realize a fast *idd* precomputation. The technique originates from volume rendering for 3D graphics (Westover, 1991). The concept can be considered as 'throwing ink balls' onto the grid at every message location, which results in a Gaussian footprint, a so-called *splat*. The local sums of the footprints at each grid-cell add up to the *td* and *dd* values. The concept is illustrated in Figure 3.

The algorithm works as follows. Let $grid = \{c_1, ..., c_i\}$ be the set of cell IDs of a grid data structure as created in Section 4.1. Furthermore, let $DD : grid \rightarrow \mathbb{R}$ and $TD_t : grid \rightarrow \mathbb{R}$ be initially empty Hash tables that map cell-ids to computed *dd* and td_t values. A given corpus G is processed according to the `splat` procedure of Listing 1. Thus, instead of iterating through the whole corpus for each grid-cell, to calculate the kernel weighted distances, we iterate through

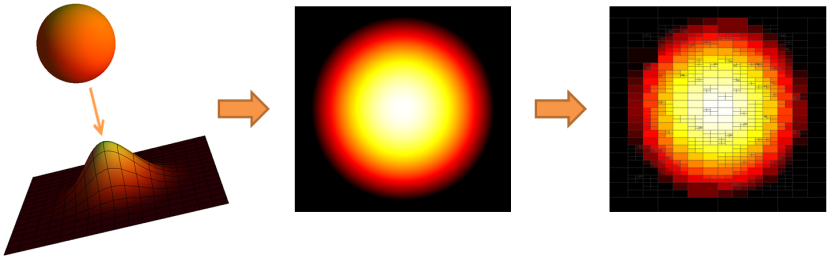


Figure 3: Left: Splatting process can be imagined as throwing ink balls onto the grid, resulting in a Gaussian signature. Middle, Right: The continuous splat is applied to the discrete grid. Each cell value is computed based on the distance from cell center to splat center.

G just once and add for each $m \in G$ and $m \in G_t$ a Gaussian splat value to all affected hash table entries $DD(c)$ and $TD_t(c)$, where the center location $loc(c)$ is within a $3h$-distance from m.

```
procedure splat(G,TD,DD,grid) is
begin
  for each m∈G do
    impact_area ← {c∈grid : d(loc(c),loc(m)) ≤ 3h}
    for each c∈impact_area do
      if DD(c)=empty then
        DD(c) ← K_h(d(loc(c),loc(m)))
      else
        DD(c) ← K_h(d(loc(c),loc(m))) + DD(c)
      end if
    end for
    for each t∈m do
      for each c∈impact_area do
        if TD_t(c)=empty then
          TD_t(c) ← K_h(d(loc(c),loc(m)))
        else
          TD_t(c) ← K_h(d(loc(c),loc(m))) + TD_t(c)
        end if
      end for
    end for
  end for
end
```

Listing 1: Splatting Algorithm

As mentioned in Section 4.1 this 'impact area' can be found quickly using the quadtree data structure. Assuming a constant upper bound for the number of terms inside a Twitter message as well as the number of cells within a splat radius, the algorithm runtime can be estimated by $O(|G| * \log(|grid|))$. Also, in terms of memory management the hash tables provide an efficient means to store the data, as large volumes of grid cells in oceans and rural areas will be unaffected by the splats of most terms. For these areas, we avoid to redundantly storing the information $idd_t(c) = 0.0$.

To limit the duration of the precomputation phase (e.g. one day instead of three) it is reasonable

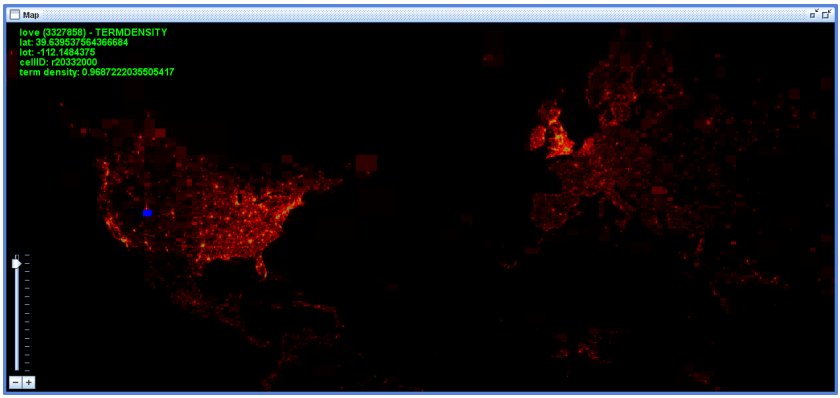


Figure 4: To evaluate the computed *TD* and *DD* results, we implemented a zoomable visualization, where individual values can be interactively examined by mouseover (blue rectangle). The result for the term density map of `love` can be seen in the window.

to restrict the computation to terms that have a certain minimum frequency. For example, in our case we included only terms that have at least 1000 mentions within a year — at application runtime, all other terms are then handled as if they occur for the first time with a default minimum term density of $td_t(x) = 1 * K_h(0)$. For evaluation purposes, the outcome of the splatting algorithm can be visualized by mapping the computed values to a color palette. The results for TD_{love} can be seen in Figure 4.

4.3 Fast Value Retrieval

The output of the splatting procedure will be a large set of filled hash tables TD_t for each term and *DD* for all documents, which can then be used for ad-hoc interpolation of the $idd_t(x)$ values at arbitrary points on the map. To actually apply the measure, two different modes were implemented. The first mode attaches the computed *idd* values directly to the twitter message as soon as it is stored. Currently, geolocated messages are collected from a continuous stream of approximately 3 – 4 million messages per day. Assuming that the number of terms per messages can be bounded by 20 terms, this means that the computation of one $idd_t(x)$ value has to be achieved in less than approx. 20 to 30 milliseconds. For future implementations the computation needs to be even faster, due to the still heavily increasing number of Twitter users.

The second mode is to retrieve the *idd* values only when they are needed inside an interactive application. In this case the retrieval from memory has to be fast enough to avoid interrupting fluid interactivity for the user. Thus it depends on the application what this means in terms of retrieval speed, i.e. for how many messages the value has to be computed in one interaction step. To actually store and fetch the values we used Apache Lucene[3], a widely known text search and storage system, and a custom built storage and query management. These methods will be reviewed in the next section.

[3] `http://lucene.apache.org/core/`

5 Implementation Performance

To evaluate its performance, the implementation was tested on a compute server driven by four Intel Xeon processors totaling to 40 physical cores, 128GB RAM, and SAS hard drives in a RAID 50 configuration. The *idd* values were precomputed based on a set of 732 895 428 geolocated Twitter messages collected between August 2011 and August 2012. The splatting algorithm described in Section 4.2 can easily be modified for parallel execution, such that it could fully benefit from the multicore capabilities of the test system. For a maximum depth of 18, the adaptive grid, described in Section 4.1, was created in less than 30 minutes and has about 300 000 cells. Based on this configuration, the complete precomputation process for the *TD* and *DD* tables was performed in less than 35 hours and it took approximately 200 Gigabytes to store the raw output.

A set of 1000 terms, drawn according to their overall frequency in the corpus, and 1000 randomly chosen cells of the grid were used to measure the retrieval speed of our storage solution. The adaptive grid tries to keep the amount of documents in each cell relatively constant; therefore the random cell node selection roughly reflects the document distribution. Because all terms share the same grid, only one instance of the quadtree needs to be held in memory for computing the cell ID for a given point x. The actual data is stored as mappings from cell IDs to term density values. We use Apache Lucene for storage and fast access to the precomputed $td_t(x)$ map. Each combination of term, cell ID, and *td* value was indexed by Lucene as a standalone *document*[4]. In order to access a value, the index for the term and cell ID combination is used to retrieve the document containing the value. This process takes 46ms on average (48ms for hits, 35ms for misses). Using parallelization with twenty threads, our implementation achieves an average of 4.8ms per value retrieval.

6 Case Studies and Evaluation

The practical usefulness of our measure was evaluated with scenario studies based on actual events and real world Twitter data. For this purpose the measure was integrated into existing tools for spatiotemporal text analysis and aggregation. In the following case studies we demonstrate how an analyst can employ and benefit from improved highlighting and term filtering based on the term normalization. Furthermore, we examine how our measure performs compared to traditional approaches.

6.1 Comic-Con 2012

The San Diego Comic-Con International is one of the largest annual conventions for comic books, science fiction/fantasy and other popular arts. From July 12 to July 15 2012 the main event was held at the San Diego Convention Center and several smaller events were co-located in nearby hotels and other venues. With more than 130 000 visitors swarming the area, observing and reporting show acts, autograph signings, meetings and other activity in the metropolitan area, the collected social media data is a perfect playground for large scale situation analysis.

For our case study we examined a set of 37 937 geolocated messages that were written in the San Diego area between 07/12 and 07/15 using the Visual Analytics system ScatterBlogs. Some of the system's core features integrate automated NLP methods for interactive visual analysis of aggregated text data. Here, we employed the *content lens*, a movable circle used to explore spatially plotted message contents by visualizing the most prominent terms. The

[4]Lucene stores documents as individually indexed string or numeric fields

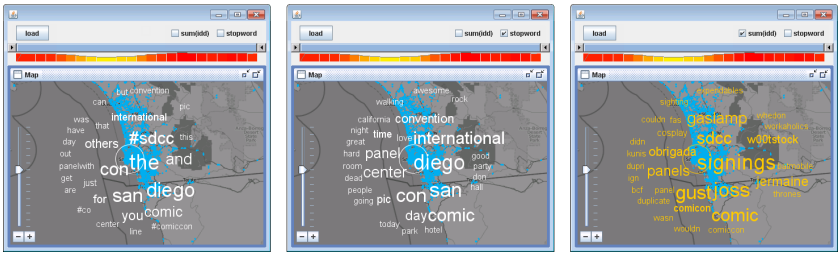


Figure 5: In ScatterBlogs a circular lens can be dragged over the map to spatially explore aggregated message contents. Left: Terms from messages inside the lens are shown according to descending occurrences. Middle: An English stopword list is used to remove irrelevant terms. Right: The *idd*-sum is used to weight the terms.

result of applying this technique to the San Diego area can be seen in Figure 5. The screenshot on the left shows the lens operating in standard mode, i.e. all terms from messages within the selected spatial and temporal frame are extracted, counted, and then arranged with descending popularity. By fixing the lens to a specific location and selecting one of the labels, the corresponding messages are shown in a table and can be further investigated by the user. In most cases, however, the standard mode will not be very useful, as it is dominated by high frequency stopwords like `the` and `you`. The user can thus activate two filtered operating modes by selecting one of the checkboxes. In *stopword*-mode, as depicted on the image in the middle, a list of English stopwords is used to remove them from the aggregation. In this mode, at least some terms related to the event (`panel`, `international`, `comic`) achieve a high ranking and can be seen in the visualization. However, the top ranking terms are still dominated by terms of regional prominence (`san`, `diego`, `center`).

The image on the right shows the *content lens* with activated measure as described in Sections 3 and 4. Here one can see several terms indicating individual smaller sub-events and activities in connection with Comic-Con. Some selected examples are:

- `w00tstock` - The music and arts show `w00tstock` was co-located with the event.
- `signings` - Artists, authors and actors gave autographs in special signing sessions.
- `joss` - Joss Whedon (an American screenwriter) gave a press conference at that day.
- `batmobile` - Props from the batman movies were on display near the convention center.

Between the hundreds of high frequency terms, these terms only achieve such a high ranking because their high prominence for the spatiotemporal frame contrasts their average prominence within the region. Such sub-event indicators are often what an analyst is interested in, when examining localized large scale events and catastrophes.

6.2 Virginia Earthquake

In August 2011 the US state Virginia was hit by a magnitude 5.8 earthquake near Richmond. The first Tweets reporting the incident were written just seconds after the shockwave occurred.

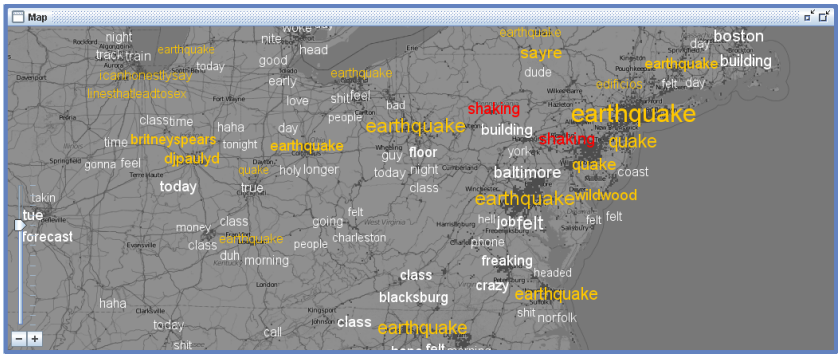


Figure 6: The *tag map* visualization: A k-means clustering scheme is employed to detect and display message clusters with similar timestamp, location and topic. Orange highlighting shows clusters/terms with high average *idd*.

To analyze this event, we use another tool of ScatterBlogs called *tag map*. It detects spatiotemporally dense clusters of messages with similar term usage and visualizes them by placing representative labels on the map. It follows the intuition that messages written by groups of local event observers will often result in this kind of *spatiotemporal content anomalies* (Thom et al., 2012) in the message streams. These anomalies can be used as an overview of the dataset and help to indicate suitable entry points for deeper investigation. Although the mechanism helps to detect many localized characteristics, it cannot distinguish whether a cluster results from a sudden and unexpected incident or from a regular venue - e.g. there could be a daily occurring cluster of messages containing `apple` at the location of an Apple store. Thus the overview showing several relevant terms is heavily cluttered with irrelevant terms that hinder the analysis.

To enhance the technique with our measure, a color coding was introduced that highlights topic clusters with high average *idd* values in orange. The threshold for highlighting is predefined based on the z-score of the *idd* value for all events. However, the analysts can also configure it to their needs using interactive controls. The result of applying the measure to a frame around the 23th of August can be seen in Figure 6. One can observe that relevant terms like `quake` and `shaking`, which are related to the sudden earthquake event, clearly stand out, while other prominent terms like `baltimore` and `blacksburg` are shown in the default color white. Besides the earthquake related terms one can also observe that `britneyspears` and `djpaulyd` are highlighted, because the artists happened to perform in Indianapolis during the examined timeframe.

6.3 Evaluation

We compared the results of our location aware measure against raw prominence and global *tf-idf* based term rankings for our observed scenario studies. For each scenario we used all messages within a predefined timeframe and area and computed the three measures for every term. For the 1000 most frequent terms a decision was made, whether they are specifically relevant for

the examined scenario. Based on this labeling, we determined incremental precision/recall curves by ranking the terms according to the respective measure's values. An example for the Comic-Con case can be seen in Figure 7.

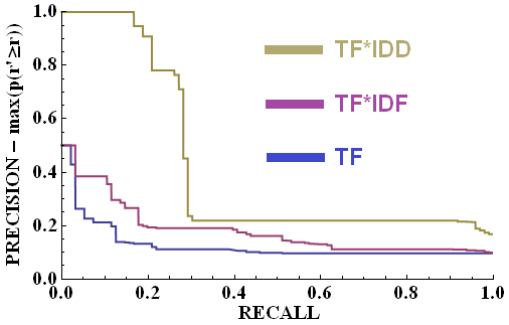


Figure 7: The graph shows interpolated precision ($p_{interp}(r) = \max_{r' \geq r} p(r')$) versus recall for Comic-Con 2012. From the messages written in the San Diego area the 1000 most prominent terms had to be ranked (96 were labeled as being relevant).

Our results show that our measure usually performs better in terms of precision for the top ranked terms and equally well or better for the lower ranked terms.[5] It is thus well suited for relevance based visualizations, where an analyst must quickly find interesting query-terms that could qualify for further investigation.

Conclusion and Perspectives

In this work a smooth measure for estimating term occurrence probabilities on a global scale was presented. Based on density adaptive large data aggregation, our measure can be employed in highly interactive environments and is suitable for real-time processing of streaming data. Because of the focus on term densities, our approach can be used in multilingual scenarios like the global Twitter corpus, as it is mostly language independent. It therefore eliminated the need for language specific stopword lists in the presented scenarios. Our case studies and evaluation have shown that the integration of the measure can lead to significant improvements for visual analysis systems.

Future work will encompass research in filtering techniques like loess smoothing to distinguish between seasonal, trending and unusual aspects of the data. In this context we will also examine means for explorative parameter steering to allow an interactive adaption of the measure's influence to tune results to the analysts' needs.

Acknowledgments

This work was funded by the German Federal Ministry for Education and Research (BMBF) and the European Commission as part of the VASA project and the FP7-project PESCaDO (FP7-248594). We would like to thank the reviewers for their valuable suggestions and comments.

[5]The relevance labeling and computed measure values for the scenario studies as well as further evaluation results can be found at `http://www.vis.uni-stuttgart.de/~thomds/iddeval/`

References

Ahern, S., Naaman, M., Nair, R., and Yang, J. H.-I. (2007). World Explorer: Visualizing aggregate data from unstructured text in geo-referenced collections. In *Proceedings of the 7th ACM/IEEE-CS joint conference on Digital libraries*, JCDL '07, pages 1–10, New York, NY, USA. ACM.

Bosch, H., Thom, D., Wörner, M., Koch, S., Püttmann, E., Jäckle, D., and Ertl, T. (2011). ScatterBlogs: Geo-spatial document analysis. In *Visual Analytics Science and Technology, 2011. VAST 2011. IEEE Conference on*.

Chew, C. and Eysenbach, G. (2010). Pandemics in the age of : Content analysis of tweets during the 2009 H1N1 outbreak. *PLoS One*, 5(11).

Eisenstein, J., O'Connor, B., Smith, N. A., and Xing, E. P. (2010). A latent variable model for geographic lexical variation. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, EMNLP '10, pages 1277–1287, Stroudsburg, PA, USA. Association for Computational Linguistics.

Finkel, R. and Bentley, J. (1974). Quad trees — a data structure for retrieval on composite keys. *Acta informatica*, 4(1):1–9.

Heverin, T. and Zach, L. (2010). Microblogging for crisis communication: Examination of Twitter use in response to a 2009 violent crisis in seattle-tacoma, washington area. In *Proceedings of the 7th International ISCRAM Conference–Seattle*.

Hughes, A. and Palen, L. (2009). Twitter adoption and use in mass convergence and emergency events. *International Journal of Emergency Management*, 6(3):248–260.

Jones, K. (1972). A statistical interpretation of term specificity and its application in retrieval. *Journal of documentation*, 28(1):11–21.

MacEachren, A., Jaiswal, A., Robinson, A. C., Pezanowski, S., Savelyev, A., Mitra, P., Zhang, X., and Blanford, J. (2011). SensePlace2: GeoTwitter analytics support for situational awareness. Providence, RI. IEEE Conference on Visual Analytics Science and Technology.

Manning, C. D., Raghavan, P., and Schtze, H. (2008). *Introduction to Information Retrieval*. Cambridge University Press, New York, NY, USA.

Marcus, A., Bernstein, M., Badar, O., Karger, D., Madden, S., and Miller, R. (2011). TwitInfo: Aggregating and visualizing microblogs for event exploration. In *Proceedings of the 2011 annual conference on Human factors in computing systems*, pages 227–236. ACM.

Mendoza, M., Poblete, B., and Castillo, C. (2010). Twitter Under Crisis: Can we trust what we RT? In *Proceedings of the First Workshop on Social Media Analytics*, pages 71–79. ACM.

Palen, L., Vieweg, S., Liu, S., and Hughes, A. (2009). Crisis in a networked world: features of computer-mediated communication in the april 16, 2007, virginia tech event. *Social Science Computer Review*.

Parzen, E. (1962). On Estimation of a Probability Density Function and Mode. *The Annals of Mathematical Statistics*, 33(3):1065–1076.

Roller, S., Speriosu, M., Rallapalli, S., Wing, B., and Baldridge, J. (2012). Supervised text-based geolocation using language models on an adaptive grid. In *EMNLP-CoNLL*, pages 1500–1510. ACL.

Rosenblatt, M. (1956). Remarks on some nonparametric estimates of a density function. *The Annals of Mathematical Statistics*, 27(3):832–837.

Sakaki, T., Okazaki, M., and Matsuo, Y. (2010). Earthquake shakes Twitter users: real-time event detection by social sensors. In *Proceedings of the 19th international conference on world wide web*, pages 851–860. ACM.

Serdyukov, P., Murdock, V., and van Zwol, R. (2009). Placing flickr photos on a map. In *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '09, pages 484–491, New York, NY, USA. ACM.

Sinnott, R. W. (1984). Virtues of the Haversine. *Sky and Telescope*, 68(2):159+.

Starbird, K. and Palen, L. (2010). Pass it on?: Retweeting in mass emergency. In *Proceedings of the 7th International ISCRAM Conference, Seattle, WA*.

Thom, D., Bosch, H., Koch, S., Wörner, M., and Ertl, T. (2012). Spatiotemporal anomaly detection through visual analysis of geolocated Twitter messages. In *IEEE Pacific Visualization Symposium*.

Westover, L. A. (1991). *Splatting: a parallel, feed-forward volume rendering algorithm*. PhD thesis, Chapel Hill, NC, USA. UMI Order No. GAX92-08005.

Wing, B. P. and Baldridge, J. (2011). Simple supervised document geolocation with geodesic grids. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1*, HLT '11, pages 955–964, Stroudsburg, PA, USA. Association for Computational Linguistics.

Zhang, D., Ooi, B., and Tung, A. (2010). Locating mapped resources in web 2.0. In *Data Engineering (ICDE), 2010 IEEE 26th International Conference on*, pages 521–532. IEEE.

Efficient Discrimination Between Closely Related Languages

Jörg TIEDEMANN[1] *Nikola LJUBEŠIĆ*[2]
(1) Department of Linguistics and Philology, Uppsala University,
Box 635, SE-751 26 Uppsala, Sweden
(2) Faculty of Humanities and Social Sciences, University of Zagreb,
Ivana Lučića 3, 10000 Zagreb, Croatia
jorg.tiedemann@lingfil.uu.se, nikola.ljubesic@ffzg.hr

ABSTRACT
In this paper, we revisit the problem of language identification with the focus on proper discrimination between closely related languages. Strong similarities between certain languages make it very hard to classify them correctly using standard methods that have been proposed in the literature. Dedicated models that focus on specific discrimination tasks help to improve the accuracy of general-purpose language identification tools. We propose and compare methods based on simple document classification techniques trained on parallel corpora of closely related languages and methods that emphasize discriminating features in terms of blacklisted words. Our experiments demonstrate that these techniques are highly accurate for the difficult task of discriminating between Bosnian, Croatian and Serbian. The best setup yields an absolute improvement of over 9% in accuracy over the best performing baseline using a state-of-the-art language identification tool.

KEYWORDS: language identification, language discrimination, closely related languages.

Proceedings of COLING 2012: Technical Papers, pages 2619–2634,
COLING 2012, Mumbai, December 2012.

1 Introduction

Language identification becomes increasingly important in applications and research that rely on data collected from the web and user contributed content. The increased interest in automatic classification of texts can be seen in the number of tools and publications related to this task. Rather simple statistical techniques based on character N-grams and other orthographic features have been shown to be very effective even for the distinction of quite a large number of different languages. However, closely related languages are much harder to distinguish (even for humans) and standard approaches usually fail badly. Furthermore, low-density languages that are strongly related to other languages with larger resources are often not supported by language identifiers with pre-trained models. Some popular tools, such as the language detector integrated in the Google Chromium project, cannot be trained at all and, therefore, cannot be extended easily.

For these reasons, we consider in this paper the specific task of discriminating between closely related languages as a sub-task in automatic language identification. We will show that dedicated, yet simple models can greatly contribute to a better classification of those languages and, in this way, can help to build up resources for low-density languages. In our experiments, we focus on three related south-slavic languages: Bosnian, Croatian and Serbian. They represent a prototypical example for the problem we like to concentrate our work on. Their genetical closeness and lexical similarities make it very hard to distinguish between texts written in either language, which we will see further down in our initial experiments with standard language identification tools.

2 Related work

The problem of language identification was for quite some time considered a rather easy one and mostly solved. One of the most frequently used system in the academic world was TextCat based on the algorithm described in (Cavnar and Trenkle, 1994). This system uses character n-grams as features of length 1<=n<=4 that are most frequent in the specific language. Beside character N-grams, some systems also use the most frequent words as features (Batchelder, 1992).

With the dramatic increase of multilingual data on the web, the language identification problem has received new attention. Additionally, with the rise of new domains, such as microblogging, domain adaptation for language identification has become an important problem. Lui and Baldwin (2011), for example, tackle the problem by calculating information gain over multiple domains choosing those character n-gram features that are the most domain independent ones.

A harder problem emerging with the availability of data in many languages is the problem of discriminating between closely related languages. However, only a few researchers dealt with that problem in the past. Padró and Padró (2004) report problems with the distinction of Catalan and Spanish when using standard language identification methods showing that a character-based Markov chain is the least erroneous one having an error rate of roughly 6% on very short texts and 1% on longer ones, This accounts for 40% of errors on a task of discriminating between six languages. Furthermore, the inability to discriminate Portugese and Brazilian Portuguese with TextCat has been reported by Martins and Silva (2005). Finally, in our previous work (Ljubešić et al., 2007) we discuss the identification of Croatian in a pool of Slovenian, Croatian and Serbian documents showing very good results with a second-order Markov chain with 100% accurate discrimination of Slovene and a 96% accurate discrimination

between Croatian and Serbian. We also show that the latter could be improved by using simple lists of forbidden words to achieve up to 99% accuracy. These findings, in particular, motivated us to further explore the identification of these three languages with the techniques we present below.

Let us first have a quick look at the differences between the languages we focus on before introducing our approaches for language discrimination.

2.1 Differences between Bosnian, Croatian and Serbian

The three languages are considered very similar and until recently there was an open discussion if these languages should be considered linguistically separate.

If we consider the usage of Latin script only,[1] the largest difference between Croatian and Serbian is the present form of the proto-Slavic vowel *jat*, resulting in Serbian following more the ekavian and Croatian the ijekavian reflex which introduces many lexical differences (*child* – *dete* (sr) vs. *dijete* (hr, bs)). In this feature, Bosnian generally follows Croatian.

Internationalisms and proper nouns are handled differently as well with transliterations being frequent in Serbian, less frequent in Bosnian while in Croatian foreign proper names from Latin-script languages are written in the original orthography (*Nju Jork* (sr,bs) vs. *New York* (hr)).

In morphology the most important difference is the existence of synthetic future tense in Serbian (*videću*) beside the analytical one (*vidjet ću* or *ću vidjeti*) while Croatian and Bosnian only use the analytical form. There are also systematic differences in the derivation of nouns, adjectives and verbs (*organizovati*, *organizovan* (sr,bs) vs. *organizirati*, *organiziran* (hr,bs)) with Bosnian using both again.

There are some syntactic differences as well. The most visible one is the structure *modal verb + da + present* in Serbian and *modal verb + infinitive* in Croatian (*hoću da radim* (sr, bs) vs. *hoću raditi* (hr, bs)), Bosnian allowing both. Finally, Croatian and Serbian show a series of differences in the general vocabulary (*fabrika* (sr, bs) vs. *tvornica* (hr,bs)), Bosnian again allowing both, but additionally introducing a lot of lexical units culturally bound with the Moslem world. Apart from the fact that these three languages have common origins, today they are three distinct and codified standards, and texts in these standard languages appear regularly.[2]

Being aware that this is an oversimplification, we conclude that Croatian and Serbian are visibly different languages while Bosnian is a mixture of the two with a tendency towards Croatian. To support this claim, we present in table 1 the overlap of lowercased tokens calculated on the parallel corpus used later on for training our classifiers. We can observe that Croatian and Serbian are the most different languages, Bosnian and Serbian coming second and Bosnian and Croatian being the most similar ones. By not calculating type, but token intersections, we are also able to observe the amount of symmetry in the token overlap inside language pairs. The largest difference in the token overlap is shown between Croatian and Serbian showing 5%

[1] In contemporary Serbian around 50% of texts appear written in Cyrillc, Bosnian much less, while Croatian doesn't use the Cyrillc script at all.

[2] In our research we did not take into account the appearance of a new Montenegrin standard, with its reformed orthography that makes it distinct from any of these three languages. This development appeared only recently and this language should be considered as an extremely low-density language (total number of speakers is less than half a million).

more tokens of Serbian appearing in Croatian texts than vice versa. The most likely reason for this phenomenon is the tendency of Croatian language to break from the tradition of language unification efforts that existed in Yugoslavian times.

	bs	hr	sr
bs		0.952	0.915
hr	0.950		0.857
sr	0.930	0.902	

Table 1: Overlap of lowercased tokens between languages in the parallel corpus of Bosnian (bs), Croatian (hr) and Serbian (sr). Rows represent tokens, columns corpora of specific languages.

In order to stress the problems in automatic discrimination between these languages, we trained models using three popular tools (TextCat[3], Lingua::Identify[4] and langid.py[5]) with data taken from SETimes[6], a collection of parallel texts from an on-line news portal for the Southeast European region that publishes "news and views from Southeast Europe" in ten languages.[7] The trilingual parallel corpus we extracted from the dataset contains 5,536 parallel documents and roughly 2.7 million words per language. We trained models for only the three languages of interest in order to avoid any further confusions of the classifiers. For evaluation purposes, we extracted and manually verified 600 documents (200 per language) from three on-line resources, one in each language.[8] Table 2 shows the confusion matrices produced by these tools.

TextCat
Overall accruacy: 55.5%

	bs	hr	sr
bs	90	36	74
hr	68	65	67
sr	14	8	178

Accuracy for Bosnian: 45%

Lingua::Identify
Overall accuracy: 48.8%

	bs	hr	sr
bs	65	117	18
hr	43	151	6
sr	41	82	77

Accuracy for Bosnian: 32.5%

langid.py
Overall accuracy: 87.7%

	bs	hr	sr
bs	139	56	5
hr	11	187	2
sr	0	0	200

Accuracy for Bosnian: 69.5%

Table 2: Special-purpose classifiers trained with standard tools for language identification: bs (Bosnian), hr (Croatian) and Serbian (sr).

As we can see, there is significant difference between the results of the three classifiers. langid.py performs much better than the other two, which is most probably due to the more sophisticated learning algorithm used in that approach (Lui and Baldwin, 2012). Namely, it uses information gain for the selection of features that discriminate best between languages. An additional remark should be made that this system focuses on domain robustness and not the problem of discriminating similar languages. On the other hand TextCat only uses the most frequent N-grams per language and Lingua::Identify uses prefixes, suffixes and frequent words and, apparently, none of these features can well discriminate similar languages such as the ones we

[3] http://www.let.rug.nl/~vannoord/TextCat/
[4] http://search.cpan.org/~ambs/Lingua-Identify-0.51/
[5] https://github.com/saffsd/langid.py
[6] http://www.setimes.com
[7] The data set is freely available from http://www.nljubesic.net/resources/corpora/setimes/.
[8] http://www.dnevniavaz.ba, http://www.vecernji.hr and http://www.politika.rs

deal with. Although langid.py performs much better than the other two systems, the overall accuracy is still much below general language identification performances reported in the literature. This is especially true for the discrimination between Bosnian and Croatian. The accuracy of 0.695 for the recognition of Bosnian texts is just not acceptable.[9]

3 Discriminating Between Closely Related Languages

In this section, we will discuss two approaches to language discrimination: One is based on a document classification method and the other one focuses on the identification of indicator features in terms of blacklisted words. Both approaches are in their essence quite similar, but they show a different performance on various amounts of training data and on data which is not entirely parallel, but comparable.

3.1 Learning a Document Classifier

The idea of using document classification techniques for language identification relies on the prerequisite of possessing parallel data of closely related languages. Since parallel texts communicate identical content, the differences in the bitext of closely related languages are exactly the differences between these two languages. By learning to discriminate between these datasets we actually learn the difference between the languages. Using document classification techniques on non-parallel data would model content alongside language specificities and is expected to perform worse than the models built on parallel data.

We chose to use the multinomial Naive Bayes classifier (McCallum and Nigam, 1998) because of its general good performance and speed. Additionally, because its parameters are actually probabilities of words given the category, i.e. language, this makes the model easily readable for humans. We estimate the model parameters as probabilities of words given the class

$$\hat{P}(w_i|c_j) = \frac{c(w_i, c_j)}{c(w_i)}$$

where $c(w_i, c_j)$ is the count for word w_i in texts of class c_j and $c(w_i)$ is the count of word w in the whole corpus.

We predict the language by maximum a posteriori class c_{map}

$$c_{map} = \arg\max_c \sum_w \log \hat{P}(w_i|c_j)$$

by summing over logarithms of probabilities of words given the class for each word in the document to be classified. We use simple add-one smoothing to take care of unseen events. Since we consider all languages equiprobable, we omit the prior probability in the procedure.

[9] We decided to refer to the performance on individual languages in terms of accuracy when applying only that subset of the data to the classifier. Alternatively, one could look at precision and recall values for individual languages computed over the entire data set. In that case, our language-specific accuracy values correspond to recall. Using the example of langid.py, precision of Bosnian would then be 0.927 and for Croatian as low as 0.77.

3.2 Learning "Blacklisted Words"

The difference between related languages can often be explained by some distinctive words that occur quite frequently in one language but never in the other language (at least not with exactly the same spelling). The use of "forbidden words" in language identification has already been shown in Ljubešić et al. (2007) as discussed in the introduction. This observation leads to the idea of building a classifier entirely based on those distinctive features that clearly discriminate between two languages. One could say that these words are on a "blacklist" for the language in which they should not appear. Observing one of those words should give very strong evidence against the language they are blacklisted for. We consider this idea to be strictly binary between two well-defined classes. Blacklists should only provide evidence to distinguish one language from exactly one other one.

Blacklists could be built manually using linguistic intuitions by native speakers. However, it is also possible to derive such data sets from corpora simply by comparing word frequencies. As stated above, we are looking for words that are (rather) common in one language but forbidden in the other. The most simplistic approach would use raw frequencies to find relatively frequent words that do not appear in texts of the other language. A simple frequency threshold could be sufficient for this purpose.

$$w \in B_2 \;\; if \;\; c_1(w) > \theta \wedge c_2(w) = 0$$

where B_2 is the set of blacklisted words for language L_2, $c_1(w)$ and $c_2(w)$ are the frequency counts of word w in language L_1 and L_2, respectively, and θ is the threshold to filter out infrequent words. These blacklists could be used as strict filters (as in spam filtering) but this would lead to (possibly a lot) of documents that are blacklisted for both languages as some of the words may still be valid for a language in which they do not appear in the training data. Certainly, it is again important to focus on texts from similar domains in order to avoid spurious signals indicating domain differences instead of language differences.

One possibility to make the classifier more robust is to use counts of matching blacklist words to, for example, classify document D:

$$lang(D) = \begin{cases} L_2 & if \;\; \sum_{w \in D \wedge w \in B_1} 1 \; > \sum_{w \in D \wedge w \in B_2} 1 \\ L_1 & \text{otherwise} \end{cases}$$

However, in this approach each blacklisted word has the same impact on the final classification and the frequency threshold θ becomes very important when collecting the data sets. Furthermore, it may not be wise to restrict the blacklists to words that do not appear at all in the other language. Certain elements (quoted expressions, names, titles) may spoil the data and useful discriminators will be missed by this strict approach.

Hence, another idea is to use the difference of relative frequencies (as an MLE-based approximation of unigram probabilities) between words appearing in one or both languages. In order to measure the difference, we define the following ratio (N_1 and N_2 are the total word counts in language L_1 and L_2, respectively):

$$\delta(w, L_1, L_2) = \frac{c_1(w)/N_1 - c_2(w)/N_2}{c_1(w)/N_1 + c_2(w)/N_2} = \frac{c_1(w)N_2 - c_2(w)N_1}{c_1(w)N_2 + c_2(w)N_1}$$

Furthermore, we restrict the candidates to words that appear less than a certain frequency threshold α in one language and more than another frequency threshold β in the other language. Yet another threshold (γ) can be set to restrict the set to the most discriminating words by adding the constraint $|\delta(w, L_1, L_2)| > \gamma$. Negative scores indicate a feature supporting L_2 and positive scores support L_1.

Using this set of weighted features, we can now define an adjusted decision rule in the following way:

$$lang(D) = \begin{cases} L_2 & if \ \sum_{w \in D} \delta(w, L_1, L_2) < 0 \\ L_1 & \text{otherwise} \end{cases}$$

Using this definition, the Blacklist classifier becomes quite similar to a two-class Naive Bayes approach but with heavy feature selection using only the most promising discriminating tokens for classification. We could even introduce yet another threshold to define a margin that describes the grey area of uncertain decisions. In that case, we would end up with a classifier that uses the following decision rule:

$$lang(D) = \begin{cases} L_1 & if \ \sum_{w \in D} \delta(w, L_1, L_2) > \mu \\ L_2 & if \ \sum_{w \in D} \delta(w, L_1, L_2) < -\mu \end{cases}$$

However, we do not apply this method in the present paper as this introduces yet another free parameter that needs to be adjusted.

Another practical consideration is to avoid spurious tokens such as proper names or tokens containing non-alphabetic characters. Some texts may frequently mention certain names, numbers, dates or other named entities that do not appear in texts of the other language. In that case, they would easily end up in blacklists without being appropriate for language discrimination. On the other hand, certain punctuation differences may also work quite well for distinguishing between languages. However, in our experiments we simply dismiss all tokens containing non-alphabetic characters.

For the discrimination between more than two languages, we can use a simple cascaded setup of pairwise classifications. The disadvantage of this procedure is that the order of classification steps may influence the final result. However, we could not see any significant impact on our results in the experiments described below. We, therefore, did not try to optimize this procedure for better discrimination.

4 Experiments

Our experiments are based on the same data sets as we have used for our baseline approaches presented in section 1. In particular, we used the portion of the parallel SETimes corpus containing Bosnian, Croatian and Serbian with its 2.7 million words per language. The evaluation data contains 200 documents per language with about 78 thousand words of Bosnian, 70 thousand words of Croatian and 113 thousand words of Serbian. Hence, the average document length is rather short – between 350 and 500 tokens per document.

In order to compare our approaches to a strong baseline, we built a second-order Markov chain as described in (Ljubešić et al., 2007) that has been proven to be very efficient for the

	bs	hr	sr	accuracy
bs	173	17	10	0.865
hr	30	170	0	0.850
sr	1	0	199	0.995

Table 3: Applying a second-order Markov chain to our test data.

discrimination between related languages such as Slovene, Croatian and Serbian. The confusion matrix of classifying our test set using this method is shown in table 3.

The overall accuracy of this approach is 90.3%. As expected, it still has major problems with distinguishing Bosnian and Croatian.

Table 4 shows the results of our proposed classifiers when applied to the test data.

Naive Bayes: overall accuracy: 95.7%

	bs	hr	sr	accuracy
bs	181	11	8	0.905
hr	7	193	0	0.965
sr	0	0	200	1.000

Blacklist Classifier: overall accuracy: 97%

	bs	hr	sr	accuracy
bs	187	12	1	0.935
hr	5	195	0	0.975
sr	0	0	200	1.000

Table 4: The confusion matrices of applying our classifiers to the test data.

For the blacklist approach, we always apply Serbian-Croatian discrimination before discriminating between the preferred language from the first step and Bosnian. The blacklisted word extraction parameters are set to intuitively chosen values: $c_{low}(w) < \alpha = 4$, $c_{high}(w) > \beta = 9$ and $|\delta| > \gamma = 0.8$. We did not perform any optimization of these settings so far, which we, however, plan to do in future work.

We can see that the overall accuracy is much higher than for the general-purpose tools even when trained for this specific task. The difference is especially striking for the case of Bosnian which could be identified with over 90% accuracy in both cases. To test the statistical significance of the differences in performance of the classifiers we use the approximate randomization procedure (Hoeffding, 1952; Yeh, 2000) with 1000 repetitions. The obtained p-values are presented in Table 5.

classifier 1	classifier 2	difference in accuracy	p-value
Naive Bayes	TextCat	0.401	**0.001**
Naive Bayes	Lingua::Identify	0.468	**0.001**
Naive Bayes	langid.py	0.080	**0.001**
Naive Bayes	Markov chain	0.053	**0.001**
Blacklist	TextCat	0.415	**0.001**
Blacklist	Lingua::Identify	0.482	**0.001**
Blacklist	langid.py	0.093	**0.001**
Blacklist	Markov chain	0.067	**0.001**
Naive Bayes	Blacklist	0.013	0.188

Table 5: Comparison of the presented approaches in terms of overall accuracy.

The difference between the Naive Bayes and Blacklist classifiers has shown not to be statistically

significant on this size of the evaluation set ($p = 0.188$), but the difference is an interesting fact that should be looked into in future work. However, the difference between the Naive Bayes and Blacklist classifiers to the other classifiers is highly statistically significant ($p < 0.001$) while the difference between our baseline Markov chain and the langid.py classifier (0.027) is marginal ($p = 0.094$).

4.1 Size of Training Data

Despite their higher performance, a possible disadvantage of our token-based classifiers (compared to classifiers based on character sequences) is that they may have larger problems when trained on limited amount of data, non-parallel texts and non-comparable domains. Figure 1 plots the learning curves for our two methods when training with various amounts of parallel data.

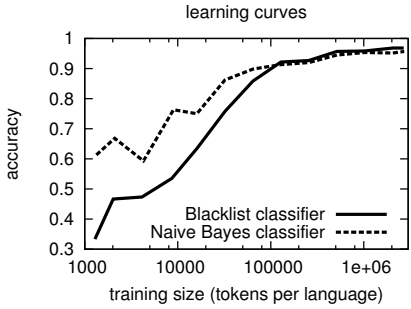


Figure 1: Learning curves of the proposed classifiers with various amounts of training data.

The figure shows that Naive Bayes is learning much quicker to distinguish between the three languages. This is not very surprising as the blacklist classifier only considers a fraction of the possible features included in the training data (see Table 6). However, after about 100,000 tokens per language, it surpasses the accuracy of the Naive Bayes classifier and performs consequently better thereafter.

It is also interesting to consider the learning curves for the individual languages. Looking a bit closer at the blacklist classifier (Figure 2), we can see that the main problem is the identification of Bosnian. It takes much more data to train a decent classifier for Bosnian than required for distinguishing the other two languages.

Interesting are also the drops in performance when recognizing Serbian in the beginning and when recognizing Croatian after about 10,000 tokens of training data. This development is due to our cascaded setup and the lack of evidence in the learned classification models. In the beginning, no (or only a few) blacklisted words are found and the classifier cannot discriminate between the three languages. Our method simply classifies every document as Serbian. After a few thousand tokens, the classifier learns to identify Croatian quickly but starts confusing it with Serbian and later with Bosnian. At about 10,000 tokens, the classifier improves significantly for Serbian again but seems to be more confused about Bosnian and Croatian before fixing most of

these problems with larger amounts of training data.

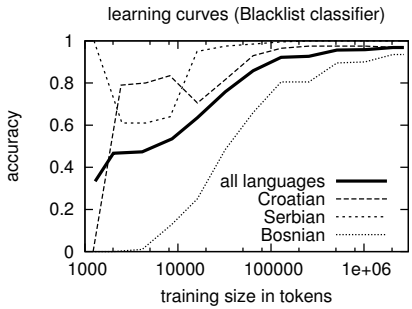


Figure 2: Learning curves for individual languages in the Blacklist classifier.

The learning curves above reveal one of the main weaknesses of the blacklist approach. It is not reliable with very little amounts of training data and the results will significantly depend on the setup of the binary decisions in that case. Table 6 lists the total number of tokens included in the blacklists when using default settings. For example, after about 32,000 tokens only a few hundred words are selected, which does not seem to be sufficient according to Figure 1. However, only a few thousand words selected at about 128,000 tokens of training material perform pretty well for all three languages. This is a very promising result. Furthermore, the amount of selected words for classification can certainly be adjusted by the extraction parameters α, β and γ. Initial experiments show that less restrictive parameters lead to better accuracies with small amounts of training data but lower overall performance when using the entire training set. This is also expected as the classifier becomes more similar to the Naive Bayes classifier and its parameters. We will leave a proper optimization of these model parameters to future work.

size in number of tokens		CPU time in seconds	
train data	blacklists	training	classification
1k	2	0.06	1.35
2k	5	0.08	1.38
4k	10	0.14	1.39
8k	32	0.27	1.40
16k	95	0.53	1.41
32k	361	1.01	1.40
64k	867	2.08	1.44
128k	1766	4.08	1.46
256k	3463	8.25	1.49
512k	6280	16.72	1.53
1M	10522	32.96	1.50
2M	16707	66.60	1.61
2.5M	19261	93.93	1.70

Table 6: Size of the learned blacklists and time spent for training and classification.

Table 6 also lists the time spent for training blacklist models with various amounts of training data and the time required for classifying our test set with those models. We can see that training is fast[10] even with our unoptimized code (based on a scripting language) and classification time is almost constant with some overhead because of the increasing size of the extracted models. The simplicity of the blacklist approach allows very efficient computation and produces compact models with high accuracy. This is certainly one of the major advantages of this approach compared to more sophisticated machine learning techniques.

4.2 Parallel versus Comparable Training Data

An additional question we like to consider in this paper is how robust our classifiers are with respect to non-parallel data. To simulate a loss in comparability we have split our trilingual dataset into three folds. In the first setting we train three models on parallel data while in the second setting we train models on the six permutations of non-parallel data so that for neither language there is any parallel data in the training set[11].

We evaluated each of the 3+6 models for both classifiers on our standard test set. The results are given in Table 7. With a loss of comparability of the training data we see an average decrease in performance of 1.6 points for the Naive Bayes classifier while there is no decrease in the case of the Blacklist classifier. The difference in the results of the two settings when using the Naive Bayes classifier is statistically significant according to the one-sided Student's t-test with unequal variance (p-value is 0.029). Furthermore, the difference between the results of the two classifiers on comparable corpora has become highly statistically significant (p-value below 0.001). Here we have identified a strong point of the Blacklist classifier – it is much more resistant to non-parallelity of the data when compared to the Naive Bayes classifier.

	Naive Bayes	Blacklist	p-value
parallel	0.953	0.963	0.233
comparable	0.937	0.963	**0.001**
p-value	**0.029**	0.875	

Table 7: Results for parallel and comparable training sets for both classifiers with the p-value from the Student's t-test.

The reason for such results can be sought in the fact that the Blacklist classifier does generalize more by selecting only the most informative features while this implementation of the Naive Bayes classifier does not perform any feature selection at all.

4.3 Sensitivity with Respect to Document Size

Finally, we would also like to investigate the influence of document size on classification performance. We expect that our word-based classifiers require larger amounts of input data for reliable classification. This is especially true for the blacklist approach that relies on strong discriminative features that might not be very frequent in all kinds of documents. Figure 3 illustrates the overall accuracy with varying document sizes. For this experiment, we selected all documents with more than 300 words from our initial test set (giving us 100 Bosnian

[10] Note that training speed is not as essential as classification speed as training is usually done once only.

[11] In the first setting we train on folds (0,0,0), (1,1,1) and (2,2,2) while in the second one we train on (0,1,2), (1,2,0) etc.

documents, 89 Croatian documents and 182 Serbian documents) and used the initial N words of each of them for classification.

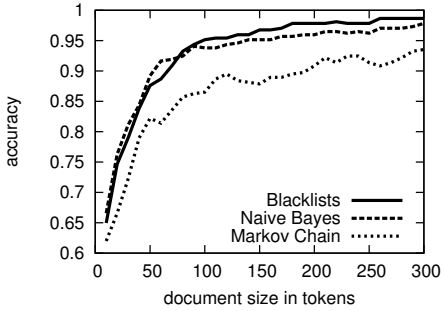


Figure 3: Classification performance with various text sizes.

As expected, we can see a significant decline of the accuracy with very short documents. However, already at about 70 words we have an overall performance of over 90%. Again, proper identification of Bosnian texts is the hardest task and about double as many words are needed to reach 90% accuracy for the Bosnian test set (not shown in Figure 3). In general, a modest document size of 150–200 words seems to be quite sufficient for our difficult task.

Another important result is that both classification approaches outperform our strongest baseline (the character-based Markov chain model) at all points. Furthermore, the blacklist approach is slightly worse on short documents, which was also to be expected.

4.4 Inspecting Language Discriminators

An interesting byproduct of both language identification approaches described above is that the parameters of their models are actually conditional probabilities and weights calculated on words and categories. This gives us the opportunity to inspect the strongest discriminators between these three languages and to classify them by the differences described in subsection 2.1.

In table 8 we list the twenty five highest conditional probabilities (and thereby strongest discriminators) of the final Naive Bayes classifier. In Bosnian there are only five conditional probabilities with a maximum value while in Croatian there are 34 of such words that do not appear in training corpora of other languages. In the Serbian model we have 21 maximum probabilities. This is in line with the previous claim from subsection 2.1 and table 1 that Croatian is the most specific language while Bosnian is a mixture of the other two.

The twenty five strongest Bosnian discriminators contain seven lexemes where Bosnian differs from Serbian regarding the ijekavian reflex like *obezbijediti* (*obezbediti* in Serbian) and *posjetioc* (*posetioc*). The latter form is written in Croatian with a different suffix morpheme – *posjetitelj*. Another difference to Serbian is the internationalism *historija* written *istorija* in Serbian while Croatian has its own word – *povijest*.

Bosnian		Croatian		Serbian	
sedmice	1.0	tjedna	1.0	evra	1.0
saopćenju	1.0	glede	1.0	sredu	1.0
izvještajima	1.0	izvješću	1.0	izveštaju	1.0
augusta	1.0	listopada	1.0	bezbednosti	1.0
saopćio	1.0	veljače	1.0	saveta	1.0
saopćila	0.999	siječnja	1.0	euleks	1.0
izvještaja	0.999	posebice	1.0	posete	1.0
obezbijediti	0.999	ožujka	1.0	bezbednost	1.0
sedmica	0.999	tvrtke	1.0	verovatno	1.0
saopćeno	0.999	prosinca	1.0	vestima	1.0
historiji	0.999	svibnja	1.0	predsednikom	1.0
istambulu	0.999	lipnja	1.0	savet	1.0
saopćili	0.999	srpnja	1.0	potpredsednik	1.0
unaprjeđivanju	0.999	rujna	1.0	cena	1.0
historijski	0.998	travnja	1.0	cene	1.0
historije	0.998	gospodarstva	1.0	vrednosti	1.0
augustu	0.998	rumunjskoj	1.0	dve	1.0
odista	0.998	tvrtka	1.0	organizovanog	1.0
historiju	0.998	izvješće	1.0	sledeće	1.0
posjetioci	0.998	priopćenju	1.0	zahtev	1.0
istambula	0.998	ravnatelj	1.0	ren	1.0
bezbjednost	0.998	gospodarstvo	1.0	nemačka	0.999
djelimično	0.998	priopćila	1.0	posetio	0.999
sedmicu	0.998	sustava	1.0	severnom	0.999
unaprjeđivanja	0.998	konca	1.0	poseti	0.999

Table 8: Twenty five highest conditional probabilities of the Naive Bayes classifier for each language

Croatian's strongest discriminators contain ten month names (*listopad*, *veljača* etc.) since Croatian uses its own names and Bosnian and Serbian use the international forms. The word *priopćiti* shows a difference in derivational morphology compared to Bosnian where we have *saopćiti*. The list contains also many words of Croatian origin like *ravnatelj*, *gospodarstvo* and *tvrtka* where in Bosnian and Serbian internationalisms or terms from the Yugoslav era are used.

The Serbian list contains many ekavian variants like *sreda* (written *srijeda* in Bosnian and Croatian), *savet* (*savjet*) and *vest* (*vijest*). A frequent morphological difference to Croatian and Bosnian is present in the word *organizovan* that would be formed as *organiziran* in these two languages.

It is important to note that not all differences caught by these models are actual differences in language use or language norm, but are sometimes just the result of the language policy applied on the website the dataset comes from. A good example is the word *izvještaj* from the Bosnian list that obviously never appears in any other language although it is regularly used in Croatian language and Croatian dictionaries define it as standard as well.

Conclusion and Future Work

In this paper, we have shown that language discrimination between closely related languages deserves special attention. Standard tools based on character sequence features are not sufficient for distinguishing languages with a large lexical overlap. We propose two token-based approaches, one based on a Naive Bayes classifier and one based on weighted lists of blacklisted words. Both perform very well and significantly outperform state-of-the-art approaches to language identification. Another conclusion from our experiments is that a Naive Bayes model performs better for smaller amounts of data but highly depends on the comparability of the language data it is trained on. The blacklist approach is similar in essence but includes heavy feature selection. This leads to a larger generalization of the model and makes it perform better on less parallel data sets. The overall performance of the blacklist approach is also higher given the entire data set we train on and improves the best baseline created using public language identification tools by over 9% absolute accuracy. The implementations of the two approaches along with the datasets used in the paper can be retrieved from `http://www.nljubesic.net/resources/tools/bs-hr-sr-language-identifier/` and `http://bitbucket.org/tiedemann/blacklist-classifier`.

In this work we were using parallel data, but did not exploit all the benefits one can expect from such a dataset. We did not use sentence and word alignments, but just the corpora as a whole having in mind that the frequencies of identical language elements across languages will agree very well. Using alignments from parallel corpora is one direction for future work.

Another direction is the opposite one: using weakly comparable, but larger corpora to obtain the same or better results. We have shown that the blacklist approach is very robust on strongly comparable corpora, but additional experiments are necessary to examine how it copes with lower comparability of the training data. On the other hand, the Naive Bayes classifier could be made less prone to non-parallelity as well by applying different feature selection methods.

Furthermore, using very large web corpora such as the hrWaC corpus (Ljubešić and Erjavec, 2011) 1.2 billion words in size and the bsWaC and srWaC corpora (under construction) opens the door for catching most of the token and N-gram domain-independent variation between closely related languages that could yield extremely high results we are used to for more distinct languages.

Finally, we would also like to look at recently proposed character-based language models that have successfully been used for the discrimination between language varieties (Zampieri and Gebre, 2012). Their approach is similar to our Markov chain baseline but uses larger character N-grams that often cover entire words. One of the disadvantages of this approach is the increase of the number of model parameters, blowing up the model size and causing a slower classification speed. However, one could hope for higher generalization and resistance to non-parallelity of the training data.

Acknowledgements

Research reported in this paper has been supported by the EU FP7 ICT-PSP project LetsMT!, grant agreement no. 250456, and the EU FP7 STREP project XLike, grant agreement no. 288342.

References

Batchelder, E. O. (1992). A learning experience: Training an artificial neural network to discriminate languages. Unpublished technical report.

Cavnar, W. B. and Trenkle, J. M. (1994). N-gram-based text categorization. In *Proceedings of SDAIR-94, 3rd Annual Symposium on Document Analysis and Information Retrieval*, pages 161–175.

Hoeffding, W. (1952). The large-sample power of tests based on permutations of observations. *Annals of Mathematical Statistics*, 23(2):169–192.

Ljubešić, N. and Erjavec, T. (2011). hrWaC and slWac: Compiling Web Corpora for Croatian and Slovene. In Habernal, I. and Matousek, V., editors, *Text, Speech and Dialogue - 14th International Conference, TSD 2011, Pilsen, Czech Republic, September 1-5, 2011. Proceedings*, volume 6836 of *Lecture Notes in Computer Science*, pages 395–402. Springer.

Ljubešić, N., Mikelić, N., and Boras, D. (2007). Language identification: How to distinguish similar languages. In Lužar-Stifter, V. and Hljuz Dobrić, V., editors, *Proceedings of the 29th International Conference on Information Technology Interfaces*, pages 541–546, Zagreb. SRCE University Computing Centre.

Lui, M. and Baldwin, T. (2011). Cross-domain feature selection for language identification. In *Proceedings of 5th International Joint Conference on Natural Language Processing*, pages 553–561, Chiang Mai, Thailand. Asian Federation of Natural Language Processing.

Lui, M. and Baldwin, T. (2012). langid.py: An off-the-shelf language identification tool. In *ACL (System Demonstrations)*, pages 25–30.

Martins, B. and Silva, M. J. (2005). Language identification in web pages. In *Proceedings of the 2005 ACM symposium on Applied computing*, SAC '05, pages 764–768, New York, NY, USA. ACM.

McCallum, A. and Nigam, K. (1998). A comparison of event models for naive bayes text classification. In *IN AAAI-98 WORKSHOP ON LEARNING FOR TEXT CATEGORIZATION*, pages 41–48. AAAI Press.

Padró, L. and Padró, M. (2004). Comparing methods for language identification. *Procesamiento del Lenguaje Natural*, (33):155–162.

Yeh, A. (2000). More accurate tests for the statistical significance of result differences. In *Proceedings of the 18th conference on Computational linguistics - Volume 2*, COLING '00, pages 947–953, Stroudsburg, PA, USA. Association for Computational Linguistics.

Zampieri, M. and Gebre, B. (2012). Automatic identification of language varieties: The case of Portuguese. In Jancsary, J., editor, *Proceedings of KONVENS 2012*, pages 233–237. ÖGAI. Main track: poster presentations.

Semi-Supervised Semantic Role Labeling: Approaching from an Unsupervised Perspective

Ivan Titov *Alexandre Klementiev*
Saarland University, Saarbrücken, Germany
{titov, aklement}@mmci.uni-saarland.de

ABSTRACT
Reducing the reliance of semantic role labeling (SRL) methods on human-annotated data has become an active area of research. However, the prior work has largely focused on either (1) looking into ways to improve supervised SRL systems by producing surrogate annotated data and reducing sparsity of lexical features or (2) considering completely unsupervised semantic role induction settings. In this work, we aim to link these two veins of research by studying how unsupervised techniques can be improved by exploiting small amounts of labeled data. We extend a state-of-the-art Bayesian model for unsupervised semantic role induction to better accommodate for annotated sentences. Our semi-supervised method outperforms a strong supervised baseline when only a small amount of labeled data is available.

KEYWORDS: semantic role labeling, semi-supervised learning, shallow semantics, Bayesian model.

Proceedings of COLING 2012: Technical Papers, pages 2635–2652,
COLING 2012, Mumbai, December 2012.

1 Introduction

Shallow representations of meaning, and semantic role labels in particular, have a long history in linguistics (Fillmore, 1968). More recently, with the emergence of large annotated resources such as PropBank (Palmer et al., 2005) and FrameNet (Baker et al., 1998), automatic semantic role labeling (SRL) has attracted a lot of attention (Gildea and Jurafsky, 2002; Carreras and Màrquez, 2005; Surdeanu et al., 2008; Hajič et al., 2009).

SRL representations encode the underlying predicate-argument structure of sentences, or, more specifically, for every predicate in a sentence they identify a set of arguments and associate each argument with an underlying *semantic role*, such as an agent (an initiator or doer of the action) or a patient (an affected entity). SRL representations have many potential applications in NLP and have been shown to benefit question answering (Shen and Lapata, 2007; Kaisser and Webber, 2007), textual entailment (Sammons et al., 2009), machine translation (Wu and Fung, 2009; Liu and Gildea, 2010; Wu et al., 2011; Gao and Vogel, 2011), and dialogue systems (Basili et al., 2009; van der Plas et al., 2009), among others.

Most of the current statistical approaches to SRL are supervised, requiring large quantities of human annotated data to estimate model parameters. However, such resources are expensive to create and only available for a small number of languages and domains. Moreover, when moved to a new domain, performance of these models tends to degrade substantially (Pradhan et al., 2008). Scarcity of annotated data has motivated the research into techniques capable of exploiting unlabeled data, that is, semi-supervised and unsupervised learning.

The existing semi-supervised approaches to SRL can largely be regarded as extensions to supervised techniques, as they use supervised learning as sub-routines in the estimation process. These include self-training and co-training methods (He and Gildea, 2006b; Lee et al., 2007; Kaljahi and Samad, 2010), mono-lingual and cross-lingual annotation projection (Fürstenau and Lapata, 2009; Pado and Lapata, 2009; van der Plas et al., 2011), and methods which exploit or induce word representations to reduce the sparsity of lexicalized features (He and Gildea, 2006a; Deschacht and Moens, 2009; Collobert et al., 2011). Most of these approaches, especially the bootstrapping-style methods (He and Gildea, 2006b; Lee et al., 2007; Kaljahi and Samad, 2010; Fürstenau and Lapata, 2009), have achieved minimal or even no improvement from using unlabeled data. Consequently, the development of effective semi-supervised techniques remains an important and largely unresolved problem.

Another vein of research exploiting unlabeled data for shallow semantic parsing has focused on purely unsupervised set-ups (Swier and Stevenson, 2004; Grenager and Manning, 2006; Lang and Lapata, 2010, 2011a,b; Titov and Klementiev, 2012; Garg and Henderson, 2012; Fürstenau and Rambow, 2012). The unsupervised setting is important in itself, and the development of these methods arguably provides interesting insights into modeling implicit supervision signals present in unlabeled data. However, given that small amounts of labeled data are often easy to obtain, it is surprising that no previous work that we are aware of looked into integration of labeled data into unsupervised SRL systems.[1] Moreover, due to the inherent difference in the clustering metrics used for unsupervised SRL and the labeled accuracy scores used to evaluate supervised SRL methods, they have so far never been properly compared. These are the gaps addressed by this paper.

In this work, we show how a state-of-the-art unsupervised Bayesian model (BayesSRL) (Titov

[1] This semi-supervised learning setting is sometimes referred to as semi-*un*supervised (Daumé III, 2009).

and Klementiev, 2012) can be used in a semi-supervised set-up. BayesSRL is especially appropriate for our study as it automatically induces a common representation encoding properties of the syntax-semantics interface that are valid across predicates, contrasting much of other research on unsupervised SRL where separate models were induced for each predicate (Grenager and Manning, 2006; Lang and Lapata, 2010, 2011a,b; Garg and Henderson, 2012; Fürstenau and Rambow, 2012). These models would not be able exploit sparse labeled data effectively, as they would essentially split this scarce data into even smaller (and often empty) training sets.

A straightforward way of integrating labeled data into learning of a generative model would amount to maximizing joint probability of labeled and unlabeled data. However, due to hard constraints in the BayesSRL model and the great disbalance between the amount of labeled and unlabeled data, we argue that a different approach is preferred. Namely, we use labeled data to construct an informed prior over the potential semantic representations and also modify the model to integrate the labels as soft constraints on admissible semantic structures.

We compare the semi-supervised approach we propose to a state-of-the-art supervised method (Johansson and Nugues, 2008a). Though the BayesSRL model exploits a cross-predicate representation, it does not align roles across predicates which prevents us from using supervised evaluation metrics. Consequently, we evaluate the methods using clustering measures: the harmonic mean of purity and collocation, a common metric for unsupervised SRL evaluation (Lang and Lapata, 2010), and the information-theoretic V-Measure (Rosenberg and Hirschberg, 2007).

The semi-supervised method outperforms its supervised counterpart when the amount of labeled data is small. Unsurprisingly, it does not fare as well when the amount of data increases. We believe that this is primarily due to the overly coarse modeling of the syntax-semantics interface, as it is optimized for the unsupervised setting. Nevertheless, these results strongly suggest that approaching the semi-supervised learning setting for SRL from an unsupervised perspective is a promising research direction and that the existing unsupervised SRL methods are already mature enough to be useful for low resource languages with little or no labeled data available.

2 Background

In this section, we begin by formally defining the semantic role labeling task, and then discuss the distance-dependent Chinese Restaurant process (Blei and Frazier, 2011), used as a component in the BayesSRL model and crucial for effective learning in the semi-supervised setting. We conclude the section with a short description of the BayesSRL model.

2.1 Task Definition

The SRL task involves prediction of predicate argument structure, i.e. both identification of arguments as well as assignment of labels according to their underlying semantic role. For example, in the following sentences:

(a) [$_{A0}$ Mary] opened [$_{A1}$ the door].
(b) [$_{A1}$ The door] opened.
(c) [$_{A1}$ The door] was opened [$_{A0}$ by Mary].

Mary always takes an agent role (*A*0 in the PropBank notation (Palmer et al., 2005)) for the predicate *open*, and *door* is always a patient (*A*1).

In this work we focus on the labeling stage of semantic role labeling. Identification, though an important problem, can be tackled with heuristics (Lang and Lapata, 2011a; Grenager and Manning, 2006; de Marneffe et al., 2006), with unsupervised techniques (Abend et al., 2009) or potentially by using a supervised classifier trained on a small amount of data. In our experiments we use the heuristic identifier of Lang and Lapata (2011a). Also, as in much of the previous work on supervised and unsupervised SRL, we rely on automatically generated syntactic dependency trees.

In the labeling stage, semantic roles are represented by clusters of arguments, and labeling a particular argument corresponds to deciding on its role cluster. However, instead of dealing with argument occurrences directly, in BayesSRL they are represented as predicate-specific syntactic signatures, called *argument keys*. The following syntactic features are used to form the argument key representation:

- Active or passive verb voice (`ACT/PASS`).
- Argument position relative to predicate (`LEFT/RIGHT`).
- Syntactic relation to its governor.
- Preposition used for argument realization.

In the above example, the argument keys for candidate arguments *Mary* for sentences (a) and (c) would be `ACT:LEFT:SBJ` and `PASS:RIGHT:LGS->by`,[2] respectively. While aiming to increase the purity of argument key clusters, this particular representation will not always produce a good match: e.g. *door* in sentence (b) will have the same key as *Mary* in sentence (a). Consequently, this introduces an upper bound on the model performance: in our experimental set-up the upper bound on the purity of clustering was equal to 91.7%.

Increasing the expressiveness of the argument key representation by using features of the syntactic frame would enable us to distinguish that pair of arguments. However, we keep this representation, in part to compare with previous work and in part because we are primarily interested in set-ups with little annotated data where this upper bound would not be as limiting.

The clustering implicitly defines the set of permissible *alternations*, or changes in the syntactic realization of the argument structure of the verb. For example, passivization can be roughly represented with the clustering of the key `ACT:LEFT:SBJ` with `PASS:RIGHT:LGS->by` and `ACT:RIGHT:OBJ` with `PASS:LEFT:SBJ`.

In sum, BayesSRL treats the unsupervised semantic role labeling task as clustering of argument keys. Thus, argument occurrences in the corpus whose keys are clustered together are assigned to the same semantic role. The objective of this work is to study how argument key clusterings can be improved by using small amounts of annotated data.

2.2 Distance-dependent CRPs

The Chinese Restaurant Process (CRP), a standard component in non-parametric Bayesian modeling, defines a probability distributions over partitions of a set of objects. It encodes general rich-get-richer dynamics and, as such, is often useful in modeling long tail distributions. CRPs do not distinguish between individual objects and, consequently, prior probability that two objects would end up in the same subset is constant for any choice of objects. Distant-dependent

[2] `LGS` denotes a logical subject in a passive construction (Surdeanu et al., 2008).

CRPs (dd-CRPs) (Blei and Frazier, 2011) use a similarity function d_{ij} in generating partitions: they prefer to place pairs (i, j) with larger similarity d_{ij} in a single subset. More formally, each object i chooses itself a partner c_i with the probability

$$p(c_i = j|D, \alpha) \propto \begin{cases} d_{ij}, & i \neq j \\ \alpha, & i = j \end{cases} \quad (1)$$

where α is a non-negative concentration parameter. The resulting partition is defined by connected components in the directed graph encoded by the partnership relation c. Unlike normal CRP, dd-CRP lacks the exchangeability property and the probability of a given partition cannot be efficiently computed. Nevertheless, efficient inference is possible with MCMC techniques or approximate MAP search methods.

The prior is invariant under joint rescaling of the concentration parameter and the similarity scores, and the proportion of the concentration parameter to the distance parameters can be regarded as a parameter controlling granularity of clustering. We use a slight extension to the original dd-CRP by allowing the concentration parameter to be different for every example and, therefore, write it as $\alpha_i = d_{ii}$.

The similarities D can be fixed and used to encode prior knowledge about the problem (Blei and Frazier, 2011; Socher et al., 2011; Duan et al., 2007; Jensen and Shore, 2011) or can be induced automatically by sharing them across several instances of the clustering problem in a multi-task setting (Titov and Klementiev, 2012). In this work, as discussed in Section 3.2, we use the dd-CRP priors to fill both of these roles.

2.3 BayesSRL Model

In this section we describe the Bayesian model which we use as a basis for our semi-supervised learning approach. For more detailed and formal description of the model we refer the reader to Titov and Klementiev (2012). In this work we use the *coupled* version of the BayesSRL model, that is the model which induces cross-predicate representations.

In Section 2.1 we defined our task as clustering of argument keys, where each cluster corresponds to a semantic role. If an argument key k is assigned to a role r ($k \in r$), all of its occurrences are labeled r.

The Bayesian model encodes two common assumptions about semantic roles. First, it enforces the selectional restriction assumption: namely it stipulates that the distribution over potential argument fillers is sparse for every role, implying that 'peaky' distributions of arguments for each role r are preferred to flat distributions. Second, each role normally appears at most once per predicate occurrence. The inference algorithm will search for a clustering which meets the above requirements to the maximal extent.

As we argued in Section 2.1, clusterings of argument keys implicitly encode the pattern of alternations for a predicate. The set of permissible alternations is predicate-specific,[3] but still most of the alternation are shared across several or many predicates (e.g., passivization or dativization). Consequently, BayesSRL regards semantic role induction as a multi-task clustering problem and encodes the relative 'popularity' of alternations by quantifying how likely a pair of keys is to be clustered. These scores (d_{ij} for every pair of argument keys i and j) are induced automatically within the model, and treated as latent variables shared across predicates.

[3] Or, at least specific to a class of predicates (Levin, 1993).

Parameters:

$D \sim NonInform$ [similarity graph]
for each predicate $p = 1, 2, \ldots$:
  $B_p \sim dd\text{-}CRP(\alpha, D)$ [partition of arg keys]
  for each role $r \in B_p$:
    $\theta_{p,r} \sim DP(\beta, H^{(A)})$ [distrib of arg fillers] $\psi_{p,r} \sim Beta(\eta_0, \eta_1)$ [geom distr for dup roles]

Data generation:

for each predicate $p = 1, 2, \ldots$:
  for each occurrence s of p:
    for every role $r \in B_p$:
      if $[n \sim Unif(0, 1)] = 1$: [role appears at least once]
        GenArgument(p, r) [draw one arg]
        while $[n \sim \psi_{p,r}] = 1$: [continue generation]
          GenArgument(p, r) [draw more args]

GenArgument(p, r):
  $k_{p,r} \sim Unif(1, \ldots, |r|)$ [draw arg key]
  $x_{p,r} \sim \theta_{p,r}$ [draw arg filler]

Figure 1: The BayesSRL model.

The model associates two distributions with each predicate: one governs the selection of argument fillers for each semantic role, and the other models (and penalizes) duplicate occurrence of roles. Each predicate occurrence is generated independently given these distributions. Let us describe the model by first defining how the set of model parameters and an argument key clustering are drawn, and then explaining the generation of individual predicate and argument instances. The generative story is formally presented in Figure 1.

The generation starts by choosing a graph D with non-negative weights $d_{i,j}$ on edges from a non-informative prior, in other words, uniformly over the space of such graphs. Then for each predicate p, a partition of argument keys B_p is drawn from a distance-dependent Chinese Restaurant Process dd-CRP(α, D), with each subset $r \in B_p$ representing a single semantic role.

Next, the parameters are generated from the corresponding prior distributions. For details, we refer the reader to Titov and Klementiev (2012).

Now, when parameters and argument key clusterings are chosen, we can summarize the remainder of the generative story as follows. We begin by independently drawing occurrences for each predicate. For each predicate role we independently decide on the number of role occurrences. Then each of the arguments is generated (see **GenArgument**) by choosing an argument key $k_{p,r}$ uniformly from the set of argument keys assigned to the cluster r, and finally choosing its filler $x_{p,r}$, where the filler is the lemma of the syntactic head of the argument.

In sum, the properties of the BayesSRL model most relevant to the discussion of the semi-supervised extension are (1) induction of predicate-specific hard clustering of argument keys and (2) learning of a cross-predicate similarity measure D over pairs of argument keys.

GenArgument(p, r):
  $b \sim Bernoulli(\epsilon)$
  if $b = 1$:
    $k_{p,r} \sim H^{(K)}$ [noisy arg key]
  else
    $k_{p,r} \sim Unif(1, \ldots, |r|)$ [true arg key]
    $x_{p,r} \sim \theta_{p,r}$ [draw arg filler]

Figure 2: A modified model of argument generation.

3 Semi-Supervised Extension

In this section, we discuss two ways that the labeled data can be exploited in estimating the BayesSRL model. In practice, we found that their combination yields the best result.

3.1 Adding Labels

The integration of labeled data in a generative model is usually trivial and amounts to maximizing the joint likelihood of the observable data. In practice, it implies that the observable labels will be clamped in the estimation process. The straightforward application of this idea to our set-up is problematic. The BayesSRL method makes hard decisions about the clustering of argument keys, and, given the imperfect purity of argument keys and potential annotation errors, no single clustering would be entirely compatible with the labeled data, resulting in zero probability for any model state. Intuitively, one would want to relax this compatibility assumption by allowing for some inconsistency between induced clusterings and labeled data, while still favoring more compatible configurations.

A standard trick to achieve this behavior within the generative framework is to assume that with some small probability ϵ the true outcome is substituted with a random pick. The parameter ϵ would serve as a penalty for inconsistency, the smaller the probability ϵ, the more severe is the penalty. In our case, it translates into modifying the **GenArgument**(p, r) by introducing the possibility of drawing the random argument key from some base distribution $H^{(K)}$, instead of choosing it from the set of keys associated with r (See Figure 2). We use the normalized counts of argument keys in the corpus as the base distribution $H^{(K)}$.

Labeled data integrated in this relaxed BayesSRL model would affect the induced shared prior D and, consequently, the information present in the labeled data would be propagated across different predicates. Unfortunately, there are two problems with using this approach which negatively affect practical results.

The first deficiency is connected with the fact that in practice the amount of unlabeled data vastly exceeds the amount of labeled data nullifying the effect of the latter during estimation. A standard heuristic approach to mitigate this deficiency is to reweigh the data to put an extra emphasis on the labeled part. This technique is unlikely to be very effective here as the argument key clusterings are drawn from dd-CRP(α, D) once for each predicate, not once per predicate occurrence, and the proportion of predicates in labeled and unlabeled data would remain unaffected by instance reweighting.

Another problem is more subtle. As discussed in Titov and Klementiev (2012), their method induces the pairwise clustering preferences D but does not attempt to learn the concentration

parameters α_k and also enforces a form of normalization on the pairwise similarity, effectively 'freezing' the granularity of clusterings. This is fairly natural for an unsupervised setting where the model designer should have some form of control over the granularity but not as desirable in the semi-supervised setting where the granularity should be learned from the annotated data. In fact, as we will see in Section 4, labeled data mostly provides evidence for combining clusters (thus increasing collocation), and, consequently, the ability to learn granularity is crucial.

Thus, a compromise is necessary between (a) learning the granularity from labeled data and (b) limiting the influence of unlabeled data on cluster granularity. We implement this idea by using annotated examples to construct an informed prior.

3.2 Constructing Informed Priors

An alternative approach to directly incorporating the labeled data in the objective function would be to use the data to define an informed 'prior' over argument key clusterings. To this end, we estimate from the labeled data how likely argument keys k and k' are to belong to the same role and how likely a specific key k is to be left unclustered. We use the former to set the similarity $\hat{d}_{kk'}$, and the latter to set the concentration parameter α_k for the dd-CRP model. More precisely, we estimate both the predicate-specific similarities $\hat{d}^{(p)}$ and the cross-predicate similarities $\hat{d}$. When generating partitions B_p (see Figure 1), we multiply $\hat{d}^{(p)}$, $\hat{d}$, and the automatically induced prior d and use the resulting combined similarity in the dd-CRP process. The concentration parameters are combined in the same way as similarities. This techniques corresponds to the standard product-of-expert combination approach (Hinton, 2002). The remaining part of the section describes this idea more formally.

Initially we will consider individual predicates and then we will generalize the approach to cross-predicate similarities. Consider a predicate p, and assume that we have K different argument keys and R different roles,[4] and that each argument key k appears N_k times in the labeled data, and is annotated $N_{k,r}$ times with role r. In order to estimate the required probabilities we need to make assumptions about the joint generation of labels and argument keys.

We assume that there exists a fixed latent mapping g from argument keys to semantic roles and any such mapping is a-priori equiprobable, $P(g) = const$. However, when generating a label $g(k)$ for a key k, we assume that it can be replaced by any of the remaining $R-1$ roles with small probability γ. The probability of the set of labeled examples X_k associated with the key k given a mapping g can be written as

$$P(X_k|g(k)=r) = (1-\gamma)^{N_{k,r}} \left(\frac{\gamma}{R-1}\right)^{N_k - N_{k,r}}.$$

The joint probability of the sets of labeled examples X_k and $X_{k'}$ under the assumptions that either (1) the two keys belong to the same (any) role or (2) belong to two different roles can

[4] In our experiments we set R to 21, the number of distinct roles in PropBank, and K to the number of argument keys appearing both in labeled and unlabeled data for the considered predicate.

be computed by summing over the roles:[5]

$$
\begin{aligned}
&P(X_k, X_{k'} | g(k) = g(k')) \\
&\quad = \sum_r P(X_k | g(k) = r) P(X_{k'} | g(k') = r) \\
&P(X_k, X_{k'} | g(k) \neq g(k')) \\
&\quad = \sum_r P(X_k | g(k) = r) \sum_{r' \neq r} P(X_{k'} | g(k') = r')
\end{aligned}
$$

The posterior probability that two keys belong to the same role $P(g(k) = g(k')|X)$, where X is the entire labeled dataset, is given by renormalizing the two likelihoods above. As the distance $d^{(p)}_{kk'}$ in dd-CRP essentially encodes how much more likely the two keys are clustered together than by random chance, we compute the similarity as

$$
\hat{d}^{(p)}_{kk'} = \frac{P(g(k) = g(k')|X)}{P(g(k) = g(k'))}, \tag{2}
$$

where $P(g(k) = g(k'))$ is the prior probability that two keys are labeled with the same role, equal to $1/R$.

A very similar algebra is used to derive the probability that an argument key k is the only key assigned to some role $P(\nexists k', k \neq k' : g(k) = g(k')|X)$. The concentration parameter $\hat{\alpha}_k$ is set to

$$
\hat{\alpha}^{(p)}_k = \frac{P(\nexists k', k \neq k' : g(k) = g(k')|X)}{P(\nexists k', k \neq k' : g(k) = g(k'))}, \tag{3}
$$

where the denominator is the prior probability of not sharing the role with any other argument key, $(R-1)^{k-1}/R^{k-1}$. Note that if no labeled data is available for the considered predicate p, equations (2) and (3) would yield 1 and, as desired, the prior would not affect prediction of other experts in the product-of-expert combination.

The above approach induces predicate specific priors but this is insufficient for all but very frequent predicates. Consequently, we use a similar approach to define cross-predicate similarities $\hat{d}$ but with a larger γ', thus penalizing less severely for violations. For the cross-predicate similarities, the assumption is that (independently over pairs of keys) each pair of keys either shares a role in all the predicates or the two keys are labeled with a different role in all the predicates. This implies that the similarities can be computed by multiplying the results of computations (2) over all the predicates, while using the parameter γ' instead of the original γ. The same multiplication is done for the concentration parameter.

Note that in this approach we never attempted to encode cross-predicate correspondence between labeled semantic roles, the prior (and the model as whole) is invariant under any renaming of roles for individual predicates.

Admittedly, this method is not a proper estimation method for the BayesSRL model but rather the use of an extrinsic probabilistic model to set the similarity scores in the dd-CRP prior. This is in line with much of the work on using dd-CRPs where the similarities were used to encode prior or external knowledge (Blei and Frazier, 2011; Socher et al., 2011; Duan et al., 2007;

[5] Note that we use here the fact that the mappings are equiprobable.

Jensen and Shore, 2011) rather than estimated as in the multi-tasking set-up of Titov and Klementiev (2012).

To induce the model in the semi-supervised set-up, we use the same approximate MAP search algorithm, as originally proposed in Titov and Klementiev (2012) for the unsupervised setting.

4 Experiments

4.1 Data

Datasets. We evaluate our semi-supervised approach on the CoNLL 2009 distribution (Hajič et al., 2009) of the Penn Treebank WSJ corpus (Marcus et al., 1993). We split the CoNLL training set roughly in half: we draw annotated sentences from the first part (20,000 sentences), and evaluate on the remaining 19,279 sentences. All, but the drawn annotated sentences are used as unsupervised training data as standard for unsupervised SRL.

Syntactic annotation. We annotate the data with dependency structures predicted by the syntactic component of the LTH system (Johansson and Nugues, 2008b), a more realistic setup than making use of the gold syntactic annotation.

Predicate and argument identification. We select all non-auxiliary verbs as predicates.[6] We identify their arguments using a heuristic proposed in (Lang and Lapata, 2011a). Since our goal is to evaluate the argument labeling stage of semantic role labeling, we use this argument identification procedure for all of the systems in our experiments. The quality of argument identification on CoNLL 2009 using predicted syntactic analyses was F1 82.7% (P 83.3% / R 82.0%).

4.2 Evaluation Metrics

We cannot use supervised metrics to evaluate our models, since we do not have an alignment between gold labels and clusters induced in the unsupervised and semi-supervised set-up.[7] Instead, we use the following two standard sets of clustering metrics for our evaluation:

Purity, Collocation, and F1. We use the standard purity (PU) and collocation (CO) metrics as well as their harmonic mean (F1) to measure the quality of the resulting clusters. Purity measures the degree to which each cluster contains arguments sharing the same gold role and collocation evaluates the degree to which arguments with the same gold roles are assigned to a single cluster, see (Lang and Lapata, 2010).

Homogeneity, Completeness, and V-Measure. Additionally, we also evaluate with the information-theoretic V-Measure (V) (Rosenberg and Hirschberg, 2007). It is defined as the harmonic mean of homogeneity (H) and completeness (C) scores, which attempt to measure similar characteristics of the induced clustering as purity and collocation, respectively.

We compute the aggregate scores for all metrics over all predicates in the same way as Lang and Lapata (2011a) by weighting the scores of each predicate by the number of its argument occurrences. Since our goal is to evaluate the clustering algorithms, we *do not* include incorrectly identified arguments when computing these metrics.

[6] In this work we do not disambiguate predicate senses.

[7] Our BayesSRL extension does not propagate role labels between predicates which we would need to compute supervised metrics.

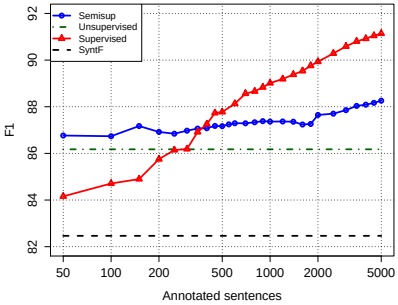


Figure 3: Performance (F1) of supervised (*Supervised*) and semi-supervised (*SemiSup*) systems vs. the number of annotated sentences, along with the original unsupervised model (*Unsupervised*) and the syntactic baseline (*SyntF*).

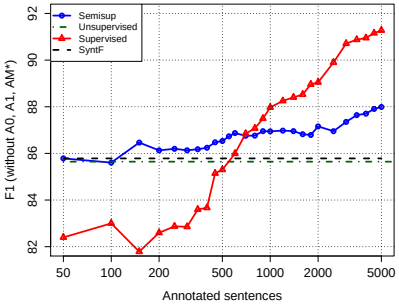


Figure 4: Performance (F1) evaluated on all roles except *A*0, *A*1, and *AM*∗ (modifier arguments) vs. the number of annotated sentences.

4.3 Model Parameters

The unsupervised model and the semi-supervised extension are robust to parameter settings. While they could be tuned by visual inspection of the induced argument roles, as in much of the previous work, we instead tuned them on the standard CoNLL held-out set primarily for replicability reasons.

4.4 Systems

In our experiments, we compare the performance of three systems: our semi-supervised extension (*SemiSup*) to the original state-of-the-art unsupervised model (*Unsupervised*) of Titov and Klementiev (2012), as well as the best CoNLL-08 shared task supervised SRL system (*Supervised*) of Johansson and Nugues (2008b). We also compare against the syntactic function baseline (*SyntF*), which is considered difficult to outperform in the unsupervised setting (Grenager and Manning, 2006; Lang and Lapata, 2010). It simply clusters predicate arguments according to the dependency relation to their head. As in previous work, we allocate a cluster for each of 20 most frequent relations in the CoNLL dataset and one cluster for all other relations.

4.5 Discussion

Figure 3 summarizes the results for the three systems and the syntactic baseline. The semi-supervised model outperforms the supervised counterpart when up to about 350 annotated sentences are available for training. It also continues to improve over the original unsupervised model as more annotated sentences are used for training. Table 1 details the single point of 300 labeled sentences on Figure 3 and breaks up the evaluation of the three systems and the syntactic baseline. It also shows the effect of the two ways of exploiting labeled data we

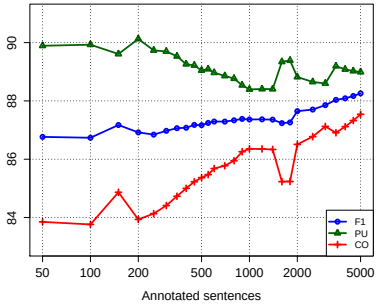


Figure 5: Purity, Collocation, and F1 of our semi-supervised extension (*SemiSup*) vs. the number of annotated sentences.

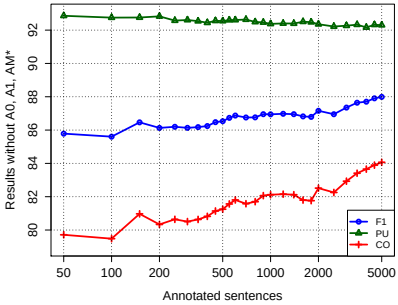


Figure 6: Purity, Collocation, and F1 of our semi-supervised extension (*SemiSup*) evaluated on all roles except *A*0, *A*1, and *AM*∗ vs. the number of annotated sentences.

	PU	CO	**F1**	H	C	**V**
Supervised	88.0	84.5	**86.2**	79.6	74.7	**77.0**
Unsupervised	89.6	83.0	**86.2**	83.8	73.3	**78.2**
SemiSup	89.7	84.4	**87.0**	83.6	74.9	**79.0**
SemiSup-l	89.5	84.2	**86.8**	83.3	74.6	**78.7**
SemiSup-p	90.0	82.5	**86.1**	84.4	72.8	**78.2**
SyntF	83.3	81.6	**82.5**	73.4	70.4	**71.9**

Table 1: Purity (PU), Collocation (CO), and F1, as well as Homogeneity (H), Completeness (C), and V-Measure (V) for for a single point (300 labeled sentences) on Figure 3. Results are for the syntactic baseline (*SyntF*), the supervised system (*Supervised*), the unsupervised model (*Unsupervised*), our semi-supervised extension (*SemiSup*), as well as our extension without adding labeled data to the generative story (*SemiSup-l*), and without the informed prior (*SemiSup-p*).

proposed in Section 3. *SemiSup-l* and *SemiSup-p* is our semi-supervised (*SemiSup*) method without adding labeled data to the generative story, and without informed priors, respectively. Note, that while adding labeled data alone does not improve over the performance of the unsupervised model for this number of labeled examples, the combination of the two methods yields a substantial improvement both in terms of F1 and V-Measure.

*A*0 and *A*1 arguments are annotated in PropBank based on the proto-role theory presented in (Dowty, 1991) and correspond to proto-agents and proto-patients, respectively, while arguments receiving an *AM*∗ label are supposed to be adjuncts, and the roles they express are consistent across all verbs. In order to evaluate the model performance on arguments which do not necessarily express consistent semantic roles across verbs, we next exclude *A*0, *A*1, and *AM*∗ from evaluation (Figure 4). The semi-supervised extension again substantially outperforms the supervised model when fewer than about 700 annotated examples are available.

Finally, Purity / Collocation breakdown for our semi-supervised extension (*SemiSup*) evaluated

an all roles and all roles except *A*0, *A*1, and *AM*∗ is shown on Figure 5 and Figure 6, respectively. Labeled data mostly provides evidence for combining clusters, so more labeled data implies collocation improvements albeit with some drop in purity.

Our semi-supervised method outperforms the state-of-the-art supervised model when the number of labeled sentences is relatively small, but falls behind when the amount of annotated data grows. This is likely due to the simplistic and overly coarse representation and modeling of the linking between syntax and semantics which places an upper bound on how well the original unsupervised model and the semi-supervised extension can do. However, our results strongly suggest that approaching semi-supervised SRL by exploiting labeled data in unsupervised methods is a promising research direction. Existing state-of-the-art methods can already be used for languages and domains for which little or no annotated data is available.

5 Additional Related Work

Additionally to the semi-supervised approaches to SRL discussed in the introduction, semi-supervised and weakly-supervised techniques have also been explored for other types of semantic representations but these studies have mostly focused on restricted domains (Kate and Mooney, 2007; Liang et al., 2009; Titov and Kozhevnikov, 2010; Goldwasser et al., 2011; Liang et al., 2011). Similarly, unsupervised induction for other shallow semantic formalisms include Poon and Domingos (2009, 2010) and Titov and Klementiev (2011).

A related problem of inducing script knowledge, or narrative event chains, has recently received a considerable attention (Chambers and Jurafsky, 2008; Manshadi et al., 2008; Chambers and Jurafsky, 2009; Regneri et al., 2010, 2011) with approaches mostly considering unsupervised or weakly-supervised setting due to scarcity of labeled data. Though in this paper we focus on the labeling of arguments the complementary task of unsupervised argument identification was considered in Abend et al. (2009).

Unsupervised learning has been one of the central paradigms for the closely-related area of relation extraction, where several techniques have been proposed to cluster semantically similar verbalizations of relations (Lin and Pantel, 2001; Banko et al., 2007). Similarly to SRL, semi-supervised approaches in this area are also typically based on bootstrapping techniques (e.g., (Agichtein and Gravano, 2000; Rosenfeld and Feldman, 2007)) and often achieve impressive results. However, their set-up is arguably different from ours as relation extractors are generally more precision-oriented, focus primarily on binary relations and can partially sidestep the complexity of language.

6 Conclusions

In this work, we demonstrated that unsupervised techniques can be improved by exploiting small amounts of labeled data yielding SRL parsers competitive with supervised approaches in a low resource setting. We also uncovered some of the deficiencies of the existing unsupervised approaches; namely, overly coarse modeling of syntax-semantics interface resulting in a lower asymptotic performance in semi-supervised settings. These results motivate further research into design of generative models appropriate for semi-supervised learning of shallow semantics.

Acknowledgements

The work was supported by the MMCI Cluster of Excellence and a Google research award. The authors thank Mikhail Kozhevnikov, Alexis Palmer and the anonymous reviewers for their suggestions.

References

Abend, O., Reichart, R., and Rappoport, A. (2009). Unsupervised argument identification for semantic role labeling. In *ACL-IJCNLP*.

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

Baker, C. F., Fillmore, C. J., and Lowe, J. B. (1998). The Berkeley FrameNet project. In *Proceedings of the Thirty-Sixth Annual Meeting of the Association for Computational Linguistics and Seventeenth International Conference on Computational Linguistics (ACL-COLING'98)*, pages 86–90, Montreal, Canada.

Banko, M., Cafarella, M. J., Soderland, S., Broadhead, M., and Etzioni, O. (2007). Open information extraction from the web. In *IJCAI*.

Basili, R., Cao, D. D., Croce, D., Coppola, B., and Moschitti, A. (2009). Cross-language frame semantics transfer in bilingual corpora. In *CICLING*.

Blei, D. M. and Frazier, P. (2011). Distance dependent chinese restaurant processes. *Journal of Machine Learning Research*, 12:2461–2488.

Carreras, X. and Màrquez, L. (2005). Introduction to the CoNLL-2005 Shared Task: Semantic Role Labeling. In *CoNLL*.

Chambers, N. and Jurafsky, D. (2008). Unsupervised learning of narrative event chains. In *Proc. of the Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 789–797.

Chambers, N. and Jurafsky, D. (2009). Unsupervised learning of narrative schemas and their participants. In *Proc. of the Annual Meeting of the Association for Computational Linguistics and International Joint Conference on Natural Language Processing (ACL-IJCNLP)*.

Collobert, R., Weston, J., Bottou, L., Karlen, M., Kavukcuoglu, K., and Kuksa, P. (2011). Natural language processing (almost) from scratch. *Journal of Machine Learning Research*, 12:2493–2537.

Daumé III, H. (2009). Semi-supervised or semi-unsupervised? In *NAACL HLT Workshop on Semisupervised Learning for Natural Language Processing*.

de Marneffe, M.-C., MacCartney, B., and Manning, C. D. (2006). Generating typed dependency parses from phrase structure parses. In *LREC 2006*.

Deschacht, K. and Moens, M.-F. (2009). Semi-supervised semantic role labeling using the Latent Words Language Model. In *EMNLP*.

Dowty, D. (1991). Thematic proto-roles and argument selection. *Language*, 67(3):547–619.

Duan, J., Guindani, M., and Gelfand, A. (2007). Generalized spatial dirichlet process models. *Biometrika*, 94:809–825.

Fillmore, C. J. (1968). The case for case. In E., B. and R.T., H., editors, *Universals in Linguistic Theory*, pages 1–88. Holt, Rinehart, and Winston, New York.

Fürstenau, H. and Lapata, M. (2009). Graph alignment for semi-supervised semantic role labeling. In *EMNLP*.

Fürstenau, H. and Rambow, O. (2012). Unsupervised induction of a syntax-semantics lexicon using iterative refinement. In *In Proceedings of the First Joint Conference on Lexical and Computational Semantics (*SEM 2012)*.

Gao, Q. and Vogel, S. (2011). Corpus expansion for statistical machine translation with semantic role label substitution rules. In *ACL:HLT*.

Garg, N. and Henderson, J. (2012). Unsupervised semantic role induction with global role ordering. In *ACL*.

Gildea, D. and Jurafsky, D. (2002). Automatic labelling of semantic roles. *Computational Linguistics*, 28(3):245–288.

Goldwasser, D., Reichart, R., Clarke, J., and Roth, D. (2011). Confidence driven unsupervised semantic parsing. In *ACL*.

Grenager, T. and Manning, C. (2006). Unsupervised discovery of a statistical verb lexicon. In *EMNLP*.

Hajič, J., Ciaramita, M., Johansson, R., Kawahara, D., Martí, M. A., Màrquez, L., Meyers, A., Nivre, J., Padó, S., Štěpánek, J., Straňák, P., Surdeanu, M., Xue, N., and Zhang, Y. (2009). The CoNLL-2009 shared task: Syntactic and semantic dependencies in multiple languages. In *Proceedings of the 13th Conference on Computational Natural Language Learning (CoNLL-2009), June 4-5*.

He, S. and Gildea, D. (2006a). Integrating cluster information for cross-frame semantic role labeling. Technical report, Technical Report 892, University of Rochester.

He, S. and Gildea, D. (2006b). Self-training and co-training for semantic role labeling: Primary report. Technical report, Technical Report 891, University of Rochester.

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

Jensen, S. and Shore, S. (2011). Semiparametric bayesian modeling of income volatility heterogeneity. *Journal of the American Statistical Association*, 106(496):1280–1290.

Johansson, R. and Nugues, P. (2008a). Dependency-based semantic role labeling of PropBank. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 69–78.

Johansson, R. and Nugues, P. (2008b). Dependency-based semantic role labeling of PropBank. In *EMNLP*.

Kaisser, M. and Webber, B. (2007). Question answering based on semantic roles. In *ACL Workshop on Deep Linguistic Processing*.

Kaljahi, Z. and Samad, R. (2010). Adapting self-training for semantic role labeling. In *Proceedings of the ACL 2010 Student Research Workshop*, pages 91–96. Association for Computational Linguistics.

Kate, R. J. and Mooney, R. J. (2007). Learning language semantics from ambigous supervision. In *AAAI*.

Lang, J. and Lapata, M. (2010). Unsupervised induction of semantic roles. In *ACL*.

Lang, J. and Lapata, M. (2011a). Unsupervised semantic role induction via split-merge clustering. In *ACL*.

Lang, J. and Lapata, M. (2011b). Unsupervised semantic role induction with graph partitioning. In *EMNLP*.

Lee, J., Song, Y., and Rim, H. (2007). Investigation of weakly supervised learning for semantic role labeling. In *Advanced Language Processing and Web Information Technology, 2007. ALPIT 2007. Sixth International Conference on*, pages 165–170. IEEE.

Levin, B. (1993). *English Verb Classes and Alternations: A Preliminary Investigation*. University of Chicago Press.

Liang, P., Jordan, M., and Klein, D. (2011). Learning dependency-based compositional semantics. In *ACL: HLT*.

Liang, P., Jordan, M. I., and Klein, D. (2009). Learning semantic correspondences with less supervision. In *ACL-IJCNLP*.

Lin, D. and Pantel, P. (2001). DIRT – discovery of inference rules from text. In *KDD*.

Liu, D. and Gildea, D. (2010). Semantic role features for machine translation. In *Coling*.

Manshadi, M., Swanson, R., and Gordon, A. (2008). Learning a probabilistic model of event sequences from internet weblog stories. In *Proceedings of the 21st FLAIRS Conference*.

Marcus, M. P., Santorini, B., and Marcinkiewicz, M. A. (1993). Building a large annotated corpus of English: The Penn Treebank. *Computational Linguistics*, 19(2):313–330.

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

Palmer, M., Gildea, D., and Kingsbury, P. (2005). The proposition bank: An annotated corpus of semantic roles. *Computational Linguistics*, 31(1):71–106.

Poon, H. and Domingos, P. (2009). Unsupervised semantic parsing. In *EMNLP*.

Poon, H. and Domingos, P. (2010). Unsupervised ontology induction from text. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 296–305. Association for Computational Linguistics.

Pradhan, S., Ward, W., and Martin, J. H. (2008). Towards robust semantic role labeling. *Computational Linguistics*, 34:289–310.

Regneri, M., Koller, A., and Pinkal, M. (2010). Learning script knowledge with web experiments. In *Proceedings of ACL 2010*, Uppsala, Sweden. Association for Computational Linguistics.

Regneri, M., Koller, A., Ruppenhofer, J., and Pinkal, M. (2011). Learning script participants from unlabeled data. In *Proceedings of RANLP 2011*, Hissar, Bulgaria.

Rosenberg, A. and Hirschberg, J. (2007). V-Measure: A conditional entropy-based external cluster evaluation measure. In *Proceedings of the 2007 EMNLP-CoNll Joint Conference*, pages 410–420.

Rosenfeld, B. and Feldman, R. (2007). Using corpus statistics on entities to improve semi-supervised relation extraction from the web. In *ACL*.

Sammons, M., Vydiswaran, V., Vieira, T., Johri, N., Chang, M., Goldwasser, D., Srikumar, V., Kundu, G., Tu, Y., Small, K., Rule, J., Do, Q., and Roth, D. (2009). Relation alignment for textual entailment recognition. In *Text Analysis Conference (TAC)*.

Shen, D. and Lapata, M. (2007). Using semantic roles to improve question answering. In *EMNLP*.

Socher, R., Maas, A., and Manning, C. (2011). Spectral chinese restaurant processes: Non-parametric clustering based on similarities. In *AISTATS*.

Surdeanu, M., Johansson, A. M. R., Màrquez, L., and Nivre, J. (2008). The CoNLL-2008 shared task on joint parsing of syntactic and semantic dependencies. In *CoNLL 2008: Shared Task*.

Swier, R. and Stevenson, S. (2004). Unsupervised semantic role labelling. In *EMNLP*.

Titov, I. and Klementiev, A. (2011). A Bayesian model for unsupervised semantic parsing. In *ACL*.

Titov, I. and Klementiev, A. (2012). A Bayesian approach to semantic role induction. In *Proc. EACL*, Avignon, France.

Titov, I. and Kozhevnikov, M. (2010). Bootstrapping semantic analyzers from non-contradictory texts. In *ACL*.

van der Plas, L., Henderson, J., and Merlo, P. (2009). Domain adaptation with artificial data for semantic parsing of speech. In *Proc. 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics, Companion Volume: Short Papers*, pages 125–128, Boulder, Colorado.

van der Plas, L., Merlo, P., and Henderson, J. (2011). Scaling up automatic cross-lingual semantic role annotation. In *ACL*.

Wu, D., Apidianaki, M., Carpuat, M., and Specia, L., editors (2011). *Proc. of Fifth Workshop on Syntax, Semantics and Structure in Statistical Translation*. ACL.

Wu, D. and Fung, P. (2009). Semantic roles for SMT: A hybrid two-pass model. In *NAACL*.

Hunting for Entailing Pairs in the Penn Discourse Treebank

SARA TONELLI[1] *ELENA CABRIO*[2]
(1) Fondazione Bruno Kessler, Trento, Italy
(2) INRIA, Sophia Antipolis, France
satonelli@fbk.eu, elena.cabrio@inria.fr

ABSTRACT
Given the growing amount of resources developed in the NLP community, it is crucial to exploit as much as possible annotated data and tools across different research domains. Past works on discourse analysis have been conducted in parallel with research on semantic inference and, although the two fields of study are intertwined, there have been only few initiatives to put them into relation. Our work addresses the issue of interoperability by investigating the connection between implicit *Restatement* relations in the Penn Discourse Treebank (PDTB) and Textual Entailment. We compare the performance of two TE systems on the *Restatement* pairs and we argue that TE is a subclass of *Restatement* through a manual validation of the pairs. Furthermore, we observe that entailing pairs extracted from the PDTB add interesting and additional levels of complexity to TE, since inference relation relies less on lexical-syntactic variations, and more on reasoning.

TITLE AND ABSTRACT IN ITALIAN

A caccia di inferenze semantiche nel Penn Discourse Tree Bank

Data l'ingente quantità di risorse sviluppate in trattamento automatico del linguaggio, l'importanza di sfruttare anche in altri campi di ricerca i dati annotati e gli strumenti implementati è diventata fondamentale. In passato, lavori sull'analisi del discorso sono stati condotti parallelamente alla ricerca sulle inferenze semantiche, ma sebbene i due campi di studio presentino numerosi punti in comune, non ci sono state iniziative per avvicinarli. Questo lavoro affronta la questione dell'interoperabilità investigando le connessioni tra la relazione implicita di *Restatement* nel Penn Discourse Treebank (PDTB) e Textual Entailment (TE) (implicazione semantica). Comparando i risultati ottenuti da due sistemi che riconoscono automaticamente la relazione di implicazione, e dall'annotazione manuale di un sottoinsieme di coppie, mostriamo come il TE sia riconducibile ad una sottocategoria di *Restatement*. Inoltre, osserviamo che le coppie in relazione di implicazione estratte dal PDTB mostrano un livello di complessità superiore rispetto a quelle considerate dai sistemi attuali, in quanto la relazione di inferenza si basa meno su variazioni lessico-sintattiche, e più sul ragionamento.

KEYWORDS: Textual entailment, Penn Discourse Treebank, implicit relations.

KEYWORDS IN ITALIAN: Implicazione testuale, Penn Discourse Treebank, relazioni implicite.

Proceedings of COLING 2012: Technical Papers, pages 2653–2668,
COLING 2012, Mumbai, December 2012.

1 Introduction

Given the growing amount of resources and automatic systems developed in the NLP community, it is crucial to guarantee the highest possible compatibility among them, and to exploit as much as possible annotated data and tools across different research domains.[1] Past works on discourse analysis have been conducted in parallel with research on semantic inference (in particular, on textual entailment phenomena, see Sammons et al. (2010), Bentivogli et al. (2010)) and, although the two fields of study seem to be intertwined, no effort has been made into the reuse of annotated data and processing tools across both domains.

With this work, we address the issue of interoperability by investigating the connection between implicit *Restatement* relations in the Penn Discourse Treebank (PDTB) (Prasad et al., 2008) and the Textual Entailment (TE) relation as defined by Dagan et al. (2009). Consider for instance the following sentences extracted from the PDTB and annotated as being in a *Restatement* relation:

(1) *Because hurricanes can change course rapidly, the company sends employees home and shuts down operations in stages.*
The company doesn't wait until the final hours to get ready for hurricanes.

(2) *He's not a reformer – he wants to have the image of a reformer.*
He doesn't want to have the image of the gun man.

Both in Example (1) and (2), a person reading the first sentence would infer that the following is most likely true, which is literally the definition of the textual entailment relation, i.e. a directional relation between a coherent textual fragment (T) and a language expression, which is considered as a hypothesis (H). Entailment holds, i.e. T $\Rightarrow$ H, if the meaning of H can be inferred from the meaning of T, as interpreted by a typical language user (Dagan et al., 2009). In the examples above, *'The company doesn't wait until the final hours[...]'* and *'He doesn't want to have the image of the gun man'* may be both inferred from the previous sentence by a typical language user, without the need of specific background knowledge.

Since in the PDTB more than 3,000 pairs have been labeled as having an implicit *Restatement* relation, it would be important to assess if they can be used also as training instances for TE systems, and if they represent categories of textual entailment pairs that up to now have not been part of the research agenda of the Recognizing Textual Entailment (RTE) evaluation campaigns[2] due to reasons of convenience for the task definition.

For such challenges, the creation of RTE data sets is a costly and time-consuming activity, requiring a lot of manual work for the creation of the T-H pairs and their annotation (about 1000 training and test instances have been usually provided by the organizers of RTE-1 to RTE-5 challenges). Furthermore, textual pairs extracted from the PDTB would represent real data in a discourse context as opposed to RTE pairs, where T is typically an excerpt extracted from a document (generally newspapers) and H is manually created. On the other hand, if *Restatement* and TE are proved to be equal, RTE technologies could be reused to identify and label this type of implicit relations, which are difficult to detect with existing discourse parsers.

[1] This issue has been recently debated during the Collaboratively Constructed Semantic Resources Workshop's panel at ACL2012, where the importance of the development of functional resources strongly connected with NLP systems was underlined, to allow for resources reusability in different tasks.

[2] `http://aclweb.org/aclwiki/index.php?title=Recognizing_Textual_Entailment`

While the importance of discourse information in TE has already been discussed in relation to anaphora and bridging phenomena (in particular starting from RTE-5, where the T was composed by longer paragraphs, requiring coreference resolution (Mirkin et al., 2010b) (Mirkin et al., 2010a)), little attention has been paid to discourse relations holding between sentences, although this is strictly connected to the problem of coreference (for further discussion on this topic, see Section 3).

In this work, we address the following research questions:

- What are the main differences between *Restatement* and TE relations, given that they are very similar from a theoretical point of view?
- Is it possible to use RTE systems to identify implicit *Restatement* relations, since current approaches have proved to have some limitations and achieve poor performance?
- From a TE perspective, is it possible to use entailing pairs extracted from the PDTB to train or evaluate TE systems?

We believe that addressing these issues is important both from a computational and a theoretical point of view: the findings of this study would be beneficial to system developers as well as to computational linguists interested in discourse and inference phenomena.

The paper is structured as follows: in Section 2 past work related to the identification of implicit relations in the Penn Discourse Treebank is presented and the task of Recognizing Textual Entailment is introduced. In Section 3 the PDTB is described, with a focus on implicit and *Restatement* relations. In Section 4 we present the experimental setting, introducing the TE systems we used, and then we detail both the first and the second experiment we carried out. Finally, we draw some conclusions and discuss future work in Section 5.

2 Related work

A number of approaches have been proposed for annotating *explicit* discourse relations following the PDTB paradigm. While the first attempts were limited to retrieving the heads (usually the main verb) of discourse arguments (Elwell and Baldridge, 2008; Wellner and Pustejovsky, 2007), or to extracting only the sentences containing the arguments (Prasad et al., 2010), more recent works have focused on the identification of the exact arguments spans and on the development of end-to-end discourse parsers (Lin et al., 2010; Ghosh et al., 2011b,a). These works rely on the information conveyed by explicit connectives, which proved quite easy to classify using syntactic information (Pitler et al., 2008; Pitler and Nenkova, 2009).

If the connective is not overtly expressed, however, the task is more challenging and requires different features compared to explicit relations. Experiments by Lin et al. (2009), Pitler et al. (2009) and Lin et al. (2010) showed that, despite the promising results and the progress with respect to their baselines, there is still room for improvement.

As introduced before (Section 1), the notion of textual entailment has been proposed as an applied framework to capture major semantic inference needs across applications in NLP (Dagan and Glickman, 2004), (Dagan et al., 2009). Given a pair of textual fragments, it considers if a competent speaker with basic knowledge of the world would typically infer the second from the first one. To promote the development of general TE recognition engines, designed to provide generic modules across applications, since 2005 the Recognizing Textual Entailment

evaluation campaigns[3] have asked participants to develop a system able to detect an inference relation between T-H pairs. In this applied framework, inferences are performed directly over lexical-syntactic representations of the texts. Current systems mainly rely on Machine Learning techniques (typically SVM), logical inference, cross-pair similarity measures between T and H, and word alignment. The definition of TE captures quite broadly the reasoning about language variability needed by different applications for natural language understanding and processing, e.g. information extraction (Romano et al., 2006), text summarization (Barzilay and McKeown, 2005), and reading comprehension systems (Nielsen et al., 2009). Following this rationale, the data sets provided by the challenge organizers are composed of T-H pairs collected from several applicative scenarios (e.g. Question Answering, Information Extraction, Information Retrieval, Summarization), reflecting the way by which the corresponding application could take advantage of automated entailment judgement.[4]

3 *Restatement* relations in the Penn Discourse Treebank

The Penn Discourse Treebank (PDTB) is a resource built on top of the Wall Street Journal (WSJ), in which discourse relations have been manually identified and classified. A *discourse relation* holds between two and only two text spans called *arguments*, that correspond to propositions, events and states.[5]

In the PDTB, relations can be explicitly signaled by a set of lexically defined connectives (e.g. "because", "however", "therefore", etc.). In these cases, the relation is overtly marked, which makes it relatively easy to detect using NLP techniques (Pitler et al., 2008). A relation between two discourse arguments, however, does not necessarily require an explicit connective, because it can be inferred also if a connective expression is missing. These cases are referred to as *implicit relations*, and in the PDTB they are annotated only between adjacent sentences within parahraphs. In case the connective is not overt, PDTB annotators were asked to insert a connective to express the inferred relation.

Examples (3) and (4) represent sentences connected, respectively, by an explicit and an implicit relation. The abstract objects involved in a discourse relation are called `Arg1` and `Arg2` according to syntactic criteria and are reported in italics and in bold respectively.[6]

(3) Explicit: *Use of dispersants was approved* when **a test on the third day showed some positive results.**

(4) Implicit: *The projects already under construction will increase Las Vegas's supply of hotel rooms by 11,795, or nearly 20%, to 75,500*. **By a rule of thumb of 1.5 new jobs for each new hotel room, Clark County will have nearly 18,000 new jobs**.

While in Example (3) the connective "when" explicitly signals a relation holding between `Arg1` and `Arg2`, in (4) no connective was originally expressed. A consequence relation is inferred

[3] `http://aclweb.org/aclwiki/index.php?title=Textual_Entailment_Resource_Pool`

[4] Trying to face more real scenarios, in recent editions of the challenge, i.e. RTE-6 and 7, more complexity was added to the traditional main task, asking TE systems to find all the sentences that entail a given H in a set of documents about a topic.

[5] In order to study entailment phenomena between arguments, we make the assumption that arguments correspond to full clauses or sentences. However, in some cases arguments are just textual fragments shorter than clauses.

[6] This notation convention will be applied to all examples reported in this paper, extracted from the PDTB.

between '*the increase in the number of rooms*' and '*the increase in the number of jobs*', though no *explicit* connective expresses this relation.

Each implicit and explicit relation is assigned a sense label based on a three-layered hierarchy of senses. The top-level, or *class level*, includes four major semantic classes, namely TEMPORAL, CONTINGENCY, COMPARISON and EXPANSION. For each class, a more fine-grained classification has been specified at *type* level. For instance, the relation in Example (1) belongs to the CONTINGENCY class and the *Cause* type. A further *subtype* level has been introduced to specify the semantic contribution of each discourse argument.

In this work, we focus on sentence pairs connected by an *implicit* relation and belonging to the EXPANSION class. In particular, we are interested in the relations in the EXPANSION class marked as *Restatement*, because the way in which such relation is defined shows high similarity with the textual entailment relation.

A *Restatement* relation is annotated between two arguments when the semantics of `Arg2` restates the semantics of `Arg1` and it is inferred that the situations described in `Arg1` and `Arg2` hold true at the same time. *Restatement* relations are further specified into three subtypes, namely "specification", "generalization" and "equivalence". The subtype label depends on the ways in which `Arg2` restates `Arg1`: ||Arg1|| $\rightarrow$ ||Arg2|| in the case of generalization, ||Arg1|| $\leftarrow$ ||Arg2|| in the case of specification, and ||Arg1|| $\leftrightarrow$ ||Arg2|| in the case of equivalence, with $\rightarrow$ indicating logical implication. If more than one subtype interpretation is possible, annotators were allowed to provide a *type* instead of a subtype label, therefore some relations are just classified as *Restatement*.

While intuitively the equivalence relation shares more commonalities with the definition of paraphrase since they both represent bidirectional relations, the specification and generalization types seem to fit well into the definition of textual entailment provided in Section 1 (where the relation of specification has to be considered as a reverse entailment, i.e. the second textual fragment entails the first one).

Let's consider three PDTB sentences annotated as *Restatement*, specifically as implicit specification (5), generalization (6) and equivalence (7):

(5) *She was the child of relative privilege*. **Her mother was a translator; her father was the eternal vice director**.

(6) *Chinese and foreign economists now predict prolonged stagflation: low growth and high inflation.* **The economy is crashing hard**.

(7) *It was like someone had turned a knife in me*. **I was dumbfounded**.

We can represent them in the format of TE pairs, setting as T the first textual fragment and as H the second one (reversing the order for the specification type, as explained before):

(5') T: **Her mother was a translator; her father was the eternal vice director**.
H: *She was the child of relative privilege*.

(6') T: *Chinese and foreign economists now predict prolonged stagflation: low growth and high inflation.*
H: **The economy is crashing hard**.

(7') T: *It was like someone had turned a knife in me*.
H: **I was dumbfounded**.

For all the three pairs, a human reading T would infer that H is most likely true (i.e. they are positive TE pairs).

Leaning on these observations, our intuition is that such pairs automatically extracted from the PDTB could therefore be used to train TE systems, integrating the data sets provided by the RTE challenges organizers (in Section 4 experiments are carried out to prove our intuition). Similarly to RTE pairs, also these sentences are extracted from newspapers. But, while the creation of new T-H pairs requires quite a lot of manual work (for the creation of the H and subsequent annotation), the PDTB is an already available resource. Moreover, in line with the direction of RTE challenges that are now moving toward more real scenarios providing entire documents as T (see RTE-5 to 7), PDTB sentences represent good examples of real data.

Partially guided by reasons of convenience for the task definition, some assumptions have been defined by the organizers of RTE challenges, as for instance the a priori truth of the texts, and the same meaning of entities mentioned in T and H. From a human perspective, the inference required are in general fairly superficial, since generally no long chains of reasoning are involved. Pairs extracted from the PDTB would therefore add interesting and additional levels of complexity to the task, since the relation of inference between T and H relies less on lexical-syntactic variations, and more on reasoning. For instance, Examples (8) and (9) (labeled as *Restatement.equivalence* in the PDTB) can still be considered as positive TE examples, even if they require a lot of background knowledge (e.g. knowledge of idioms and metaphors) for their resolution.

(8) T: **Yet for all his cynicism, he's at heart a closet idealist, a softy with a secret crush on truth, justice and the American Way**.
H: *He's the kind of guy who rescues trampled flags*.

(9) T: *It was like flying without a pilot in the front of the plane*.
H: **It was crazy**.

Even if such level of complexity is still not afforded by current TE systems, the study of these types of arguments could bring new light into textual inference, encouraging the exploration of cases that up to now have not been part of the research agenda.

In the next section we carry out an experimental study *i)* to evaluate the performances of current TE systems on the pairs extracted from the PDTB, and *ii)* to better understand if the theoretical similarity in the definitions of the *Restatement* and the TE relation is actually proved on real data.

4 Experimental setting

In this section, we first introduce the TE systems we used (Section 4.1), and then we present the two sets of experiments we carried out. The first was performed to verify if implicit *Restatement* relations can be detected using current TE systems (Section 4.2). The second was run on a subset of manually re-annotated sentences (Section 4.3), to further verify the relationship between entailment and *Restatement* relations on a controlled data set.

4.1 TE systems description

In order to analyze the correspondence between the *Restatement* and the textual entailment relation, we run different experiments using two off-the-shelf TE systems: VENSES (Delmonte et al., 2009) and EDITS (Kouylekov and Negri, 2010). We choose these two systems because: *i)* they are freely available, *ii)* they obtained similar performances at the last RTE campaigns, and *iii)* they rely on different NLP approaches: VENSES is a rule-based system incorporating and combining different levels of linguistic information, from lexical to semantic knowledge. EDITS, instead, is a supervised TE system implementing a distance-based framework, whose modular architecture combines distance and similarity algorithms.

4.1.1 The VENSES system

VENSES is a rule-based system for recognizing textual entailment based on a linguistic analyzer and an evaluation module.

The first relies on a number of submodules common in Information Extraction systems, i.e. a tokenizer, a multiword and NE recognition module, a PoS tagger based on finite state automata, an in-built syntactic and semantic parser and a pronominal binding module. It also marks and interprets negation, modals and progressive mood.

The evaluation system uses a cost model with rewards/penalties for T-H pairs, where textual entailment is interpreted in terms of semantic similarity: the closest the T-H pairs are in semantic terms, the more probable is their entailment. Rewards in terms of scores are assigned for each 'similar' semantic element.

The system release used in our experiments, which we downloaded at `http://project.cgm.unive.it/venses.html`, is available in two versions: one assigns higher similarity to sentence pairs according to 'shallow' criteria, while the other accounts for 'deep' semantics.

4.1.2 The EDITS system

The EDITS system (Edit Distance Textual Entailment Suite) is an open-source software package for recognizing TE[7] (Kouylekov and Negri, 2010) implementing a distance-based framework which assumes that the probability of an entailment relation between a given T-H pair is inversely proportional to the distance between T and H (i.e. the higher the distance, the lower is the probability of entailment). Within this framework, the system implements different approaches to distance computation, i.e. both edit distance algorithms (that calculate the T-H distance as the cost of insertions, deletions and substitutions that are necessary to transform T into H), and similarity algorithms (e.g. Word Overlap, cosine similarity). Each algorithm returns a normalized distance score between 0 and 1. At a training stage, distance scores calculated over annotated T-H pairs are used to estimate a threshold that best separates positive from negative examples. Such threshold is then used at a test stage to assign a judgment and a confidence score to each test pair.

4.2 Experiment 1: complete data set

Our first experiment is aimed at checking if implicit *Restatement* relations can be detected using existing textual entailment systems. Given that this relation type in the PDTB is defined in

[7] `http://edits.fbk.eu/`

a similar way to textual entailment, we expect TE systems to label sentence pairs having a *Restatement* relation as 'entailing', while sentence pairs connected through another relation type (*Comparison*, *Contingency* or *Temporal*) are likely to be classified as 'not entailing'.

4.2.1 Data set description

We extract all sentence pairs having an implicit *Restatement* relation in the PDTB and we include them in our data set as positive examples. Then, we extract the same number of pairs from the PDTB having an implicit *Comparison*, *Contingency* or *Temporal* relation (the proportion of the three classes reflects their proportion in the PDTB). These pairs are included in the data set as negative examples. The other pairs labeled as *Expansion* but not belonging to the *Restatement* subtype have not been considered in the experiment.

In order to create sentence pairs resembling Text-Hypothesis pairs from RTE challenges, it was necessary in some cases to change the order of the arguments in the positive examples:

- Sentence pairs connected through a *Generalization* label were kept in their original format. Since ||Arg1|| → ||Arg2||, then the sentence corresponding to `Arg1` was considered the Text and `Arg2` the Hypothesis.
- Sentence pairs connected through a *Specification* label were reversed, with `Arg2` being the Text and `Arg1` the Hypothesis. For instance, given the sentences reported in (10) and connected through an implicit *Specification* relation, we build the T-H pair reported in (11).

  (10) *This is an old story.* (`Arg1`). **We're talking about years ago before anyone heard of asbestos having any questionable properties.** (`Arg2`).

  (11) T: We're talking about years ago before anyone heard of asbestos having any questionable properties.
  H: This is an old story.

- In case of sentence pairs connected through an *Equivalence* relation, the longer sentence in the pair was considered as the Text and the shorter one the Hypothesis. This was done in order to resemble as much as possible the T-H pairs in RTE data sets, where the hypothesis is usually shorter than the text. According to the definition of equivalence as ||Arg1|| ⟷ ||Arg2||, however, the entailment relation should hold in both directions.
- For sentence pairs connected through a generic *Restatement* label, we applied the same rule as for the *Equivalence* cases.

Our data set was built to include as positive examples all implicit *Restatement* relations (of any subtype) extracted from the PDTB. In case a relation was annotated with multiple labels, we selected it only if the first option was *Restatement*, otherwise the sentence pair was not included in the data set, neither as positive nor as negative example. Then, the same number of negative examples was collected, having the same proportion of implicit COMPARISON, CONTINGENCY and TEMPORAL relations as in the PDTB. In the end, the data set comprises *6,244 sentence pairs*, equally divided into positive and negative examples.

Since EDITS needs a training set to learn the threshold that best separates positive from negative pairs, the data set was split into a training and a test set, each being 50% of the complete

data set with an equal distribution of the sense labels. VENSES does not require supervision, therefore only the test set was used with this system.

4.2.2 Results and discussion

We run VENSES (both deep and shallow versions) and EDITS on the test set. The training set was used by EDITS to learn the threshold, as explained in Section 4.1.2. We apply two basic configurations of EDITS, i.e. Word Overlap and Cosine similarity algorithms on lemmatized texts (stopwords removed).

We compute a simple baseline based on word overlap between the two arguments: we first calculate for each training pair a similarity score using the Text::Similarity::Overlaps library[8] (we use the F1 value, which is a weighted average between the percentage of overlapping words in the text and the percentage of overlapping words in the hypothesis). Then, we train a simple NaiveBayes classifier using only this value as feature (an SVM classifier was trained as well but achieved poorer performance). The model was further used to classify the pairs in the test set.

We report the results in Table 1.

	Baseline	**VENSES** (d.)	**VENSES** (s.)	**EDITS** (wo)	**EDITS** (cos.)
Positive Pairs	*67.53*	33.33	36.14	55.48	51.48
Negative Pairs	*32.07*	62.46	60.16	48.5	49.94
Overall	*49.88*	**47.89**	**48.15**	**49.33**	**50.06**

Table 1: Systems performances on test set (% correctly classified pairs)

VENSES shows a different performance on negative and positive examples both with the shallow and the deep settings. The system has a conservative approach towards entailment, in that it underestimates the positive examples and overestimates the negative ones. Therefore, its performance is better on non-entailing pairs. EDITS strategy, instead, seems to be better balanced. Nevertheless, the systems do not significantly outperform the baseline, which tends to label as 'entailing' also negative pairs.

Given that the performances we obtained on the PDTB data set are below the average system performances in standard RTE tasks (the accuracy of most of the systems ranges between 55% to 65% for the two-way judgement task), these results may depend on two reasons: *i)* either the entailment phenomena underlying the positive examples are more difficult to detect than those in RTE data sets, due to the fact that PDTB pairs are extracted from a real corpus, or *ii)* our hypothesis that the pairs in a *Restatement* relation express also entailment does not hold.

Manually analyzing a set of pairs from the data set for error analysis, we realized that both reasons we hypothesized are (partially) true. As introduced before, TE definition is based on (and assumes) common human understanding of language, as well as common background knowledge. However, the entailment relation is said to hold only if the statement in the text licenses the statement in the hypothesis, meaning that the content of T and common knowledge together should entail H, and not background knowledge alone. For instance, let's consider Example (12).

(12) T: The earlier generation of our crowd bankers had stressed above all probity, tradition,

[8] http://search.cpan.org/tpederse/Text-Similarity-0.08/lib/Text/Similarity/Overlaps.pm

continuity and reputation.
H: They were old-fashioned elegant gentlemen.

Even assuming that in H "They" co-refers with the "bankers" in T, according to the definition of TE this is not a positive example, since the amount of background knowledge to be assumed to judge this pair asides from information provided by T. Quite a lot of the pairs tagged as *Restatement* in the PDTB and present in our data set fall into that category, and cannot therefore be considered as positive textual entailment pairs. At the same time, what the literature assumes as background knowledge to be introduced in the inference process is not completely clear (see the debate among Dagan et al. (2006), Manning (2006) and Zaenen et al. (2005)), making the assignment of the entailment judgment to such pairs particularly difficult.

In order to understand and prove if the low performances obtained by the TE systems in our first experiment are due to the presence of *Restatement* sentences that are negative entailment pairs, we conduct a second experiment on a reduced data set.

4.3 Experiment 2: reduced data set

To verify the correctness of our initial hypothesis, i.e. that the sentences annotated as being into a *Restatement* relation express entailment, we run a second experiment focusing on a manually-annotated subset of pairs, as described in the following sections.

4.3.1 Data set description

To investigate to what extent the *Restatement* type is related to entailment, we manually annotated a subset of sentences randomly extracted from the positive examples of Experiment 1 as being *Entailing*, *Non Entailing* or *Entailing with Coreference*. The first two labels were assigned following the RTE annotation guidelines, while the third one was introduced because we observed that in many cases the content of T and H could be put into relation only assuming that coreference was resolved.

We assign a *Coreference* label and not an *Anaphoric* one because we do not limit this analysis to sentences in which some entities are identical, but we also cover pairs in which some information is implicit, and a coreference relation different from identity holds between the two (e.g. bridging). Since the pairs were extracted from adjacent sentences in real documents, this phenomenon is very frequent, because coreference is frequently used as a device to improve textual cohesion. Note that the antecedent is not always in T, as in the following example showing a *Restatement.generalization* relation, in which "She" in T may be resolved through "Marie-Louise (called Marie Latour in the film)" in H:

(13) T: She was untrained and, in one botched job killed a client. Her remorse was shallow and brief. Although she was kind and playful to her children, she was dreadful to her war-damaged husband; she openly brought her lover into their home.
H: Marie-Louise (called Marie Latour in the film) was not a nice person.

In some cases, one of the two sentences is a direct speech restating what was described in the other sentence. Also for these cases, we adopted the *Coreference* label. As a clarification, we report the sentence pair (14): the pronouns "He" and "me" in H should be resolved by "Mr. Sorrell" and "Mr. Roman" in T.

(14) T: But Mr. Roman flatly denied the speculation, saying Mr. Sorrell had tried several times to persuade him to stay, offering various incentives and in one instance sending a note with a case of wine.
H: He asked me not to resign.

We are aware that this kind of sentences would not be considered entailing in standard RTE data sets, but we decided to mark them as *Entailing with Coreference* because direct speech is very frequent in newswire documents, on which the PDTB is based, and we wanted to account also for these cases.

We annotated 160 sentence pairs for each of the four Restatement subtypes (*Restatement.specification*, *Restatement.generalization*, *Restatement.equivalence* and generic *Restatement*), thus collecting 640 annotated pairs. Two annotators were involved in the task. Each annotated independently 240 pairs, while 160 additional pairs were annotated by both, so as to compute inter-annotator agreement.

While the percentage of agreement between the two annotators is 84%, weighted kappa is 0.59. As a rule of thumb, this is a satisfactory agreement, although it reflects the fact that the great majority of assignments are "not entailing", making the probability of chance agreement very high. It is interesting to note that only in one case annotators disagree on whether an entailing relation is also coreferential, meaning that this distinction is well founded and linguistically motivated. In RTE tasks, agreement (Fleiss' Kappa) is usually around 0.98 after reconciliation. Although it is not directly comparable to our agreement (we apply weighted kappa and we do not perform reconciliation), this may reflect the fact that our hypotheses are not manually defined, thus their level of complexity is higher than in standard RTE tasks. As an example, we report in (15) a *Restatement.equivalence* relation on which the annotators disagree. The sentence in H is a sort of lesson that can be drawn from T:

(15) T: The problem is, if people get down in the dumps, they stop selling.
H: Discouragement feeds on itself.

In order to build the final data set, we removed the pairs on which the annotators disagreed (26 in total) and then merged the other annotations. The data set we obtain includes 614 pairs connected through some type of *Restatement* relation and annotated as *Entailing*, *Not Entailing* or *Entailing with Coreference*. Some statistics on the data are reported in Table 2.

Relation Type	**YES**	**YES-COREFERENCE**	**NO**
Specification	7	7	144
Generalization	26	7	123
Equivalence	39	11	93
Restatement	17	16	124
Overall	89	41	484

Table 2: Statistics on manually annotated data in the reduced data set

We observe that most of the pairs are not entailing, and this is generally due to: *i)* the presence of additional information in H not present in T (so the truth of H cannot be verified), as in Example (16), or to *ii)* the presence of not relevant information in H, as in Example 17.

(16) T: Each right entitles the shareholder to buy $100 face amount of 13.5% bonds due 1993 and warrants to buy 23.5 common shares at 30 cents a share.
H: Under the offer, shareholders will receive one right for each 105 common shares owned.

(17) T: It responds to it.
H: Arbitrage doesn't cause volatility.

These results provide a first answer to the research questions posed in Section 1: despite the definition of the *Restatement* relation in the PDTB, it does not exactly match with the textual entailment relation. Nevertheless, there is an overlap between the two: the annotation suggests that entailing pairs may be a subclass of *Restatement*. As regards the different *Restatement* types, *Specification* is the least related to entailment, with 91% of the pairs annotated as not entailing. This depends on the fact that T and H appear in the original documents in a reversed order, with H typically containing additional information which is not in T. Surprisingly, the relation type showing highest proportion of entailing pairs is *Equivalence* (35% of the examples), and not *Generalization* (21%). In fact, the latter includes many cases in which H is a sort of motto that cannot be inferred directly from T. *Equivalence*, instead, implies less abstraction and is often a reformulation of T at lexical level. Intuitively, pairs into an Equivalence relation are expected to be paraphrases (bidirectional entailment relation), but it is not always the case. For instance, Example (8) can be considered as a positive example of entailment (T⇒ H) once coreference is resolved (both T and H are talking about the same event), but the opposite does not hold, i.e. H does not entail T.

The pairs manually annotated as entailing (130 in total) were used to build a second data set. To balance this new data set with respect to positive and negative pairs in order to run our experiments, we randomly extracted 130 pairs from the negative examples of Experiment 1 (Section 4.2). In the end, we created a data set including 260 pairs, equally divided into positive and negative examples. Compared to the extended data set, all positive pairs in this reduced version have been manually checked, so that they are certain examples of Restatement *and* entailing relation.

4.3.2 Results and discussion

We re-run EDITS and VENSES on the reduced data set to see whether the systems perform better on correct manually annotated entailing pairs. We prepared two versions of the data set: one includes the pairs as they are, while in the other the entailing pairs marked as coreferring were manually resolved, e.g. pronouns were replaced by their extended form, bridging relations were made explicit, and so on.

Since EDITS requires supervision, we split the data set randomly selecting 160 pairs (balanced with respect to positive and negative pairs) to be used for training and 100 for testing. Again, VENSES was run only on the test set, in both the shallow and the deep configuration. We also computed a word-overlap baseline, as in Experiment 1. Results on both versions of the data set (*with* and *without* coreference resolution) are reported in Table 3.

We observe that when running our experiments on the new data set without coreference resolution, both VENSES and EDITS strongly outperform the baseline, while in Experiment 1 there was no significant difference between the three systems. With VENSES, the deep

	Baseline	**VENSES** (d.)	**VENSES** (s.)	**EDITS** (wo)	**EDITS** (cos)
Without coref. res.					
Positive Pairs	*38.00*	40.00	54.00	60.78	38.81
Negative Pairs	*66.00*	82.00	84.00	59.18	69.17
Overall	*52.00*	**61.00**	**69.00**	**60.00**	**59.00**
With coref. res.					
Positive Pairs	*36.00*	56.00	46.00	48.48	41.1
Negative Pairs	*80.00*	82.00	84.00	49.5	66.14
Overall	*58.00*	**69.00**	**65.00**	**49.00**	**57.00**

Table 3: Systems performances on test set (% correctly classified pairs)

version performs better after coreference resolution, but the shallow one achieves a better performance on the original data. Even if EDITS performances are better then the baseline on the original data set, they drop on the reduced data set (in particular, the configuration based on the word overlap algorithm). One of the reasons for that can be seen in the complexity of the pairs extracted from the PDTB, where lexical overlap between T and H is close to 0, and therefore word overlap algorithms fail to correctly detect the positive entailment pairs. VENSES performances show in fact that a linguistically-motivated system including some semantics in the process performs better on such pairs. Moreover, EDITS is negatively influenced by the small size of the data set, since only 160 pairs are used for system training, while for the RTE challenges about 1000 pairs are used for this goal. As short term future goal, we plan to re-run those experiments exploring more customized configurations of EDITS (combining different edit distance algorithms), and including entailment rules to provide it with semantics. In any case, all approaches (including the baseline) improve significantly compared to Experiment 1. On the manually annotated data set the systems performance is comparable to the performance achieved in RTE tasks.

The results of Experiments 1 and 2 allow us to provide some answers and observations concerning the second and the third research questions posed in Section 1. The second question was asking whether it is possible to use RTE systems to identify implicit *Restatement* relations. Current approaches have proved to have some limitations and achieve poor performance, obtaining on average 0.35 F1 on the detection of implicit *Restatement* relations (Lin et al., 2009). Experiment 2 shows that currently available RTE systems outperform such results, and could therefore represent an interesting direction to explore to accomplish the task of identifying implicit *Restatement* relations on a subset of PDTB sentences (i.e. sentences tagged as *Restatement*, where the first argument entails the second one).

The third research question was asking - from a TE perspective - whether it is possible to use entailing pairs extracted from the PDTB to train or evaluate TE systems. As showed in the error analysis following Experiment 1 (Section 4.3.1), less than 1/4 of the *Restatement* pairs are also entailment pairs. At the same time, this subset of *Restatement* and entailment pairs contain interesting and particularly complex pairs, since the relation of inference between T and H relies less on lexical-syntactic variations and more on background knowledge and reasoning. Moreover, such pairs contain entailment phenomena that up to now have not been part of the research proposed by the RTE evaluation campaigns organizers. For instance, in a set of sentences from the PDTB the second argument (i.e. the fragment we consider as H) is a motto, or a metaphor (see Examples (8) and (9)). The presence of such types of arguments,

that are easy to understand and solve for humans thanks to their knowledge of the world but almost impossible for machines, could bring new light into textual inference. Indeed, it would encourage the exploration of categories of entailment phenomena that up to now TE systems are not able to face, but that, due to their frequency in real data, cannot be disregarded.

5 Conclusion and future perspectives

In this paper, we provide an analysis of the relation between *Restatement* and textual entailment in the PDTB. Starting from their (similar) theoretical definition, we empirically verified if systems developed for textual entailment recognition can be successfully used to detect implicit *Restatement* relations. Although this first experiment proved that RTE systems are not effective in the task, a manual annotation of the experimental data set showed that part of the *Restatement* examples are also entailing, suggesting that the relation of textual entailment may be a subset of *Restatement* relation. Therefore, RTE techniques could be explored to solve the task on this subset of sentences. Data annotation allowed us to observe also that the *Specification* subtype is the least correlated to entailment cases and that coreference resolution is crucial in identifying entailing sentences in real texts, in line with past research on this topic (Mirkin et al., 2010a,b).

We further showed that the entailing pairs being in a *Restatement* relation represent interesting and particularly complex pairs, because they contain entailment phenomena that are not yet considered in the data provided for RTE evaluation campaigns. Taking into consideration also these cases extracted from real data would bring new light into research on textual inferences, in line with the most recent editions of RTE challenges that are now moving toward more real scenarios. In order to foster this new interesting research direction, the manually tagged pairs used for Experiment 2 will be made available at `http://hlt.fbk.eu/en/people/tonelli/Resources`.

Several research lines have to be considered as future research. As a first step, we plan to improve our experimental evaluation with different respects: *i)* augmenting the size of the reduced data set, in particular annotating more pairs of the PDTB to obtain more TE pairs for RTE systems training and evaluation; *ii)* customizing the EDITS system configuration to increase its performances; *iii)* experimenting with different available RTE systems to compare several approaches to RTE (e.g. logic-based, Machine Learning) on these particularly complex data and *iv)* including other pairs belonging to the EXPANSION class (but not labeled as *Restatement*) in our data set.

Moreover, we would like to verify if the relation between *Restatement* and textual entailment holds also in the other direction, i.e. if positive pairs in RTE data sets would actually be labeled as cases of *Restatement*, and if one of the *Restatement* subtypes can be more frequently associated with positive pairs. Nevertheless, such an annotation could be problematic because *Restatement* is usually defined in the discourse context and taking two sentences in isolation without the surrounding discourse is not likely to lead to meaningful annotations.

Acknowledgments

This work was partially supported by the EC-funded project EXCITEMENT (FP7 ICT-287923). The authors would like to thank Marco Guerini for providing useful feedback on earlier versions of this work. We thank also the PDTB group and the participants to the PDTB Workshop 2012 at IRCS for valuable discussion and insights.

References

Barzilay, R. and McKeown, K. (2005). Sentence fusion for multidocument news summarization. *Computational Linguistics*, 31(3):297–327.

Bentivogli, L., Cabrio, E., Dagan, I., Giampiccolo, D., Leggio, M. L., and Magnini, B. (2010). Building Textual Entailment Specialized Data Sets: a Methodology for Isolating Linguistic Phenomena Relevant to Inference. In *Proceedings of the 7th International Conference on Language Resources and Evaluation (LREC)*, Valletta, Malta. 19-21 May.

Dagan, I., Dolan, B., Magnini, B., and Roth, D. (2009). Recognizing textual entailment: Rationale, evaluation and approaches. *Natural Language Engineering (JNLE)*, 15(Special Issue 04):i–xvii.

Dagan, I. and Glickman, O. (2004). Probabilistic Textual Entailment: Generic Applied Modeling of Language Variability. In *Proceedings of the PASCAL Workshop on Learning Methods for Text Understanding and Mining*, Grenoble, France. 26-29 January.

Dagan, I., Glickman, O., and Magnini, B. (2006). The PASCAL Recognizing Textual Entailment Challenge. In *MLCW 2005, LNAI Volume 3944*. Springer-Verlag.

Delmonte, R., Tonelli, S., and Tripodi, R. (2009). Semantic Processing for Text Entailment with VENSES. In *TAC 2009 Proceedings Papers*. NIST - National Institute of Standards in Technology.

Elwell, R. and Baldridge, J. (2008). Discourse Connective Argument Identification with Connective Specific Rankers. In *Proceedings of ICSC-2008*, Santa Clara, United States.

Ghosh, S., Johansson, R., Riccardi, G., and Tonelli, S. (2011a). Shallow Discourse Parsing with Conditional Random Fields. In *Proceedings of the 5th International Joint Conference on Natural Language Processing (IJCNLP)*, Chiang Mai, Thailand.

Ghosh, S., Tonelli, S., Riccardi, G., and Johansson, R. (2011b). End-to-End Discourse Parser Evaluation. In *Proceedings of the Fifth IEEE International Conference on Semantic Computing (ICSC 2011)*, Palo Alto, United States.

Kouylekov, M. and Negri, M. (2010). An Open-Source Package for Recognizing Textual Entailment. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics (ACL 2010) System Demonstrations*, Uppsala, Sweden. 11-16 July.

Lin, Z., Kan, M.-Y., and Ng, H. T. (2009). Recognizing Implicit Discourse Relations in the Penn Discourse Treebank. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 343–351, Singapore.

Lin, Z., Ng, H. T., and Kan, M.-Y. (2010). A PDTB-Styled End-to-End Discourse Parser. Technical Report TRB8/10, School of Computing, National University of Singapore.

Manning, C. (2006). Local textual inference: it's hard to circumscribe, but you know it when you see it - and NLP needs it. In *Proceedings of the Eighth International Conference on Computational Semantics (IWCS-8)*, Unpublished manuscript. 25 February.

Marcus, M., Santorini, B., and Marcinkiewicz, M. (1993). Building a Large Annotated Corpus of English: the Penn Treebank. *Computational Linguistics*, 19(2):313–330.

Mirkin, S., Berant, J., Dagan, I., and Shnarch, E. (2010a). Recognising Entailment within Discourse. In *Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010)*, pages 770–778, Beijing, China. Coling 2010 Organizing Committee.

Mirkin, S., Dagan, I., and Pado, S. (2010b). Assessing the role of discourse references in entailment inference. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 1209–1219. Stroudsburg, PA, USA.

Nielsen, R. D., Ward, W., and Martin, J. H. (2009). Recognizing entailment in intelligent tutoring systems. *The Journal of Natural Language Engineering, (JNLE)*, 15:479–501.

Pitler, E., Louis, A., and Nenkova, A. (2009). Automatic sense prediction for implicit discourse relations in text. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 683–691, Suntec, Singapore.

Pitler, E. and Nenkova, A. (2009). Using syntax to disambiguate explicit discourse connectives in text. In *Proceedings of the 47th Annual Meeting of the Association for Computational Linguistics and the 4th International Joint Conference on Natural Language Processing*.

Pitler, E., Raghupathy, M., Mehta, H., Nenkova, A., Lee, A., and Joshi, A. (2008). Easily Identifiable Discourse Relations. In *Proceedings of the 22nd International Conference on Computational Linguistics (Coling 2008)*, pages 87–90, Manchester, United Kingdom.

Prasad, R., Dinesh, N., Lee, A., Miltsakaki, E., Robaldo, L., Joshi, A., and Webber, B. (2008). The Penn Discourse Treebank 2.0. In *Proceedings of the 6th International Conference on Language Resources and Evaluations (LREC 2008)*, Marrakech, Morocco.

Prasad, R., Joshi, A., and Webber, B. (2010). Exploiting Scope for Shallow Discourse Parsing. In *Proceedings of the Seventh conference on International Language Resources and Evaluation (LREC'10)*, Valletta, Malta.

Robaldo, L., Miltsakaki, E., and Bianchini, A. (2010). Corpus-based Semantics of Concession: Where do Expectations Come from? In *Proceedings of LREC*, pages 3593–3600.

Romano, L., Kouylekov, M. O., Szpektor, I., Dagan, I., and Lavelli, A. (2006). Investigating a Generic Paraphrase-Based Approach for Relation Extraction. In *Proceedings of the 11th Conference of the European Chapter of the Association for Computational Linguistics (EACL 2006)*, Trento, Italy. 3-7 April.

Sammons, M., Vydiswaran, V., and Roth, D. (2010). Ask Not What Textual Entailment Can Do for You... In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics (ACL-10)*, Uppsala, Sweden. 11-16 July.

Wellner, B. and Pustejovsky, J. (2007). Automatically Identifying the Arguments of Discourse Connectives. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 92–101, Prague, Czech Republic.

Zaenen, A., Karttunen, L., and Crouch, R. (2005). Local Textual Inference: Can it be defined or circumscribed? In *Proceedings of the Workshop on the Empirical Modeling of Semantic Equivalence and Entailment*, Ann Arbor, MI. 30 June.

Implicitness of Discourse Relations

Fatemeh Torabi Asr and Vera Demberg
Cluster of Excellence Multimodal Computing and Interaction (MMCI)
Saarland University
Campus C7.4, 66123 Saarbrücken, Germany
fatemeh@coli.uni-saarland.de, vera@coli.uni-saarland.de

Abstract
The annotations of explicit and implicit discourse connectives in the Penn Discourse Treebank make it possible to investigate on a large scale how different types of discourse relations are expressed. Assuming an account of the Uniform Information Density hypothesis, we expect that discourse relations should be expressed explicitly with a discourse connector when they are unexpected, but may be implicit when the discourse relation can be anticipated. We investigate whether discourse relations which have been argued to be expected by the comprehender exhibit a higher ratio of implicit connectors. We find support for two hypotheses put forth in previous research which suggest that continuous and causal relations are presupposed by language users when processing consecutive sentences in a text. We then proceed to analyze the effect of Implicit Causality (IC) verbs (which have been argued to raise an expectation for an explanation) as a local cue for an upcoming causal relation.

Keywords: Causality, Continuity, Discourse relations, Discourse cues, Implicit discourse relations, Corpus study, Uniform Information Density.

Proceedings of COLING 2012: Technical Papers, pages 2669–2684,
COLING 2012, Mumbai, December 2012.

1 Introduction

David Hume, in his prominent work "An enquiry concerning human understanding" proposed that ideas in the human mind were associated according to at least three types of relations: resemblance, contiguity in time or place, and causality (Hume, 1784). Since then, many language scientists have tried to adapt this idea about human general reasoning to the world of language (Simon, 1952; Hobbs, 1990; Kehler, 2000). A *discourse relation* as defined by linguists is an inference intended by the writer or made by a reader to establish local coherence among individual sentences. For example, a binary *causal* relation refers to a logical cause-consequence inference whose elements are directly accessible in the text: two propositional arguments plus an operator that could be an explicit connective such as "because" (1-a). The operator might sometimes not be explicit in the text, but in most cases a suitable connective could be inserted to specify the coherence relation between the involved propositions (1-b).

(1) a. Bill took his daughter to the hospital, because she looked pale and sick in the morning.
    b. I was very tired last night. [Therefore] I went to sleep earlier than usual.

According to some cognitive theories on discourse processing, people have expectations about inter-sentential relations when reading a text, which bias their inferential decisions during comprehension (Segal et al., 1991; Murray, 1997; Levinson, 2000; Sanders, 2005). Two important characteristics that are expected to exist between consecutive events in a text are continuity and causality. Segal et al. (1991) and Murray (1997) argue that readers expect a sentence to be causally congruent and continuous with respect to its preceding context. Continuity in the sense of Segal et al. means that the same frame of reference is maintained, for example by subsequent sentences talking about the same event, without shift in perspective. Continuous discourse relations are claimed to be easier to process and more expected than other types. On the other hand, relations that are discontinuous (for example adversatives) would be less expected and more difficult to process. This notion also includes temporal continuity, implying that a non-linearity in presenting a consequence before its effect or any two situations in a reverse temporal order (1-a) is less expected than a relation keeping the forward temporal transition between events (1-b). A second hypothesis about what kind of discourse relations are typically assumed by comprehenders was proposed by Sanders (2005) in his "causality-by-default" hypothesis, which states that language users prefer causal relations to other types (such as mere expansion or temporal relations) when establishing discourse coherence.

To investigate the validity of these hypotheses in the experimental paradigm, researchers have studied participants' sentence completion preferences, coherence judgments, and reading patterns during reading of the sentence pairs implying different types of discourse relations (Irwin, 1980; Trabasso et al., 1984; Caron et al., 1988; Millis et al., 1995; Murray, 1997; Kuperberg et al., 2011). We are, however, not aware of a large-scale study of these hypotheses in naturally occurring texts. To address this, we analyzed the use of causal/temporal relations and their markers in the Penn Discourse Tree Bank (PDTB; Prasad et al., 2008), a large corpus of newspaper texts which is annotated with discourse relations.

Our methodology relies strongly on the assumption of *implicitness of the discourse connector as a sign of expectation of the discourse relation*: if readers have a default preference to infer a specific relation in the text, this type of relation should tend to appear without explicit markers. The assumption is drawn on a usage-based approach to the study of language preferences that links comprehension phenomena with typicalities in production (Langacker, 2000). It also can be thought of in terms of the Uniform Information Density (UID) hypothesis (Frank and Jaeger, 2008) which suggests that humans tend to spread information evenly across a text or utterance, thereby reducing or omitting redundant optional markers (e.g., Levy and Jaeger, 2007; Florian Jaeger, 2010). At the level of inter-sentential relations, this would mean that the presence of explicit connectives is necessary when the relation is unexpected, but that a connective may be implicit if the relation is predictable. Hence, we investigate the validity of the following predictions in the corpus:

1. continuous discourse relations should be implicit more often than discontinuous ones
2. causal relationships should be implicit more often than other discourse relationships
3. relations which keep the forward temporality should be implicit more often than their backward counterparts.

These predictions refer to general tendencies of comprehenders to expect that phrases and sentences are linked causally, and that the sentences in e.g. a narrative are interpreted in a continuous manner. Beside these general expectations regarding the upcoming discourse relation, expectations can also be influenced by local factors: (Pitler et al., 2008) found that discourse relations in the PDTB are not independently distributed. For example, explicit comparisons are significantly more often followed by implicit contingencies than would be expected under the independence assumption. Furthermore, Sanders (1997) and (Sagi, 2006) showed that the genre of a text, or more specifically, the distribution of different types of relations in a particular text can shape the expectation of a reader about what relations will appear more frequently later in the text.

The discourse relation between two sentences can also become clear by other means such as shared entities, adverbial phrases and even the type of verbs used in the arguments. An interesting case are so-called implicit causality (IC) verbs (as the verb "scolded" in (2-a); see Section 5.6 for more details), which have recently been argued to trigger the expectation for a reason (see (2-b)) to be communicated in the following sentence (Rohde and Horton, 2010), such that the discourse relation between the sentences is a backward causal relation.

(2) a. Arthur scolded Patricia.
    b. She had put thumbtacks on the teacher's chair.

According to the UID hypothesis, we would expect that causal relations which contain an IC verb in their first argument (Arg1) would be expressed without an explicit connective more often, as the reason relation is predictable. In other words, as our third hypothesis, we investigate whether:

4. backward causal relations that contain an IC verb (which already marks causality) in their first argument are implicit more often than those that contain a non-IC verb.

This work strives to get a better understanding of when discourse relations are expressed implicitly, and when they should be explicitly stated. Furthermore, it can possibly inform work on automatic identification of discourse relations, especially in the absence of explicit discourse markers (Sporleder, 2008; Lin et al., 2009; Zhou et al., 2010). The rest of the paper is structured as follows. Section 2 includes a review of the causality-by-default and the continuity hypotheses and the related experimental studies. In Section 3, we introduce the PDTB corpus and the relevant relations we extract to obtain evidence for the aforementioned hypotheses. Section 4 will describe our method of analysing implicitness and Section 5 includes the results. Finally, conclusions are presented in Section 6.

2 Background on the Continuity and Causality Hypotheses

In this section we explain the *continuity* hypothesis (Segal et al., 1991; Murray, 1997) and the *causality-by-default* hypothesis (Sanders, 2005). Taken together, these hypotheses suggest that language users first try to establish causal relatedness and temporal linearity between phrases when processing a text.

2.1 The Continuity Hypothesis

Levinson (2000) notes in his discussion on presumptive meanings that *"when events are conjoined they tend to be read as temporally successive and if at all plausible, as causally linked."*. An early notion of the continuity hypothesis proposed by Segal et al. (1991) suggests that discourse connectives are used to mark the deictic continuity or discontinuity in texts (their study was limited to narratives). As Murray (1997) states, the hypothesis is that comprehension difficulty ensues when a text event is discontinuous without this discontinuity having been explicitly marked. Segal et al. (1991) found that comprehenders expect subsequent sentences to be causally congruent and temporally continuous. Support for the continuity hypothesis comes also from a series of experiments by Murray (1997). In an initial sentence completion experiment, subjects had to write a continuation to the preceding discourse starting with a connective that was either additive, causal, or adversative. Murray confirmed that sentences generated in response to an additive or causal connective generally depicted continuous events with respect to the preceding discourse, while completions following adversative connectives depicted discontinuous events. In a subsequent experiment, he asked people to read sentence pairs with inappropriately placed connectives, and found that the disruptive effect was largest for inappropriately placed adversative connectives. Sentences with inappropriately placed adversative connectives were also judged as less coherent than ones with inappropriately placed causal or additive connectives. In particular, there was no difference in processing disruption or coherency judgment between inappropriately used additive and causal connectives, which is why Murray attributes the difference between these connectors and adversative ones to the underlying continuity or discontinuity of the described event. Murray interprets his findings in terms of readers generally expecting a continuous event and their processing being more disrupted by a connective that signals an upcoming discontinuity, than by one that would signal continuity. Taken together with an earlier experiment (Murray, 1995), which showed that correctly placed adversative connectors also have a greater beneficiary effect than correctly placed causal or additive connectors, Murray concludes that adversative connectives are more *salient* than connectives that signal continuity.

While he generally classifies causal relations as continuous ones, Murray also notes that a

connective like "because" which signals a temporally non-linear causal relation (*backward* transition from the effect to the cause) should have stronger contextual effects than connectives such as "so" or "therefore". In the literature, some types of adversative relations are interpreted as *negative causal* relations, e.g., a pair of sentences connected with *although* implies a causal relation in which an unexpected consequence has happened (König, 1991). According to Segal et al. (1991) and Murray (1997), such relations — usually referred to as *concession* — are not expected to the same degree as positive causal relations, which benefit from a higher degree of continuity. The continuity hypothesis also predicts that temporal relations between sentences which cue a non-linear relationship between the arguments would be more difficult to process, and that cues for temporal non-linearity (such as "after", as opposed to "before", which indicates the expected temporal order) should be more salient.

2.2 The Causality-by-default Hypothesis

The main motivation behind our study of causal relations comes from Sanders's cognitive theory of discourse representation. Specifically, the causality-by-default hypothesis states:

"because experienced readers aim at building the most informative representation, they start out assuming the relation between two consecutive sentences is a causal relation" – Sanders (2005) .

Experimental evidence for this claim comes from a range of studies on understanding causal relations in narratives (Trabasso et al., 1984), the effect of connectives on recall of inter-sentential relations (Irwin, 1980; Caron et al., 1988) and preferences in sentence completion tasks (Murray, 1997). A recent study of online sentence processing furthermore reveals that causal relations between sentences facilitate processing even in the absence of discourse connectors: Kuperberg et al. (2011) finds that a small discourse consisting of three sentences was easier to process when the sentences were causally related. Specifically, a larger N400 (an EEG signal which typically indicates semantic anomalies) was found when sentences were irrelevant. All of these findings suggest that readers have a prior expectation that consecutive sentences in a text should be causally related and congruent, unless an explicit cue such as adversative connectives (e.g., *but*) provides marking for another type of relation. The most relevant experiment specific to the tendency towards causal inference is again the one by Murray (1997) in which subjects were asked to continue individual sentences that ended with either a period or a connective of the aforementioned types (additive, causal or adversative). The majority of the answers for the no-connective condition conveyed a causal relation, meaning that subjects often chose a type of continuation that provided a cause or a consequence for the given sentence instead of a simple additive continuation or an adversative one.

However, arguments against the causality-by-default hypothesis can also be found in the literature. Millis et al. (1995) performed an experiment where two consecutive sentences (that did not stand in an obviously causal relationship) were connected with a full stop, or one of the three discourse connectors "because", "and" or "after", as the indicators of causal, additive and temporal relations, respectively. The sentence pairs inherently could be interpreted as expressing any of the mentioned relation types. Millis at al. found that causal inferences (as measured by asking participants a "Why?" question after pair of sentences) were only reliably made in the "because" condition, but not in the conditions where the sentences were connected by a period or one of the other connectors. They concluded that

the discourse marker "because" played a very important role in people's forming of an inference, and that this inference was not formed automatically in these contexts.

The studies we reviewed are all small scale and use carefully designed experimental materials; it is, however, an open question whether the hypotheses generated based on the experimental studies also hold for naturally occurring texts. An additional difficulty is that results from previous studies are often not easily comparable, as they use a slightly different taxonomy of discourse relation types (e.g., *adversative* relations in Murray (1997) includes both negative causal and negative additive relations, while in other related studies such as König (1991),Couper-Kuhlen and Kortmann (2000), and Köhne and Demberg (2011) the former is referred to as a *concession* relation). In this paper, we use the discourse relation categorization of the Penn Discourse Treebank and see how different sentence connectives are being used in naturally occurring text with respect to the extent they reflect causality and continuity.

3 An Overview of the PDTB Corpus

Penn Discourse Tree Bank (PDTB; Prasad et al., 2008) is a large corpus of texts from the Wall Street Journal, which is annotated with discourse relations between every pair of adjacent clauses, sentences or larger text spans[1]. The PDTB covers all 25 sections of the Penn Treebank, which has been annotated with various other linguistic information such as syntactic structures and semantic frames.

Discourse relations can either be marked explicitly with a discourse connector (such as "because", "nevertheless", "and"), or be implicit[2]. Both explicit and implicit discourse relations are annotated in the corpus, and each discourse relation is marked with a discourse connector and two propositional arguments. In the case of implicit relations, a suitable discourse connector was identified (and inserted) by the annotators. Each argument is an independent text segment whose boundaries are also determined by the annotators. Labeling of the relations has been done according to a hierarchy of discourse relation senses (see Figure 1), including four top-level classes: `CONTINGENCY`, `COMPARISON`, `TEMPORAL` and `EXPANSION`.

Temporal Relations `TEMPORAL` relations include `Synchronous` and `Asynchronous` types. `Asynchronous` relations have two subtypes `precedence` and `succession`, which mark forward (3-a) and backward (3-b) temporal transitions respectively.

(3) a. He believes [that $55 a share is the most you can pay for Georgia Gulf], before [it becomes a bad acquisition].
b. [The fields were developed], after [the Australian government decided in 1987 to make the first 30 million barrels from new fields free of excise tax].

We classify `Asynchronous` temporal relations as markers of *discontinuity*, following (Segal et al., 1991). Among the `Asynchronous` temporal relations, the ones where events are in the correct temporal order should be easier to process than the ones where temporal order is

[1]For details on the choice of text spans and adjacency we refer the reader to Prasad et al. (2008).

[2]Other than explicit and implicit relations, the corpus contains two other categories of relations, namely EntRel, indicating the relation between sentences only according to the common entities, as well as AltLex, in which the relation between the two arguments is not lexicalized via the defined set of connectives. We do not include AltLex category in our analysis as it only contains one percent of all the tagged relations.

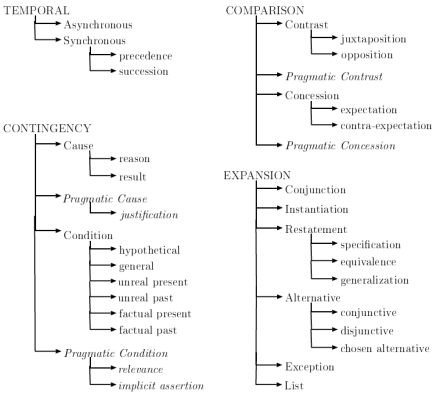


Figure 1: Hierarchy of senses in PDTB (Prasad et al., 2008)

reversed. `Synchronous` temporal relations are harder to classify: they sometimes introduce new events and should thus be classified as discontinuous.

Contingency Relations Causal relations in the PDTB hierarchy are categorized as members of the `CONTINGENCY` family. `Cause` itself is divided into two subtypes, namely `reason` and `result`. These two subtypes, respectively, indicate forward and backward cause-consequence relations between their arguments as shown in (4-b) (`reason`) and (4-a) (`result`).

(4) a. [The governor couldn't come]$_{cause}$, so [the lieutenant governor welcomed the special guests]$_{consequence}$.
b. [There was some profit-taking]$_{consequence}$, because [prices for all precious metals had risen to levels at which there was resistance to further advance]$_{cause}$, he said.
c. [Mrs Yeargin is lying], because [they found students in an advanced class a year earlier who said she gave them similar help].

In addition to the `Cause` type in the `CONTINGENCY` class, there is a type called `Pragmatic Cause` which is much less frequent but still relevant to our study. It includes only one subtype, namely `justification`, indicating an epistemic causal relation in which the second argument provides a justification for a claim expressed in the first argument, see (4-c). In the literature, this has also been referred to as *diagnostic* relation (Traxler et al., 1997). In PDTB hierarchy conditional discourse relations are also categorized as `CONTINGENCY` relations, see Figure 1. They typically include "if" sentences but are not in the domain of our study, as we cannot classify them as either continuous / discontinuous or causal.

Comparisons All types of `COMPARISON` relations can be classified as discontinuous. As mentioned at the end of Section 2.1, König (1991) proposed that concessive relations are the dual of causal discourse relations in the sense that they involve a negative cause-consequence inference. In the PDTB, `Concession` relations are divided into two subtypes: `expectation` and `contra-expectation`, which represent as the duals of `reason` and `result`

respectively. In `expectation` relations (5-b) the second argument of the connective is a cause for something that is in contrast with the first argument, and in `contra-expectation` (5-a) the first argument of the connective is a cause for something in contrast to what the second argument asserts.

(5) a. [The demonstrators have been non-violent]$_{cause}$, but [the result of their trespasses has been to seriously impair the rights of others unconnected with their dispute]$_{neg-consequence}$.
b. [Third, oil prices haven't declined]$_{neg-consequence}$, although [supply has been increasing]$_{cause}$.

Other relation types of the comparison class, namely `contrast` and `pragmatic contrast` cover some of the adversative relations that have been compared with causal relations in studies such as Murray (1997).

Expansions Among the `EXPANSION` relations we classify as continuous the types `Instantiation`, `Restatement` and `List`. `Alternative` and `Exception` senses are obviously types of discontinuity. The remaining type, namely `Conjunction` relations are used in cases where "Arg2 provides additional, discourse new, information that is not related to Arg1 in any of the ways described for other types of `EXPANSION`." (The PDTB Research Group, 2008, p. 37). It is difficult to make one classification for the whole class - during manual inspection we found that these relations should sometimes be classified as *discontinuous* because they indicate a deictic shift in entity.

4 Methods

Given the continuity hypothesis, we would expect continuous relations to be very frequent in the corpus, and in particular more frequent than discontinuous relations. Similarly, we expect to see a relatively high frequency of causal discourse relations. An analysis of simple frequencies of total occurrences would however be rather limited, as there are a number of possible confounds. In particular, we would like to be able to test whether there is any evidence for people generating expectations of upcoming discourse relations, as argued by the continuity and causality literature.

To test whether comprehenders in fact do generate expectations about upcoming discourse relations, we draw on the uniform information density (UID) hypothesis (Frank and Jaeger, 2008). The UID hypothesis suggests that humans tend to spread information evenly across a text or utterance, and thereby use linguistic means in order to reduce or omit highly predictable linguistic material (which, because it is predictable, carries only a small amount of information and therefore would lead to a dip in information density if not reduced or omitted). Florian Jaeger (2010) show this effect for the optional "that" in English complement clauses, which they find is omitted more often if the complement clause is predictable from the verb than when it is not. It has been suggested that the function of such a constant rate of information density would be to facilitate information transfer (Genzel and Charniak, 2002). In analyzing the rate of implicit discourse connectors in function of the predictability of the discourse relation according to the continuity and causality-by-default hypotheses, we hypothesize that speakers are also able to dynamically choose to use an explicit discourse connector or drop it in order to achieve UID. In order to measure the ratio of implicit discourse connectors in a discourse relation, we define the

implicitness measure as follows:

$$Implicitness(relation) = \frac{\#\text{ of implicit } relation}{\#relation}$$

A large value of implicitness (in particular, larger than average implicitness among all relations, which is 0.46) for a particular discourse relation in the corpus would thus indicate that the relation is expressed without the use of a specific discourse connector more often than average. We interpret high values of implicitness as the producer not needing to explicitly specify the relation for the comprehender because it is predictable (or otherwise easily inferable from local cues[3]). It would also suggest that the annotators of the corpus tended to mark that relation even in the absence of direct textual signals. On the other hand, a small value of implicitness means that the discourse relation is expressed with an explicit discourse cue more often than average, and we would interpret that as the relation being not easily predictable or difficult to process, such that an explicit marker is needed to avoid a peak in information density.

5 Results

This section will first discuss evidence for general patterns of the use of discourse markers, and then proceed to analyze a specific case of IC verbs, which are a local cue of causality.

5.1 Analysis of Global Expectations

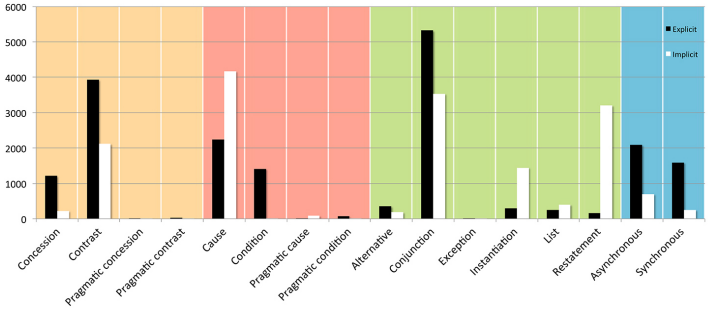


Figure 2: Frequency distribution of different types of relations in PDTB corpus: relation types are arranged across the horizontal axis according to their comprising classes: comparison, contingency, expansion and temporal.

Figure 2 depicts the distribution of different relation types over 19,009 explicit and 16,327 implicit considered relations[4]. There are 16 relation types in the hierarchy, some of which

[3]This notion of *no need for marking* of specific discourse relations such as causal ones reminds us also of Kehler's coherence theory, which indicates that arguments on syntactic decisions should be revisited with respect to the inferences underlying the establishment of discourse relations. For example, Kehler (2000) asserts that in elliptic structures, requirement of parallelism between the two involved discourse segments is important when a *resemblance* relation is targeted. In contrast, when a causal relation is being expressed via an elliptic construction, people understand it with no need of exact structural parallelism to help them with argument identification and alignment.

[4]Not all relations in the PDTB are annotated down to the third level of the hierarchy (this happened

are more frequent than others. The most frequent labels used to annotate the explicit relations (those associated with a textual discourse marker present in the original text) are `Conjunction`, `Contrast`, `Cause` and `Asynchrony`. It is obvious from the chart that the distribution of explicitly marked discourse relations is different from the distribution of implicit occurrence. The most frequent types among implicit relations are in order `Cause`, `Conjunction`, `Restatement`, and `Contrast`. In terms of total frequencies of the different discourse relationships (see Figure 2), `Conjunction`, `Cause` and `Contrast` are the most common relations, and hence could be claimed to be most expected on a pure frequency-based account. However, the relations `Conjunction` and `Contrast` have rather low rates of implicitness (see Figure 3).

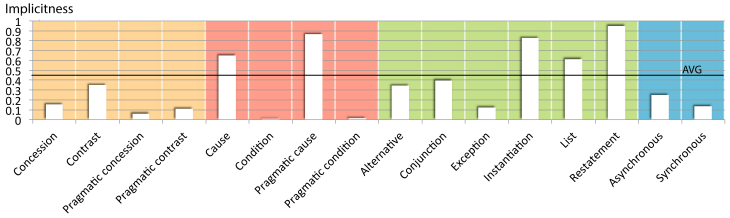


Figure 3: Implicitness of different types of relations in PDTB corpus: relation types are arranged across the horizontal axis according to their comprising classes: comparison, contingency, expansion and temporal.

5.2 Evidence for the Continuity Hypothesis

In Section 3 we classified the PDTB discourse relations with respect to continuity or discontinuity of an event in the sense of Segal et al. (1991). In particular, we argued that all of the discourse relations in the `COMPARISON` and `TEMPORAL`[5] family describe discontinuous events, while `Cause` and `Pragmatic Cause` are continuous. Within the `EXPANSION` family, we argued that `Instantiation`, `Restatement` and `List` are continuous, while `Exception` and possibly `Conjunction` can be viewed as denoting discontinuous events. Figure 3 depicts the implicitness of different level-2 types of discourse relations in the PDTB. We find that the relations that denote continuous events are exactly the ones that have implicitness values larger than average implicitness, while the ones that can be classified as discontinuous are exactly the ones with lower values of implicitness. The PDTB data thus provides strong supporting evidence for the continuity hypothesis. We will get back to more detailed analyzes concerning the concept of temporal continuity in Sections 5.4 and 5.5.

5.3 Evidence for the Causality-by-default Hypothesis

The causality-by-default hypothesis (Sanders, 2005) proposes that people preferentially interpret consecutive sentences as standing in a causal relationship. The data from the

when no consensus on the more detailed classification could been reached among the annotators). In our study, we only included those relations that were annotated down to at least the second level of the hierarchy. Also, some relations are tagged with more than one type. Hence, the sum over occurrences of all types does not necessarily match the total number of argument pairs in the corpus.

[5]Note: `Synchronous` is not straightforward to classify.

PDTB corpus shows that causal relations are not the only ones that are often expressed without an explicit connective, and hence with view on the UID hypothesis are not the only predictable relation (e.g., `Restatement` is a more implicit relation). Nevertheless, causal relations are the most frequent type of implicit discourse relations in the PDTB, see Figure 2. The implicitness of causal discourse relations is significantly higher (at 0.65) than the implicitness of the other frequent discourse relations, in particular `Conjunction` (0.39), `Concession` (0.15), `Asynchronous` (0.25), as well as the average overall relation types (0.46), – all comparisons significant at $p < 0.001$, according to a binomial test. This result supports the hypothesis we constructed on top of the causality-by-default and the UID hypotheses: in the absence of explicit connectives, a causal relation is expected between neighboring sentences.

5.4 Temporal Continuity

Now it is time to compare the subtypes of causal and temporal relations to investigate whether continuity in the temporal ordering of events is implicit or marked explicitly most often. Table 1 includes frequencies of the forward vs. backward causal, concessive and temporal relations in the corpus. As predicted, for each pair of the same type, the forward relation is associated with a higher degree of implicitness. We performed a binomial significance test and obtained $p < 0.001$ for all pairwise comparisons. This result supports the continuity hypothesis in the sense that discourse markers tend to be dropped when the relation between the arguments conform to linearity in time (Murray, 1997).

Type:subtype	Explicit	Implicit	Implicitness	Signif.
Cause:result	752	1704	**0.69**	***
Cause:reason	1488	2467	0.62	
Concession:contra-expectation	804	186	**0.19**	***
Concession:expectation	392	31	0.08	
Asynchrony:precedence	986	536	**0.35**	***
Asynchrony:succession	1101	161	0.12	

***: significant at $p < 0.001$ according to a binomial test

Table 1: Forward and backward occurrences of causal, concessive and temporal relations.

5.5 Textual Order of Arguments

It should be kept in mind that forward/backwardness of the relation (or the connective) in some cases does not correspond with the order in which the arguments of the relation appear in the text. Connectives such as "because" and "although" can take their subordinate clause (Arg2 according to PDTB) to the beginning of a composite sentence, see for example (6).

(6) Because the drought reduced U.S. stockpiles, they have more than enough storage space for their new crop, and that permits them to wait for prices to rise.

Since all of the occurrences of a particular relation type (e.g., *reason*) in the PDTB are tagged with the same label regardless of the textual order of the arguments, we performed a second analysis. Table 2 presents separate statistics for the Arg1-connective-Arg2 (ordered)

versus the connective-Arg2-Arg1 (reversed) occurrences for the relations `Cause`, `Concession` and `Asynchrony`, which we also focused on in our analysis of temporal continuity. PDTB annotators always put connectives between the arguments of the implicit relations. That is why we only present numbers for the explicit occurrences. Interestingly, there are always more ordinal modifications (the reversed presentation) when a backward relation of any type is being expressed. This implies that even in the presence of the cues, people have a tendency to keep the textual order of the arguments the same as the temporal order in which the associated events happened. In order to conduct a more accurate analysis of the temporal transition, given the information about argument organization we gathered all implicit and explicit occurrences of temporally linear vs. non-linear instances from all the 6 relation subtypes and performed a correlation analysis. In this analysis reversed occurrences of backward relations count as temporally linear and reversed occurrences of forward relations are taken as temporally non-linear (e.g., a reason relation in which Arg2 appears first in the text is taken as linear, just like a result relation). The chi-square test of the temporal linearity and the implicitness of the relation shows a significant correlation between these two factors ($\chi^2 = 67.31$, $df = 1$, $p < 0.001$). This is a more accurate result (compared with our analysis of continuity merely according to the relation types in 5.4) which indeed supports our hypothesis based on the UID and the continuity hypotheses: temporal forwardness is implicit in relations between consecutive sentences and its explicit cues are typically dropped.

Type:subtype (explicit only)	Ordered	Reversed	Signif.
Cause:result	746	6	***
Cause:reason	1324	**164**	
Concession:contra-expectation	791	13	***
Concession:expectation	183	**209**	
Asynchrony:precedence	931	55	***
Asynchrony:succession	867	**234**	

***: significant at p < 0.001 according to a binomial test

Table 2: Distribution of textually ordered vs. reversed occurrences of arguments in causal, concessive and temporal relations with explicit connectors.

5.6 A potential local predictor: Implicit Causality Verbs

Implicit Causality verbs (such as *adore*, *inspire*, *humiliate*) have been studied for many years, mostly in the context of coreference (whether the subject of the clause explaining the reason is the subject or object of the IC verb). Recently, IC verbs have however also been argued to make comprehenders anticipate an upcoming causal discourse relation, i.e. comprehenders expect to learn about the reason if they hear a sentence with an IC verb, like "Peter adored his older brother.": In a visual world experiment, Rohde and Horton (2010) compared sentences with IC verbs to sentences with transfer-of-possesion (TOP) verbs, and found that participants are much more likely to expect a reason following an IC verb sentence than when they have heard a sentence with a TOP verb. This indicates that people are able to take into account local cues like IC verbs and anticipate upcoming discourse relations.

To test on the corpus data the validity of the hypothesis that IC verbs lead people to anticipate causal relationships, we extracted all sentences from the PDTB which contained

an IC verb in the Arg1 of a discourse relation. In order to identify the IC verbs, we used a list of 300 IC verbs provided by Ferstl et al. (2011). In order to avoid noise in the data, we extracted only those instances where the IC verb was the main verb of an Arg1 which contained only a single sentence. Therefore, we simultaneously queried the PDTB annotation and the syntactic annotation of the Penn Treebank. To make sure that the IC verb worked as a cue in the sentence, we only considered relations with ordered arguments, namely the Arg1-connective-Arg2 occurrences. We found that the discourse relation was labeled as `reason` significantly more often if Arg1 contained an IC verb than when it did not ($p < 0.01$), however, the size of the effect was small: likelihood of `reason` given an IC verb in Arg1 was 14.0%, and 11.7% for other verbs. This is support (though relatively weak) for IC verbs actually affecting the upcoming discourse relation.

The more interesting question in the context of the UID hypothesis, however, is whether markers for causal relations following IC verbs are more likely to be absent, due to the added predictability of the `reason` relationship. We compared the implicitness of reason relations where the Arg1 contains an IC verb to the implicitness of reason relations with non-IC verbs as the head of Arg1. Counterintuitively, we found that the implicitness of reason relations with an IC verb in the Arg1 was smaller than for non-IC Arg1s. To make sure that this effect was not due to noise, we also checked all occurrences of IC verbs manually and only included the correct verbs and not homonyms (such as for *lie*) or other unintended semantic sense (e.g. "leave it up to somebody" instead of "leave somebody"). We found that such incorrectly tagged verbs were evenly distributed among implicit and explicit relations, such that the implicitness value was not affected by the noise (implicitness was 61% for IC verb reason relations and 65% for non-IC verb reason relations). Table 3 shows the manually checked numbers within reason relations (for all relations, only automatically extracted numbers are available due to the large number of occurrences, and absence of effect from manually checking the IC verbs in reason relations[6]).

	Total	IC verb in Arg1	
Implicit: reason relations	2462	153	(manually checked)
Explicit: reason relations	1324	96	(manually checked)
Implicit: all relations	15682	910	(automatically extracted)
Explicit: all relations	16147	1034	(automatically extracted)

Table 3: Total frequency of relations and the frequency of IC verbs appearing as the head of a single-sentence Arg1. All selected relations are also filtered to have ordered arguments.

The result goes against our predictions and merits further investigation in further work: all verbs need to be classified into their IC semantic class according to the previous research, and finer-grained verb sense disambiguation should also be considered[7], and further factors such as whether the verb occurs in passive voice could be taken into account. Also, the list

[6]The number of automatically extracted reason relations including IC verbs were 164 and 108 for the implicit and explicit occurrences, respectively.

[7]Ferstl et al. (2011) categorize IC verb usages into 4 classes: AgentPatient, AgentEvocator, Experiencer-Stimulus, and StimulusExperiencer. For example, the AgentPatient class covers *activity* transitive verbs such as *"carry"* which associate Agent and Patient roles to the involved entities. The 300 IC verb list that we employed also did not contain information about which fine-grained verb sense should be treated as an IC verb.

of 300 IC verbs from (Ferstl et al., 2011) are only annotated for their subject or object bias. In future work, these verbs should all be tested for the strength of predicting an upcoming reason relationship; and the result could be taken into account to see whether the prediction of higher implicitness of discourse cues signaling a reason relationship following IC verbs possibly holds for those IC verbs which strongly predict a reason relationship.

6 Conclusions

We conducted an empirical study of discourse relations in newspaper text, specifically the PDTB treebank, with respect to the causality-by-default and continuity hypotheses. We found supporting evidence for both hypotheses: As the *continuity hypothesis* (Murray, 1997; Segal et al., 1991) predicts, discourse relations which are discontinuous or temporally non-linear are much more likely to be expressed with an explicit discourse marker than those which are continuous. The statistics on the forward vs. backward temporal transition between arguments of discourse relations furthermore show a higher degree of implicitness for the forward directionality of all causal, concessive and temporal relations than for the backward versions of them.

Causal relations constitute the largest proportion of the implicit discourse relations in the corpus, which suggests that they are more expected when no discourse marker is present, compared to many other relation types such as temporals, adversatives and additives. From a usage-based perspective, this provides partial support for the *causality-by-default* hypothesis put forth by Sanders (2005) in the sense that causal relations are identified even if no textual element explicitly marks them. However, in the absence of explicit sentence connectives other types of discourse relations could also be inferred, such as restatement or instantiation, which also account for a large proportion of unmarked relations in the PDTB corpus.

We also investigated implicit causality verbs which have been argued to act as local cues for an upcoming causal relationship. However, we found that their presence in the first sentence of a reason relationship does not increase the probability of the explicit connective to be dropped. This finding stands in contrast to what we predicted via an account of the UID hypothesis, which suggests that optional markers might be dropped if they contribute less information. Nevertheless, we observed that presence of an IC verb in the first argument of a sentence pair could generally signal a reason relation in the corpus, in line with the experimental finding of (Rohde and Horton, 2010) on discourse-level predictions. Taken together, our findings raise an interesting question for the future work: to what extent can global patterns vs. local cues account for the discourse relations being left implicit?

While the patterns we observed in the production data are compatible with the mentioned hypotheses about causality and continuity, they do not give us insight about the source of the tendency. Our results along with the related experimental findings can be considered from a frequency-based perspective, meaning that typical patterns in language production lead to expectations during comprehension about causality and continuity. Alternatively, it could be that people have an intrinsic tendency towards congruent and temporally ordered relations both in production and interpretation.

References

Caron, J., Micko, H. C., and Thuring, M. (1988). Conjunctions and the recall of composite sentences. *Journal of Memory and Language*, 27(3):309–323.

Couper-Kuhlen, E. and Kortmann, B. (2000). *Cause, Condition, Concession, Contrast: Cognitive and Discourse Perspectives*, volume 33. Walter de Gruyter.

Ferstl, E., Garnham, A., and Manouilidou, C. (2011). Implicit causality bias in english: a corpus of 300 verbs. *Behavior Research Methods*, 43(1):124–135.

Florian Jaeger, T. (2010). Redundancy and reduction: Speakers manage syntactic information density. *Cognitive Psychology*, 61(1):23–62.

Frank, A. and Jaeger, T. (2008). Speaking rationally: Uniform information density as an optimal strategy for language production. *Proceedings of the 28th meeting of the Cognitive Science Society*.

Genzel, D. and Charniak, E. (2002). Entropy rate constancy in text. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, pages 199–206.

Hobbs, J. (1990). *Literature and Cognition*, volume 21. Center for the Study of Language and Information (CSLI lecture notes).

Hume, D. (1784). *An Enquiry Concerning Human Understanding*. New York: The Liberal Arts Press, 1955 edition.

Irwin, J. (1980). The effects of explicitness and clause order on the comprehension of reversible causal relationships. *Reading Research Quarterly*, pages 477–488.

Kehler, A. (2000). Coherence and the resolution of ellipsis. *Linguistics and Philosophy*, 23(6):533–575.

Köhne, J. and Demberg, V. (2011). Incremental and predictive discourse based on causal and concessive discourse markers – a visual study. In *24th Annual CUNY Conference on Human Sentence Processing*.

König, E. (1991). Concessive relations as the dual of causal relations. *Semantic Universals and Universal Semantics. Groningen-Amsterdam Studies in Semantics*, 12:190–209.

Kuperberg, G., Paczynski, M., and Ditman, T. (2011). Establishing causal coherence across sentences: An ERP study. *Journal of Cognitive Neuroscience*, 23(5):1230–1246.

Langacker, R. (2000). A dynamic usage-based model. *Usage-based Models of Language*, pages 1–63.

Levinson, S. (2000). *Presumptive Meanings: The Theory of Generalized Conversational Implicature*. The MIT Press.

Levy, R. and Jaeger, T. F. (2007). Speakers optimize information density through syntactic reduction. In *Advances in Neural Information Processing Systems*.

Lin, Z., Kan, M., and Ng, H. (2009). Recognizing implicit discourse relations in the penn discourse treebank. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 343–351.

Millis, K., Golding, J., and Barker, G. (1995). Causal connectives increase inference generation. *Discourse Processes*, 20(1):29–49.

Murray, J. (1995). Logical connectives and local coherence. *Sources of Coherence in Reading*, pages 107–125.

Murray, J. (1997). Connectives and narrative text: The role of continuity. *Memory and Cognition*, 25(2):227–236.

Pitler, E., Raghupathy, M., Mehta, H., Nenkova, A., Lee, A., and Joshi, A. (2008). Easily identifiable discourse relations. *Technical Reports (CIS)*, page 884.

Prasad, R., Dinesh, N., Lee, A., Miltsakaki, E., Robaldo, L., Joshi, A., and Webber, B. (2008). The Penn Discourse Treebank 2.0. In *Proceedings of the 6th International Conference on Language Resources and Evaluation (LREC 2008)*, pages 2961–2968.

Rohde, H. and Horton, W. (2010). Why or what next? eye movements reveal expectations about discourse direction. In *Proceedings of 23rd Annual CUNY Conference on Human Sentence Processing*, pages 18–20.

Sagi, E. (2006). Context and the processing of discourse: Priming and genre effects on discourse comprehension. In *Proceedings of the 28th Annual Conference of the Cognitive Science Society*.

Sanders, T. (1997). Semantic and pragmatic sources of coherence: On the categorization of coherence relations in context. *Discourse Processes*, 24(1):119–147.

Sanders, T. (2005). Coherence, causality and cognitive complexity in discourse. In *Proceedings/Actes SEM-05, First International Symposium on the Exploration and Modelling of Meaning*, pages 105–114.

Segal, E., Duchan, J., and Scott, P. (1991). The role of interclausal connectives in narrative structuring: Evidence from adults' interpretations of simple stories. *Discourse Processes*, 14(1):27–54.

Simon, H. (1952). On the definition of the causal relation. *The Journal of Philosophy*, 49(16):517–528.

Sporleder, C. (2008). Lexical models to identify unmarked discourse relations: Does WordNet help? *Lexical-Semantic Resources in Automated Discourse Analysis*, page 20.

The PDTB Research Group (2008). The Penn Discourse Treebank 2.0 annotation manual. Technical report, Institute for Research in Cognitive Science, University of Pennsylvania.

Trabasso, T., Secco, T., and van den Broek, P. (1984). Causal cohesion and story coherence. *Learning and Comprehension of Text*, pages 83–111.

Traxler, M., Sanford, A., Aked, J., and Moxey, L. (1997). Processing causal and diagnostic statements in discourse. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, 23(1):88.

Zhou, Z., Lan, M., Niu, Z., Xu, Y., and Su, J. (2010). The effects of discourse connectives prediction on implicit discourse relation recognition. In *Proceedings of the 11th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 139–146.

Combining Statistical Translation Techniques for Cross-Language Information Retrieval

Ferhan Ture[1] Jimmy Lin[2,3] Douglas W. Oard[2,3]
(1) Department of Computer Science, University of Maryland, College Park
(2) College of Information Studies, University of Maryland, College Park
(3) UMIACS, University of Maryland, College Park
fture@cs.umd.edu, jimmylin@umd.edu, oard@umd.edu

ABSTRACT
Cross-language information retrieval today is dominated by techniques that rely principally on context-independent token-to-token mappings despite the fact that state-of-the-art statistical machine translation systems now have far richer translation models available in their internal representations. This paper explores combination-of-evidence techniques using three types of statistical translation models: context-independent token translation, token translation using phrase-dependent contexts, and token translation using sentence-dependent contexts. Context-independent translation is performed using statistically-aligned tokens in parallel text, phrase-dependent translation is performed using aligned statistical phrases, and sentence-dependent translation is performed using those same aligned phrases together with an *n*-gram language model. Experiments on retrieval of Arabic, Chinese, and French documents using English queries show that no one technique is optimal for all queries, but that statistically significant improvements in mean average precision over strong baselines can be achieved by combining translation evidence from all three techniques. The optimal combination is, however, found to be resource-dependent, indicating a need for future work on robust tuning to the characteristics of individual collections.

KEYWORDS: cross-language information retrieval, machine translation, context.

Proceedings of COLING 2012: Technical Papers, pages 2685–2702,
COLING 2012, Mumbai, December 2012.

1 Introduction

Cross-Language Information Retrieval (CLIR) is the problem of retrieving documents relevant to a query written in a different language. There are two main approaches to tackle this problem: translating the query into the document language, or translating documents into the query language. Query translation has become the more popular approach for experimental work due to the computational feasibility of trying different system variants without repeatedly translating the entire document collection (Oard, 1998; McCarley, 1999).

Query translation approaches for CLIR can be pursued either by applying a Machine Translation (MT) system or by using a token-to-token bilingual mapping, with or without translation probabilities. These approaches have complementary strengths: MT makes good use of context but at the cost of typically producing only one-best results, while token-to-token mappings can produce *n*-best token translations but without leveraging available contextual clues. This has led to a small cottage industry of what we might refer to as "context recovery" in which postprocessing techniques are used to select or reweight translation alternatives, usually based on evidence from term co-occurrence.

We argue that this false choice between MT and *n*-best token-by-token translation results from thinking of MT systems as black boxes. A modern statistical MT system has internally a series of increasingly rich representations that are exploited during the training and decoding processes. First, token alignments are generated for each training sentence pair, a process that also creates context-independent token-to-token translation probabilities. Second, these alignments are generalized to learn a Synchronous Context-Free Grammar (SCFG), in which probabilistic rules describe the translation of larger units of text. Finally, the translation grammar is combined with a language model to produce translations of entire sentences. As the whole process is statistically generated, it is at any point able to produce a ranked list of the highest scoring translations rather than only the one best choice. Although it is desirable to exploit these internal representations when performing retrieval, one possible disadvantage of using such a complex translation model is efficiency. However, modern decoders, e.g., `cdec` (Dyer et al., 2010), use pruning methods to efficiently search for the most likely translations of a given text.

In this paper, we describe two ways to exploit these internal representations and construct context-sensitive term translation probabilities. One method is to extract a context-aware portion of the SCFG by selecting only the grammar rules that apply to a given query. Using token alignments within each rule, a probability distribution can be constructed to represent the translation candidates for each query token, an approach that we refer to as "phrase-based." Another solution is to perform translation in context using the full MT system on the entire query and then to reconstruct context-sensitive token translation probabilities by accumulating translation likelihood evidence from each of the top *n* query translations.

These context-sensitive token translation probabilities can then be used in the same way as context-independent probabilities. In this work we use a technique based on mapping term statistics before computing term weights (Pirkola, 1998; Darwish and Oard, 2003), leading to a representation known as Probabilistic Structured Queries (PSQ). By doing this, we establish a strong context-independent "token-based" baseline that we can then compare directly with our proposed context-sensitive approaches.

Experiments on TREC 2002, NTCIR-8, and CLEF 2006 CLIR tasks, with topics in English and documents in Arabic, Chinese, and French, respectively, show that our approach consistently yields significant improvements over this baseline. The best results are achieved when we

perform a linear interpolation of all three approaches (query-based, phrase-based, and token-based). The remainder of this paper is organized as follows: Related work is described in Section 2, followed by our proposed approaches in Section 3, evaluation on the three collections in Section 4, and conclusions in Section 5. All of our code and test data are available as part of the open-source Ivory retrieval engine, available at `http://ivory.cc/`.

2 Background and Related work

Drawing inspiration from MT, Ponte and Croft (1998) introduced a monolingual information retrieval approach based on language models. This approach, which models the retrieval problem as if some noisy channel had corrupted some document into the query, was later extended by Berger and Lafferty (1999) and others. Combining language models with translation models to perform CLIR was a natural next step, and that approach yielded substantial improvements over earlier dictionary-based baselines, reporting Mean Average Precision (MAP) scores in the range of 90% of monolingual comparison conditions (Xu and Weischedel, 2005; Kraaij et al., 2003; Federico and Bertoldi, 2002). Nie (2010) summarizes this line of work well.

One limitation of applying language and translation models in CLIR is that they have mostly focused on isolated tokens (i.e., unigram models). To address this, there has been a substantial amount of work on exploiting query context in CLIR, dominated by approaches that use term co-occurrence statistics to select the most appropriate set of translation terms, based on some cohesion measure (Gao et al., 2006; Liu et al., 2005; Adriani and Rijsbergen, 2000; Seo et al., 2005). Expressing term dependency relations explicitly has been shown to produce good results in monolingual retrieval (Gao et al., 2004; Metzler and Croft, 2005), but extending that idea to CLIR has proven not to be as straightforward as one might expect. The closest approximation to be widely explored has been translation of multi-word expressions (so-called "phrase translation," although the "phrases" are often statistical rather than linguistic phenomena) in order to limit polysemy effects (Adriani and Rijsbergen, 2000; Arampatzis et al., 1998; Ballesteros and Croft, 1997; Chen et al., 2000; Meng et al., 2004; Zhang et al., 2007). Gao et al. (2012) recently introduced a query expansion approach also inspired by MT, modeling how query tokens are transformed into document tokens based on query context.

Inside an MT system we find a rich representation of alternative translations of the source "sentence" (which in our case is a query). MT-based CLIR approaches typically use one-best results since it has proven to be convenient to treat MT systems as black boxes (Magdy and Jones, 2011). One early CLIR system did try augmenting MT output using a bilingual dictionary (Kwok, 1999) in order to include alternative translations. More recently, Nikoulina et al. (2012) described techniques to maximize MAP when tuning an MT system and rerank the top n translations. Our approach focuses on combining different sources of evidence within an MT system, and can be considered complementary to these techniques. Combining the n-best derivations is also routinely used in speech retrieval (Olsson and Oard, 2009).

2.1 Context-independent Query Translation

As a baseline, we consider the technique presented by Darwish and Oard (2003). Given a source-language query $s = s_1, s_2, ...$, we represent s in the target language as a Probabilistic Structured Query (PSQ), where each token s_j is represented by its translations in the target language, weighted by the bilingual translation probability. These token-to-token translation probabilities are learned independently from a separate parallel bilingual text using automatic word alignment techniques, and we call this probability distribution Pr_{token}. In this approach,

the score of document d, given source-language query s, is computed by the following equations:

$$\text{Score}(d|s) = \sum_{j=1}^{\#\text{ terms}} \text{BM25}(\text{tf}(s_j, d), \text{df}(s_j)) \quad (1)$$

$$\text{tf}(s_j, d) = \sum_{\{t_i | Pr_{\text{token}}(t_i|s_j) > L\}} \text{tf}(t_i, d) Pr_{\text{token}}(t_i|s_j) \quad (2)$$

$$\text{df}(s_j) = \sum_{\{t_i | Pr_{\text{token}}(t_i|s_j) > L\}} \text{df}(t_i) Pr_{\text{token}}(t_i|s_j) \quad (3)$$

where L is a lower bound on translation probability. We also impose a cumulative probability threshold, C, so that translation alternatives of s_j are added (starting from the most probable ones) until the cumulative probability has reached C. As shown above, we use the Okapi BM25 term weighting function (with parameters $k_1 = 1.2$, $b = 0.75$), although in principle any other weighting function can be substituted.

Let us demonstrate this "token-based" representation model by an example. Following an Indri-like (Metzler and Croft, 2004) notation for query representations, the English query *Maternal leave in Europe* yields the following PSQ under this model for target language French:

```
#comb(#weight(0.74 matern, 0.26 maternel)
      #weight(0.49 laiss, 0.17 quitt, 0.09 cong, ...)
      #weight(0.91 europ, 0.09 européen))
```

Some of the translations are omitted due to space constraints. The `#comb` operator corresponds to the sum operator in equation (1), whereas the `#weight` operator is implemented as the weighted sum in equations (2) and (3). Within the `#weight` structure, terms follow their probabilities, which correspond to the Pr_{token} values in these equations. Notice that the translation distribution for the source token *leave* is uninformed by the context *maternity leave*, therefore the candidates *laisser* (Eng. let go, allow) and *quitter* (Eng. quit) have higher probabilities than *congé* (Eng. vacation, day off) in this model.

2.2 Machine Translation for Cross-Language IR

State-of-the-art statistical MT systems typically use hierarchical phrase-based translation models based on a Synchronous Context-Free Grammar (SCFG) (Chiang, 2005). In an SCFG, the rule `[X] || α || β || 𝒜 || ℓ(α → β)` indicates that the context free expansion $X \rightarrow \alpha$ in the source language occurs synchronously with $X \rightarrow \beta$ in the target language, with a likelihood of $\ell(\alpha \rightarrow \beta)$.[1] In this case, we call α the Left-Hand Side (LHS) of the rule, and β the Right-Hand Side (RHS) of the rule. We use indexed nonterminals (e.g., [X,1]) since in principle more than one nonterminal can appear on the right side. A sequence of token alignments $\mathcal{A}$ indicates which token in α is aligned to which target token in β.

Consider the following four rules from an SCFG:

R_1. `[S] || [S,1] || [S,1]`
R_2. `[S] || [X,1] || [X,1]`
R_3. `[X] || [X,1] leav in europ || cong de [X,1] en europ || 1-0 2-3 3-4 || 1`
R_4. `[X] || matern || matern || 0-0 || 0.69`

In the above notation, S refers to the sentence; therefore, the first two rules are special rules,

[1] The likelihood function ℓ is not a probability density function because it is not normalized.

describing that there is one sentential form, consisting of a single variable. In the third and fourth rules, we see the structure of the English phrase and how it is translated into French.

In contrast to the baseline model, this approach can handle both token and phrase translations. It can consider dependencies between query terms and therefore provide a more context-sensitive and appropriate translation of a given query. On the other hand, it is more dependent on training data and thus may not be as useful when the training set size is limited.

3 Context-Sensitive Query Translation

In this paper, we explore ways to improve the baseline token-translation model discussed above by exploiting the internal representations of the MT system. We describe two ways to construct a context-sensitive probability distribution for each query term, which can then be used directly by a similarly structured PSQ to retrieve ranked documents using equation (1). The first of these techniques (Section 3.1) was described in our previous paper and evaluated on a single collection (Ture et al., 2012); the second approach is new.

3.1 Probabilities from n-best Derivations

In MT, decoding is the process that finds the most probable translation with respect to an SCFG trained on a bilingual parallel corpus and a language model trained on monolingual target-language text. To control computational complexity, most decoders search for the most probable derivations by using pruning strategies.[2] The efficiency of our approaches is discussed in detail in Section 4.

When using one-best query translation, equations (1), (2) and (3) simplify to:

$$\text{Score}(d|s) = \sum_{i=1}^{m} \text{BM25}(\text{tf}(t_i^{(1)}, d), \text{df}(t_i^{(1)})) \tag{4}$$

where $t^{(1)}$ is the most probable translation of s, computed by:

$$\begin{aligned} t^{(1)} &= \arg\max_t [\max_{D \in \mathscr{D}(s,t)} \ell(t, D|s)] = \arg\max_t [\max_{D \in \mathscr{D}(s,t)} \text{TM}(t, D|s)\text{LM}(t)] \\ &= \arg\max_t \left[\text{LM}(t) \max_{D \in \mathscr{D}(s,t)} \prod_{r \in D} \ell(r)\right] \end{aligned} \tag{5}$$

where TM and LM correspond to the translation and language model scores, and $\mathscr{D}(s,t)$ is the set of possible derivations that generates the pair of sentences (s,t) (e.g., the sequence of four rules that translate the example query in Section 2.2 is one such derivation). The likelihood of each grammar rule r, $\ell(r)$, is learned as part of the training process of the translation model, by generalizing from token alignments on the training data (Chiang, 2007).

Decoders produce a set of candidate sentence translations in the process of computing equation (5), so we can generalize our model to consider the n candidates with the highest likelihoods, for some $n > 1$. We start by preprocessing the source query s and each candidate translation $t^{(k)}$. For each source token s_j, we use the derivation output to determine which grammar rules were used to produce $t^{(k)}$, and the token alignments in these rules to determine which target tokens are associated with s_j in the derivation. By doing this for each translation candidate $t^{(k)}$, we construct a probability distribution of possible translations of s_j based on the n query

[2] We use *derivation* to make it clear that what results is a rule sequence, not just a translated string.

translations. Specifically, if source token s_j is aligned to (i.e., translated as) t_i in the k^{th} best translation, the value $\ell(t^{(k)}|s)$ is added to its probability mass, producing the formula for Pr_{nbest}:

$$Pr_{\text{nbest}}(t_i|s_j) = \frac{1}{\varphi} \sum_{\substack{k=1 \\ s_j \text{ aligned to } t_i \text{ in } t^{(k)}}}^{n} \ell(t^{(k)}|s) \quad (6)$$

where φ is the normalization factor.[3] We should emphasize that Pr_{nbest} is a well-defined probability distribution for each s_j, so if a source token is translated consistently into the same target token in all n translations, then it will have a single translation with a probability of 1.0. Mapping *tf* and *df* statistics from source to target vocabulary is achieved by replacing Pr_{token} with Pr_{nbest} in equations (2) and (3).

Pr_{token} and Pr_{nbest} are similar in describing the probability of a target-language token given a source-language token, but differ by how the probability values are learned. For both approaches, we start from a large, potentially out-of-domain, sentence-aligned bilingual corpus. This corpus is first token-aligned using a word aligner. From these token alignments, one can directly deduce token translation probabilities, which correspond to Pr_{token}. In order to learn Pr_{nbest}, we add the MT system as an intermediate component, which creates a translation model from the token alignments, and then applies it (along with a language model) to the query text, using a decoder. Therefore, the distribution is informed by the query context and its derivation.

The advantage of Pr_{token} is the ability to model all of the translational varieties existent in the bilingual corpus, although these may be too noisy to properly translate a given query. For Pr_{nbest}, on the other hand, we would expect the distribution to be biased in favor of appropriate translations, but perhaps at the cost of some reduction in variety due to overfitting to the query context. For comparison, below are the translation probabilities for the same example query:

```
#comb(#weight(0.91 matern, 0.09 maternel, ...)
      #weight(1.0 cong) #weight(1.0 europ))
```

The overfitting issue is partially mitigated by using the n-best translation derivations, as opposed to the "1-best translation" approach, which treats the MT system as a black box. However, the lack of textual variety in the n most probable derivations is a known issue, caused by the fact that statistical MT systems identify the most probable derivations (not the most probable strings), many of which can correspond to the same surface form. This phenomenon is called "spurious ambiguity" in the MT literature, and it occurs in both phrase-based (Koehn et al., 2003) and hierarchical phrase-based MT systems (Chiang, 2007). For instance, according to Li et al. (2009), a string has an average of 115 distinct derivations in Chiang's `Hiero` system. Researchers have proposed several ways to cope with this situation, and we plan to integrate some of these in our future work. However, an alternative approach is to exploit grammar rules directly: this allows us to increase variety without introducing noisy translations, and we discuss this approach next.

3.2 Probabilities from the Translation Grammar

An alternative approach to exploit the MT system is to learn context-sensitive translation probabilities directly from the translation grammar. Hierarchical phrase-based MT systems use suffix arrays to extract all rules in an SCFG which apply to a given source text, requiring a

[3] Since a source token may be aligned to multiple target tokens in the same query translation, we still need to normalize the final likelihood values.

smaller memory footprint in the decoding phase (Lopez, 2007). We can use this feature to learn a token translation probability mapping that is a middle point between Pr_{token} and Pr_{nbest} in terms of context-aware choices and providing a varied set of translation alternatives.

We propose the following method to construct a probability distribution from a set of SCFG rules: For each grammar rule, we use the token alignments to determine which source token translates to which target token(s) in the phrase pair. Going over all grammar rules that apply to a given query, we construct a probability distribution for each token that appears on the LHS.

More specifically, given a translation grammar $\mathcal{G}$ and query s, we first use a suffix array extractor (Lopez, 2007) to obtain the subset of rules $\mathcal{G}(s)$ for which the source side pattern matches s. For each rule r in $\mathcal{G}(s)$, we identify each source token s_j on the LHS of r, ignoring any non-terminal symbols. From the token alignment information included in the rule structure, we can find all target tokens that s_j is aligned to. For each such target token t_i, the likelihood value of s_j being translated as t_i is increased by the likelihood score of r. If there are multiple target tokens, we increase the likelihood of each one equally. After repeating this process for all rules in the subset, we have constructed a list of possible translations and associated likelihood values for each source token that has appeared in any of the rules. We can then convert each list into a probability distribution, Pr_{phrase}, by normalizing the likelihood scores:

$$Pr_{\text{phrase}}(t_i|s_j) = \frac{1}{\psi} \sum_{\substack{r \in \mathcal{G}(s) \\ s_j \leftrightarrow t_i \text{ in } r}} \ell(r) \tag{7}$$

where ψ is the normalization factor and $s_j \leftrightarrow t_i$ represents an alignment between tokens s_j and t_i. Pr_{phrase} is different than Pr_{token} because it takes query context into account. Basically, we only look at the part of the grammar that applies to phrases in the source query, therefore create a bias in the probability distribution based on this context. It is also different than Pr_{nbest} because we do not perform any search, and there is no use of a language model. Thinking in terms of the MT pipeline, the representation we are exploiting in this approach is the extracted grammar, right before decoding has taken place, after token alignments have been generated. In order to illustrate the intuition behind this approach, the same example query is represented as follows using Pr_{phrase}:

```
#comb(#weight(0.68 matern, 0.06 maternel, ...)
      #weight(0.35 cong, 0.24 laiss, 0.13 quitt, ...)
      #weight(0.90 europ, 0.07 européen, ...))
```

When compared to Pr_{token}, notice that the translation distribution of *leave* shifts towards the more appropriate translation *congé* as a result of this approach.

3.3 Combining Sources of Evidence

All three approaches for query translation (i.e., token-based, phrase-based, and query-based) have complementary strengths, so we introduce a unified CLIR model by performing a linear interpolation of the three probability distributions:

$$Pr_{\text{c}}(t_i|s_j;\lambda_1,\lambda_2) = \lambda_1 Pr_{\text{nbest}}(t_i|s_j) + \lambda_2 Pr_{\text{phrase}}(t_i|s_j) + (1-\lambda_1-\lambda_2) Pr_{\text{token}}(t_i|s_j) \tag{8}$$

Replacing Pr_{token} with Pr_{c} in equation (1) gives us the document scoring formula for the combined model (call Score_{c}).

Until now, we focused on the translation of single-token terms, but we can also use the n-best derivation list to identify how multi-token "phrases" are translated in context. The right hand side of the rules in the n most probable derivations provides us with statistically meaningful target-language phrases, along with their associated probabilities (described by Pr_{multi} below). With this addition, we score each document by a weighted average of the single-token approach (i.e., Score_c) and the sum of document scores for the multi-token terms:

$$\text{Score}(d|s;\gamma) = \gamma \text{Score}_{\text{c}}(d|s;\lambda_1,\lambda_2) + (1-\gamma) \sum_{\text{phrase } p} \text{BM25}(\text{tf}(p,d), \text{df}(p)) Pr_{\text{multi}}(p) \quad (9)$$

$$Pr_{\text{multi}}(p) = \frac{1}{\psi} \sum_{k=1}^{n} \sum_{\substack{\text{rule } r \in D^{(k)} \\ p \in RHS(r)}} \ell(r) \quad (10)$$

where ψ is the normalization factor and $D^{(k)}$ is the derivation of the k^{th} best translation. Below is the representation of the example query under this model, with γ set to 0.8:

```
#combweight(0.8 #comb(#weight(0.81 matern, 0.12 maternel, ...)
                      #weight(0.45 cong, 0.25 laiss, 0.10 quitt, ...)
                      #weight(0.95 europ, 0.04 européen, ...))
            0.1 ''en europ'', 0.08 ''cong de'', 0.01 ''cong matern'', ...)
```

The `#comb` structure represents Pr_{c} and the remaining multi-token terms represent Pr_{multi}, all extracted from the top n derivations. The `#combweight` operator corresponds to equation (9).

4 Evaluation

We evaluated our system on the latest available CLIR test collections for three languages: TREC 2002 English-Arabic CLIR, NTCIR-8 English-Chinese Advanced Cross-Lingual Information Access (ACLIA), and CLEF 2006 English-French CLIR. For the Arabic and French collections, we used title queries because they are most representative of the short queries that searchers frequently pose to web search engines. Chinese queries in the NTCIR-8 ACLIA test collection are in the form of complete syntactically correct questions, but for consistency we treated them as bag-of-words queries in our experiments with no special processing. The collections contain 383,872, 388,589 and 177,452 documents, and 50, 50, and 73 topics, respectively.

We learned our English-to-Arabic translation model using 3.4 million aligned sentence pairs from the GALE 2010 evaluation. Our English-to-Chinese translation model was trained on 302,996 aligned sentence pairs from the FBIS parallel text collection. We trained an English-to-French translation model using 2.2 million aligned sentence pairs from the latest Europarl corpus (version 7) that was built from the European parliament proceedings.[4]

Token alignments were learned with GIZA++ (Och and Ney, 2003), using 5 Model 1 and 5 HMM iterations. An SCFG serves as the basis for the translation model (Chiang, 2007), which was extracted from these token alignments using a suffix array (Lopez, 2007). We used `cdec` for decoding, due to its support for SCFG-based models and its efficient C-based implementation, making it faster than most of the other state-of-the-art systems (Dyer et al., 2010). A 3-gram language model was trained from the target side of the training data for Chinese and Arabic, using the SRILM toolkit (Stolcke, 2002). For French, we trained a 5-gram LM from the monolingual dataset provided for WMT-12. The Chinese collection was segmented using the Stanford segmenter (Tseng et al., 2005), English topics and the French collection

[4] http://www.statmt.org/europarl

were tokenized using the OpenNLP tokenizer,[5] and Arabic was tokenized and stemmed using the Lucene package.[6] For English and French, we also lowercased text, stemmed using the Snowball stemmer, and removed stopwords.

4.1 Effectiveness

We used Mean Average Precision (MAP) as the evaluation metric. The baseline token-based model yields a MAP of 0.2712 for Arabic, 0.1507 for Chinese, and 0.2617 for French. Direct comparisons to results reported at TREC, NTCIR, and CLEF (respectively) are hard to make because of differences in experimental conditions, but the comparisons we are able to make suggest that these baseline MAP values are reasonable.[7] For Arabic, the best reported results from TREC-2002 were close to 0.40 MAP (Fraser et al., 2002), but those results were achieved by performing query expansion and learning stem-to-stem mappings; our experiment design requires token-to-token mappings (which result in sparser alignments). For Chinese, the NTCIR-8 topics are in the form of questions, and systems that applied question rewriting performed better than those that did not. Also, 15 of the questions are about people, for which our vocabulary coverage was not tuned. If we disregard these 15 topics, our baseline system achieves 0.1778, close to the best reported results with comparable settings, with a MAP of 0.181 (Zhou and Wade, 2010). For French, our baseline achieves close to the same score as the one reported result at CLEF-2006 that did not incorporate blind relevance feedback (0.2606 MAP) (Savoy and Abdou, 2006).

As discussed in Section 3, we implemented three techniques to construct a term translation probability distribution: Pr_{nbest}, Pr_{phrase} and Pr_{token}, described by equations (6), (7) and (1) above.[8] We assessed these three approaches by (i) comparing them against each other, and (ii) measuring the benefit of a linear combination, i.e., Pr_{c}, described by equation (8).

Experiment results are summarized in Table 1 and illustrated in Figure 1. In that figure, we provide three connected scatterplots of MAP scores within a range of values for λ_1 and λ_2. In order to see the effectiveness of the interpolated model with respect to parameters λ_1 and λ_2, we performed a grid search by applying values in increments of 0.1 (ranging from 0 to 1) to the interpolated model Pr_{c}. For readability, figures only include a representative subset of λ_2 settings, where different lines represent different values for λ_2. To distinguish the extreme settings of $\lambda_2 = 0$ and $\lambda_2 = 1$, we use a filled circle or square, respectively.

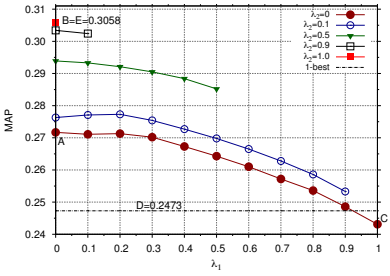
The left edge represents $\lambda_1 = 0$, meaning that we do not use probabilities learned from the n-best derivations (i.e., Pr_{nbest}) in our interpolation. Along the y-axis on the left edge, we see results for various settings of λ_2, which controls how much weight is put on Pr_{phrase} and Pr_{token}. Within these settings, a particularly interesting one is when λ_2 is set to 0. In this case, the approach is solely based on context-independent translation probabilities (i.e., Pr_{token}), which is the baseline model (call this condition A). When λ_2 is set to 1, we rely on phrase-based term translation probabilities (i.e., Pr_{phrase}, call this condition B). By contrast, at the right edge, $\lambda_1 = 1$, so we rely only on Pr_{nbest} when translating query terms (call this condition C). For

[5] http://opennlp.apache.org

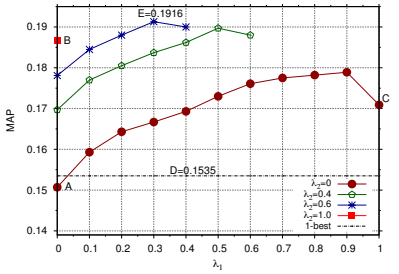
[6] http://lucene.apache.org

[7] The best results often employ blind relevance feedback, multiple lexical resources and/or very long queries. While these techniques can be useful in deployed applications, we have chosen not to run such conditions in order to avoid masking the effects that we wish to study.

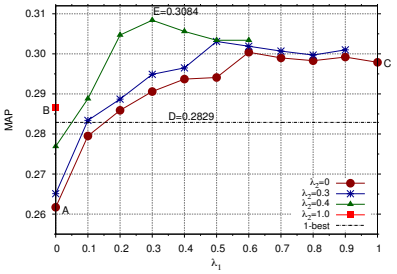
[8] We fixed $C = 0.95, L = 0.005, n = 10$ for all models after manually trying a range of values.



(a) TREC 2002 English-Arabic CLIR task



(b) NTCIR-8 English-Chinese CLIR task



(c) CLEF 2006 English-French CLIR task

Figure 1: Effectiveness results.

reference, the dotted horizontal line represents simply taking the one-best translation from the MT system (i.e., described by equation (4), call this condition D).

In the case of the Arabic collection, we observe a strictly decreasing trend for the MAP scores as λ_2 decreases, and the best results are obtained when λ_1 is 0 and λ_2 is 1.0 (call the condition with the best MAP score E). In other words, the interpolation yields a maximum 0.3058 MAP when it was based entirely on Pr_{phrase}, ignoring distributions Pr_{token} and Pr_{nbest}. For the Chinese collection, λ_1=0.2 and λ_2=0.7 yields the best result (MAP=0.1916), whereas effectiveness peaks at λ_1=0.3 and λ_2=0.4 for the French collection, with a MAP score of 0.3084.

Based on the randomized significance test proposed by Smucker et al. (2007), the combined approach (E) outperforms all models (except for the phrase-based approach) in the Arabic collection with 95% confidence. When we ran the same test on the other two collections, we found that the combined approach is significantly better than the baseline (A) and 1-best (D) approaches for Chinese, whereas MAP is significantly higher than baseline A for French. These results confirm that the complementary advantages of each model can be combined into a single superior model using our approach.

We also experimented with the multi-token term representation in equation (9), by varying the γ parameter. With γ set empirically to 0.8, the MAP increased by 0.005 for French, and remained about the same for Arabic and Chinese.

When the three individual models (conditions A, B and C) are compared (i.e., ignoring the

Condition	MAP		
	Arabic	Chinese	French
A: λ_1=0, λ_2=0 (Pr_{token})	0.2712	0.1507	0.2617
B: λ_1=0, λ_2=1 (Pr_{phrase})	0.3058	0.1867	0.2868
C: λ_1=1, λ_2=0 (Pr_{nbest})	0.2431	0.1709	0.2979
D: 1-best	0.2473	0.1535	0.2829
E: best $\{\lambda_1, \lambda_2\}$	$0.3058^{a,c,d}$	$0.1916^{a,d}$	0.3084^{a}

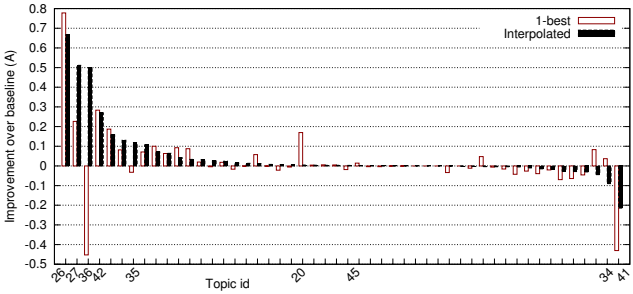
Table 1: A summary of experimental results under different conditions, for all three CLIR tasks. Superscripts indicate if the best result is significantly better than conditions A, B, C, and D.

interpolated results), the phrase-based model (B) is significantly better than the token-based baseline (A) for Arabic and Chinese, but statistically indistinguishable from the same baseline model in the case of French. For French, the best retrieval effectiveness results from the *n*-best full query translation model (C), significantly better than the baseline model (A). This shows that there is no individual model that outperforms the rest in all three collections. The real strength of our approach is therefore to introduce a unified probabilistic model that can combine all of these different approaches in a principled manner.

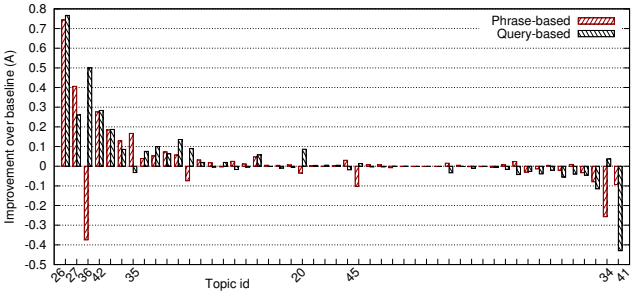
For topic-specific analysis, we looked at the distribution of the average precision (AP) differences between the various models. We observe that our best interpolated model (E) yields better AP than the token-based baseline model (A) for 36 of the 43 Arabic topics in which there was a noticeable difference (7 of the 50 Arabic topics exhibited differences of 0.001 or less). For the Chinese collection, the same was true for 41 of 57 topics (with 16 exhibiting a negligible difference), whereas the comparable statistic is 29 of 44 for French.

For space reasons, we illustrate these differences only for French. Figure 2(a) plots the AP improvement of the best interpolated model (E) and the one-best MT approach (D) over (or the average degradation below) the token-based baseline (A), sorted left to right by decreasing AP improvement for the interpolated model (E); Figure 2(b) similarly plots the same AP differences for the phrase-based (B) and *n*-best full query translation approaches (C), again with reference to the token-based baseline (A), with topics sorted in the same order to facilitate comparison. These plots make it quite clear that the three approaches vary in their per-topic effectiveness. Rather than slight variations in all of the topics, we see several cases in which one of the models is superior to the others. For instance, the *n*-best full query translation approach is a clear winner for topics 26, 36 and 42, whereas the phrase-based approach outperforms for topics 27 and 35. Despite its drawbacks, there are topics in which the token-based model is superior to our more sophisticated approaches, notably topics 34 and 41. Once again, this analysis supports our argument that combining these three probabilistic models into one unified approach can capture some of the best of each. In general, we expect the interpolated model to be more robust, since it has access to more evidence than the individual models.

In practice, we would like to select model parameters without observing all the test topics. Therefore, we ran 10-fold cross-validation experiments on each collection, selecting parameters that maximize MAP on nine folds and evaluating on the remaining one. This method yields a MAP of 0.2979 for Arabic, 0.1733 for Chinese, and 0.2872 for French, all significantly better than the token-based baseline (A). We also explored if we could use two of the collections to tune parameters for the third. For this, we first ranked each (λ_1, λ_2) pair by MAP on each collection. In order to select the parameters for a particular collection, we added the ranks



(a) Interpolated and 1-best models vs. the token-based baseline approach.



(b) Phrase- and query-based (i.e., n-best) models vs. the token-based baseline approach.

Figure 2: Per-topic AP improvement over token-based baseline (condition A) for French.

from the other two collections and picked the one with the lowest sum. Using this method, the selected parameters were $(0.1, 0.1)$ for Arabic, $(0.3, 0.5)$ for Chinese, and $(0.1, 0.1)$ for French. When compared to the token-based baseline (A), this approach showed significant improvements only for Chinese. We conclude from this analysis that the optimal combination of models depends on the collection, language, and resources. Once these are fixed, we can use a subset of the topics to appropriately tune parameters for the rest. However, better tuning methods need to be devised for a truly robust approach to combining these CLIR models.

4.2 Efficiency

We compared the various CLIR approaches in terms of efficiency (query evaluation time), performing experiments on a machine running Red Hat Linux on a 2.4 GHz processor. We processed the Arabic topics using each model and measured running time per query in milliseconds. Averages over three repeated runs are reported in Table 2 (with 95% confidence intervals).

As described before, there are three processes in the MT pipeline: token alignment, grammar extraction, and decoding. Token alignment is query-independent and required for all three approaches, so we did not include it in our running time comparison of running times. For the construction of Pr_{phrase}, we only need to extract grammar rules that apply to each given query,

	Process	Pr_{token}	Pr_{phrase}	Pr_{nbest}			Pr_{c}
				1-best	**5-best**	**10-best**	
MT	Grammar extraction	-	7.57				
	Decoding	-	-	134.94			134.94
IR	Initialization	negligible	64.38	negligible			64.38
	Generation	48.12	negligible	5.80	59.47	62.25	49.11
	Ranking	545.64	514.17	97.64	158.81	179.07	601.95
Total time (in *ms*)		594±22	586±13	246±15	361±28	383±22	858±20

Table 2: Average running times for processes in the CLIR pipeline (in *ms*).

whereas Pr_{nbest} also requires decoding.[9]

The remaining processes that we need to consider are part of the IR pipeline: initialization of the CLIR model, generation of query representations in the target language, and ranking of the most relevant documents in the collection. We only count query-dependent initialization costs, since other costs such as loading the bilingual dictionary need to be done only once, even with many queries. The input of the generation step is the source-language query, and the output is a PSQ that represents that query in the target language. In the phrase-based method, this step takes a negligible amount of time, because the probability distribution is already in memory at the beginning of this step, and it is very small (i.e., probabilities for a few query terms only). For Pr_{nbest}, generation time rises linearly as n is increased.

Ranking time depends on the complexity of the query representation. With more complex representations, it is possible to increase effectiveness, but at the cost of efficiency. Therefore, a desirable CLIR approach would express all the relevant information and nothing more. The distributions Pr_{token} and Pr_{phrase} tend to include more translation alternatives per query term, resulting in a more complex representation and longer ranking time. As a result, interpolating all three distributions generates a complex representation as well.

When we look at the total running times in Table 2, we observe that the n-best approach is significantly more efficient than the token-based baseline, even though it requires additional MT processes to fully translate the queries. When $n = 1$, the reduction in total running time is nearly 60%. The savings become more modest as n increases, approximately 39% and 35% for 5-best and 10-best MT approaches. Increasing n also improves effectiveness, thus there is a tradeoff to consider when deciding on the value for n. There is a similar tradeoff for the token-based approach: the representation can be simplified if more aggressive thresholding is used, e.g., if C increased in equation (1); however, this may result in a less effective model.

We do not see the same efficiency improvements from reduction in query complexity with the phrase-based model; the query complexity is similar to the baseline approach. As a result, the phrase-based approach runs in about the same total time. However, the MAP score improves considerably for all of the collections, so we can say that Pr_{phrase} is superior to Pr_{token}.

The combined model Pr_{c} yields the highest MAP scores but also takes the longest time to complete. When compared to the baseline model, running time increases by 44%, which seems acceptable given the consistently significant improvements. We should note that our implementation is not fully optimized, and is open to further improvements in the future.

As a summary of our evaluation, we believe that the best choice depends on user expectations. For a faster and possibly more effective model, Pr_{nbest} and Pr_{phrase} seem to be good alternatives

[9] It is reasonable to assume that the decoder time to find top n translations is the same as finding the one-best result.

to Pr_{token}. For best effectiveness, the interpolation of the three probability distributions is a good choice, providing significantly better results at the cost of additional complexity.

5 Conclusions and Future Work

In this paper, we introduced a theoretical framework that uses a statistical translation model for cross-language information retrieval. Our approach combines the representational advantage of probabilistic structured queries with the richness of the intermediate information produced by translation models. We proposed two ways of exploiting the internal representation of translation models to learn context-sensitive term translation probabilities: (1) aggregate information from the *n*-best translation outputs by an MT decoder, or (2) extract the subset of the translation grammar that applies to a given query, and use the token alignments in each rule to construct a probability distribution. Although using translation models for CLIR is not a novel approach, we have introduced novel ideas on *how* one can utilize the rich internal representation of MT systems for this task.

We evaluated our models on an English-Arabic task from TREC 2002, an English-Chinese task from NTCIR-8, and an English-French task from CLEF 2006, finding in all three cases that an optimal linear combination of the three approaches can significantly improve MAP, but that the optimal parameters vary by collection. We also compared approaches in terms of efficiency and showed that our framework provides a set of choices, allowing a beneficial tradeoff between improving efficiency and effectiveness. Because we used only one collection per language, experiments with multiple collections for the same language will be needed before we can begin to speculate on whether these differences are language-dependent, collection-dependent, or some combination of the two. Additionally, we would like to try this approach on more languages to further study the consistency in improvements, and also with different parallel corpora and monolingual language modeling collections, in order to tease out whether the differences we are seeing in the optimal combination weights are resource dependent (varying principally with different parallel corpora and/or language models).

In terms of modeling, we plan to revisit the rather ad hoc way we have incorporated multi-word expressions, exploring ways of leveraging them in each model separately rather than at the final evidence combination stage. Also, since the benefit of performing full machine translation would be expected to increase as available context increases, we would like to explore the potential for translating documents in addition to queries. Following the same methods described in this paper, we could learn a new set of probability distributions from the document translations, which could be combined with the current three approaches to construct an even richer and possibly more accurate CLIR model. We also plan to explore the effect of using a phrase-based MT system as an alternative to the SCFG-based model in our experiments.

In conclusion, we have introduced ways of using statistical translation models for CLIR that take greater advantage of the capabilities of current statistical MT systems, and we hope that the promising results we have reported will spur the community to further explore this space.

Acknowledgments

This research was supported in part by the BOLT program of the Defense Advanced Research Projects Agency, Contract No. HR0011-12-C-0015; NSF under awards IIS-0916043 and IIS-1144034. Any opinions, findings, conclusions, or recommendations expressed in this paper are those of the authors and do not necessarily reflect views of the sponsors. The second author is grateful to Esther and Kiri for their loving support and dedicates this work to Joshua and Jacob.

References

Adriani, M. and Rijsbergen, C. J. V. (2000). Phrase identification in cross-language information retrieval. In *Proceedings of RIAO 2000*.

Arampatzis, A. T., Tsoris, T., Koster, C. H. A., and Weide, P. V. D. (1998). Phrase-based information retrieval. *Information Processing & Management*, 34(6):693–707.

Ballesteros, L. and Croft, W. B. (1997). Phrasal translation and query expansion techniques for cross-language information retrieval. *SIGIR Forum*, 31:84–91.

Berger, A. and Lafferty, J. (1999). Information retrieval as statistical translation. In *Proceedings of SIGIR 1999*, pages 222–229.

Chen, A., Jiang, H., and Gey, F. (2000). English-Chinese cross-language IR using bilingual dictionaries. In *Proceedings of TREC-9*.

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

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

Darwish, K. and Oard, D. W. (2003). Probabilistic structured query methods. In *Proceedings of SIGIR 2003*, pages 338–344.

Dyer, C., Weese, J., Setiawan, H., Lopez, A., Ture, F., Eidelman, V., Ganitkevitch, J., Blunsom, P., and Resnik, P. (2010). cdec: a decoder, alignment, and learning framework for finite-state and context-free translation models. In *Proceedings of the ACL 2010 System Demonstrations*, pages 7–12.

Federico, M. and Bertoldi, N. (2002). Statistical cross-language information retrieval using n-best query translations. In *Proceedings of SIGIR 2002*, pages 167–174.

Fraser, A., Xu, J., and Weischedel, R. (2002). TREC 2002 cross-lingual retrieval at BBN. In *Proceedings of TREC-11*.

Gao, J., Nie, J.-Y., Wu, G., and Cao, G. (2004). Dependence language model for information retrieval. In *Proceedings of SIGIR 2004*, pages 170–177.

Gao, J., Nie, J.-Y., and Zhou, M. (2006). Statistical query translation models for cross-language information retrieval. *ACM Transactions on Asian Language Information Processing (TALIP)*, 5:323–359.

Gao, J., Xie, S., He, X., and Ali, A. (2012). Learning lexicon models from search logs for query expansion. In *Proceedings of EMNLP 2012*, pages 666–676.

Koehn, P., Och, F. J., and Marcu, D. (2003). Statistical phrase-based translation. In *Proceedings of NAACL-HLT 2003*, pages 48–54.

Kraaij, W., Nie, J., and Simard, M. (2003). Embedding web-based statistical translation models in cross-language information retrieval. *Computational Linguistics*, 29:381–419.

Kwok, K. L. (1999). English-Chinese cross-language retrieval based on a translation package. In *Workshop of Machine Translation for Cross Language Information Retrieval, Machine Translation Summit VII*, pages 8–13.

Li, Z., Eisner, J., and Khudanpur, S. (2009). Variational decoding for statistical machine translation. In *Proceedings of ACL 2009*, pages 593–601.

Liu, Y., Jin, R., and Chai, J. Y. (2005). A maximum coherence model for dictionary-based cross-language information retrieval. In *Proceedings of SIGIR 2005*, pages 536–543.

Lopez, A. (2007). Hierarchical phrase-based translation with suffix arrays. In *Proceedings of EMNLP-CoNLL 2007*, pages 976–985.

Magdy, W. and Jones, G. J. F. (2011). Should MT systems be used as black boxes in CLIR? In *Proceedings of ECIR 2011*, pages 683–686.

McCarley, J. S. (1999). Should we translate the documents or the queries in cross-language information retrieval? In *Proceedings of ACL 1999*, pages 208–214.

Meng, H. M., Chen, B., Khudanpur, S., Levow, G.-A., Lo, W. K., Oard, D. W., Schone, P., Tang, K., Wang, H.-M., and Wang, J. (2004). Mandarin-English information (MEI): Investigating translingual speech retrieval. *Computer Speech & Language*, 18(2):163–179.

Metzler, D. and Croft, W. B. (2004). Combining the language model and inference network approaches to retrieval. *Information Processing & Management*, 40:735–750.

Metzler, D. and Croft, W. B. (2005). A Markov random field model for term dependencies. In *Proceedings of SIGIR 2005*, pages 472–479.

Nie, J.-Y. (2010). *Cross-Language Information Retrieval*. Morgan & Claypool Publishers.

Nikoulina, V., Kovachev, B., Lagos, N., and Monz, C. (2012). Adaptation of statistical machine translation model for cross-language information retrieval in a service context. In *Proceedings of EACL 2012*.

Oard, D. W. (1998). A comparative study of query and document translation for cross-language information retrieval. In *Proceedings of AMTA 1998*, pages 472–483.

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

Olsson, J. S. and Oard, D. W. (2009). Combining LVCSR and vocabulary-independent ranked utterance retrieval for robust speech search. In *Proceedings of SIGIR 2009*, pages 91–98.

Pirkola, A. (1998). The effects of query structure and dictionary-setups in dictionary-based cross-language information retrieval. In *Proceedings of SIGIR 1998*, pages 55–63.

Ponte, J. M. and Croft, W. B. (1998). A language modeling approach to information retrieval. In *Proceedings of SIGIR 1998*, pages 275–281.

Savoy, J. and Abdou, S. (2006). Experiments with monolingual, bilingual, and robust retrieval. In *Proceedings of CLEF 2006*, pages 137–144.

Seo, H.-C., Kim, S.-B., Rim, H.-C., and Myaeng, S.-H. (2005). Improving query translation in English-Korean cross-language information retrieval. *Information Processing & Management*, 41(3):507–522.

Smucker, M. D., Allan, J., and Carterette, B. (2007). A comparison of statistical significance tests for information retrieval evaluation. In *Proceedings of CIKM 2007*, pages 623–632.

Stolcke, A. (2002). SRILM - an extensible language modeling toolkit. In *Proceedings of ICSLP 2002*, pages 901–904.

Tseng, H., Chang, P.-C., Andrew, G., Jurafsky, D., and Manning, C. (2005). A conditional random field word segmenter. In *Proceedings of the Fourth SIGHAN Workshop on Chinese Language Processing*.

Ture, F., Lin, J., and Oard, D. W. (2012). Looking inside the box: context-sensitive translation for cross-language information retrieval. In *Proceedings of SIGIR 2012*, pages 1105–1106.

Xu, J. and Weischedel, R. (2005). Empirical studies on the impact of lexical resources on CLIR performance. *Information Processing & Management*, 41:475–487.

Zhang, W., Liu, S., Yu, C., Sun, C., Liu, F., and Meng, W. (2007). Recognition and classification of noun phrases in queries for effective retrieval. In *Proceedings of CIKM 2007*, pages 711–720.

Zhou, D. and Wade, V. (2010). The effectiveness of results re-ranking and query expansion in cross-language information retrieval. In *Proceedings of NTCIR-8 Workshop Meeting*.

Multi-way Tensor Factorization for Unsupervised Lexical Acquisition

Tim Van de Cruys[1,3] Laura Rimell[1,2]
Thierry Poibeau[1,4] Anna Korhonen[1,2]
(1) DTAL, University of Cambridge, UK
(2) NLIP, University of Cambridge, UK
(3) IRIT, UMR 5505, CNRS, Toulouse, France
(4) LaTTiCe, UMR 8094, CNRS & ENS, Paris, France
tim.vandecruys@irit.fr, laura.rimell@cl.cam.ac.uk,
thierry.poibeau@ens.fr, anna.korhonen@cl.cam.ac.uk

ABSTRACT

This paper introduces a novel method for joint unsupervised aquisition of verb subcategorization frame (SCF) and selectional preference (SP) information. Treating SCF and SP induction as a multi-way co-occurrence problem, we use multi-way tensor factorization to cluster frequent verbs from a large corpus according to their syntactic and semantic behaviour. The method extends previous tensor factorization approaches by predicting whether a syntactic argument is likely to occur with a verb lemma (SCF) as well as which lexical items are likely to occur in the argument slot (SP), and integrates a variety of lexical and syntactic features, including co-occurrence information on grammatical relations not explicitly represented in the SCFS. The SCF lexicon that emerges from the clusters achieves an F-score of 68.7 against a gold standard, while the SP model achieves an accuracy of 77.8 in a novel evaluation that considers all of a verb's arguments simultaneously.

TITLE AND ABSTRACT IN FRENCH

Factorisation de tenseurs à plusieurs dimensions pour l'acquisition lexicale non supervisée

Cet article présente une méthode originale pour l'acquisition simultanée de cadres de sous-catégorisation (*subcategorization frames*) et de restrictions de sélection (*selectional preferences*) appliquée au lexique verbal. L'induction simultanée de ces deux types d'information est vue comme un problème de cooccurrence à plusieurs dimensions. On introduit donc une méthode de factorisation de tenseurs, afin de classer les verbes fréquents d'un grand corpus suivant leur comportement syntaxique. L'approche est fondée sur un ensemble de traits de nature syntaxique et lexicale, y compris des informations de cooccurrence au sein des relations grammaticales qui ne sont pas explicitement représentées dans les schémas de sous-catégorisation. Le dictionnaire de sous-catégorisation produit par la méthode de classification obtient une F-mesure de 68,7 lors de l'évaluation face à un dictionnaire de référence tandis que les restrictions de sélection ont une exactitude (*accuracy*) de 77,8 en tenant compte de tous les arguments simultanément.

KEYWORDS: subcategorization frames, selectional preferences, lexical acquisition, tensor factorization, unsupervised machine learning.

KEYWORDS IN FRENCH: cadre de sous-catégorisation, restriction de sélection, acquisition lexicale, factorisation de tenseurs, apprentissage non supervisé.

Proceedings of COLING 2012: Technical Papers, pages 2703–2720,
COLING 2012, Mumbai, December 2012.

1 Introduction

Verb subcategorization lexicons and selectional preference models capture two related aspects of verbal predicate-argument structure, with subcategorization describing the syntactic arguments taken by a verb, and selectional preferences describing the semantic preferences verbs have for their arguments. Each type of information can support NLP tasks requiring information about predicate-argument structure. For example, subcategorization has proved useful for parsing (Carroll and Fang, 2004; Arun and Keller, 2005; Cholakov and van Noord, 2010), semantic role labeling (Bharati et al., 2005; Moschitti and Basili, 2005), verb clustering, (Schulte im Walde, 2006; Sun and Korhonen, 2011) and machine translation (Han et al., 2000; Hajič et al., 2002), while selectional preferences have benefited parsing (Zhou et al., 2011), semantic role labeling (Gildea and Jurafsky, 2002; Zapirain et al., 2009), and word sense disambiguation (Resnik, 1997; Thater et al., 2010; Seaghdha and Korhonen, 2011).

Verb subcategorization frame (SCF) induction involves identifying the arguments of a verb lemma in a corpus, and generalizing about the *frames* taken by the verb, where each frame includes a number of arguments and their syntactic types. Consider e.g. sentence (1), where the verb *show* takes the frame SUBJ-DOBJ-CCOMP (subject, direct object, and clausal complement).

(1) $[\text{Our October review}]_{\text{SUBJ}}$ comprehensively $[\text{shows}]_{\text{VERB}}$ $[\text{you}]_{\text{DOBJ}}$ $[\text{what's in store in next month's magazine}]_{\text{CCOMP}}$.

Predicting the set of SCFs for a verb can be viewed as a multi-way co-occurrence problem of a verb and its different arguments. One of the main challenges is distinguishing arguments from adjuncts (e.g. temporal, locative, or manner modifiers). Most SCF induction work to date considers only the co-occurrences of verb lemmas with different grammatical relation types (subject, object, prepositional phrase, etc.). Taking SCF acquisition to the next level requires consideration of the lexical fillers of potential argument slots for more accurate argument-adjunct discrimination.

Selectional preference (SP) induction involves predicting the likelihood of a given lexical item occurring in an argument slot, and generalizing about the lexical classes which occur in the slot, which may be dependent on the SCF. In sentence (2), for example, the verb *show* takes the frame SUBJ-DOBJ, and the direct object of *show* in this frame is likely to be inanimate.

(2) $[\text{Stalin}]_{\text{SUBJ}}$, who must have been well informed through his network of spies, $[\text{showed}]_{\text{VERB}}$ $[\text{no emotion}]_{\text{DOBJ}}$.

Most SP induction work to date has focused on discovering lexical preferences within the direct object slot alone, or at most three-way co-occurrences between verb, subject, and direct object, and has not considered the full range of potential argument slots for which verbs subcategorize, thus losing some of the contextual information which may be helpful in learning SPs. Moreover, the ability of SP acquisition methods to model the full range of verbal arguments, including e.g. clausal complements, has not been tested.

As the two types of lexical information – SCFs and SPs – are closely interlinked and can complement each other, it would make sense to acquire them jointly. However, to the best of our knowledge, no previous work has developed a model for their joint acquisition.

Unsupervised machine learning is attractive for lexical acquisition because it works where little labeled data is available, and ports easily between tasks and languages. Increasingly sophisticated techniques have been applied to SP induction (Rooth et al., 1999; Van de Cruys, 2009; Ó Séaghdha, 2010; Ritter and Etzioni, 2010; Reisinger and Mooney, 2011) while work

on unsupervised SCF acquisition has been limited (Carroll and Rooth, 1996). In this paper we present a largely unsupervised method for the joint acquisition of SCFs and SPs, adapting a method that has been successfully used for SP induction (Van de Cruys, 2009) so that it learns *whether* a verb subcategorizes for a particular argument slot together with *which* lexical items occur in the slot.

Our method uses a co-occurrence model augmented with a factorization algorithm to cluster verbs from a large corpus. Specifically, we use non-negative tensor factorization (NTF) (Shashua and Hazan, 2005), a generalization of matrix factorization that enables us to capture latent structure from multi-way co-occurrence frequencies. The factors that emerge represent clusters of verbs that share similar syntactic and semantic behaviour. To evaluate the performance on SCF acquisition, we identify the syntactic behaviour of each cluster. The SCF lexicon that emerges from the clusters achieves a promising F-score of 68.7 against a gold standard. We further introduce a novel SP evaluation in which we investigate the model's ability to induce preferences for the co-occurrence of a particular verb lemma and all of its arguments *at the same time*. The model achieves a high accuracy of 77.8 on this new evaluation. We also perform a qualitative evaluation which shows that the joint model is capable of learning rich lexical information about both syntactic and semantic aspects of verb behaviour in data.

2 Related Work

Recent SCF acquisition approaches use the output of an unlexicalized parser to generate SCF hypotheses, followed by statistical filtering and/or smoothing to remove noise. Briscoe and Carroll (1997); Korhonen (2002); Preiss et al. (2007) use handcrafted rules to match parser output to a pre-defined set of SCFs, achieving an F-measure of about 70 against a manually annotated gold standard, while O'Donovan et al. (2005); Chesley and Salmon-Alt (2006); Ienco et al. (2008); Messiant (2008); Lenci et al. (2008); Altamirano and Alonso i Alemany (2010); Kawahara and Kurohashi (2010) induce the inventory of SCFs from parsed corpus data. Candidate frames are identified by grammatical relation (GR) co-occurrences, often aided by language-specific heuristics. Statistical filtering or empirically-tuned thresholds are used to select frames for the final lexicon. These 'inductive' approaches have achieved respectable accuracy (60-70 F-measure against a dictionary) and are more portable than earlier methods. However, their ability to improve in accuracy is limited by their inability to incorporate information beyond the GR co-occurrences and heuristics that identify candidate SCFs on a per-sentence basis. Such cues provide no capacity for learning further from the data, e.g. from the lexical content of verbal arguments or from other GRs which are not part of the SCF.

Unsupervised machine learning has been applied to tasks where portability is equally important (Blei et al., 2003; Dinu and Lapata, 2010) but its application to SCF acquisition remains limited. Carroll and Rooth (1996) combined a head-lexicalized context-free grammar with an expectation-maximization (EM) algorithm to acquire an SCF lexicon. Dębowski (2009) used a filtering method based on the point-wise co-occurrence of arguments in parsed data to acquire a Polish SCF lexicon, but this method does not take the semantics of the verb's arguments into account. Lippincott et al. (2012) developed a graphical model for inducing verb frames in corpus data. The model identifies argument types of verbs but not *sets* of SCFs taken by a verb, as full scale SCF systems do.

Recent SP acquisition approaches use latent semantic information to model SPs, making use of probabilistic models, such as latent Dirichlet allocation (LDA) (Ó Séaghdha, 2010; Ritter and Etzioni, 2010; Reisinger and Mooney, 2011), or non-negative tensor factorization (NTF)

(Van de Cruys, 2009). Other approaches solely make use of distributional similarity methods (Bhagat et al., 2007; Basili et al., 2007; Erk, 2007). All approaches model two-way verb-argument co-occurrences, with the exception of Van de Cruys (2009) which models three-way verb-subject-object co-occurrences.

To our knowledge, no previous method has learned SCFs and SPs jointly. Scheible (2010) used SCFs as features in a Predicate-Argument Clustering (Schulte im Walde et al., 2008) approach to SP acquisition, but did not evaluate the resulting clusters for SCFs and found that the SP method did not outperform previous methods. Abend et al. (2009) used co-occurrence measures to perform unsupervised argument-adjunct discrimination for PPs, but not full SCFs.

Our method makes use of non-negative tensor factorization (NTF) (Shashua and Hazan, 2005). Tensor factorization is the multilinear generalization of matrix factorization. It has been extensively studied in the field of statistics (Kolda and Bader, 2009), and has yielded promising results on SP acquisition (Van de Cruys, 2009). We introduce a novel way of considering SCFs with an arbitrary number of arguments, and SPs as multi-way co-occurrences in the context of these larger SCFs. The resulting model provides an ideal framework for joint acquisition of SCF and SP information. The only form of supervision in the model is parameter estimation and choice of the best feature set via cross-validation.

3 Subcategorization Frame Inventory

To facilitate thorough qualitative evaluation (Section 5.6), we defined our SCFs in terms of syntactic slots, and in the form of common GRs. Finer-grained inventories including lexicalized elements and semantic interpretation were left for future work (see Section 7).

We use the GR types produced by the RASP parser (Briscoe and Carroll, 2002). Altogether we experimented with combinations of nine GR types out of the 13[1] which can be headed by verbs, selected on the basis of their frequency in the parsed BNC corpus and relevance for subcategorization. For this initial experiment, we focused on higher-frequency arguments since they will have the greatest impact on downstream applications.

Our first eight basic GR types are as follows. In subject position we included non-clausal subjects (SUBJ)[2], ignoring sentences with clausal subjects, which are much less frequent. Since objects are key arguments for subcategorization, we included all three object types – direct objects (DOBJ), second objects of ditransitive constructions (OBJ2), and prepositional arguments (IOBJ). Although OBJ2 is less frequent than other objects, it is important for identifying ditransitive frames. We included both types of clausal complements – XCOMP (infinitival/unsaturated) and CCOMP (finite/saturated) – and also PCOMP, which often signifies a *wh*-object of a preposition. We also included particles (PRT). Together, these eight GR types account for 62% of the GRs in the parsed BNC corpus. Using these GRs, there are 23 SCFs in our gold standard (see Section 5.1), of which the 15 with the highest type frequency are shown in Table 1.

Although modifiers are generally not included in SCFs (and are also excluded from our gold standard) we experimented with using them as features, to determine whether their distribution could help reach a better generalization. We focused on non-clausal modifiers (NCMOD). Counting them, the nine GR types account for 95% of the GRs in the BNC corpus.

[1] We count particles (here PRT) as a separate type, though RASP classifies them as a subtype of non-clausal modifiers.
[2] NCSUBJ in RASP.

Frame	Example sentence	Frame	Example sentence
SUBJ-DOBJ	Susan *found* the book.	SUBJ-XCOMP	Susan *wanted* to find the book.
SUBJ-DOBJ-IOBJ	Susan *put* the book on the table.	SUBJ-DOBJ-XCOMP	Susan *asked* Peter to attend.
SUBJ	Susan *knocked*.	SUBJ-DOBJ-IOBJ-PRT	Susan *filled* Peter in on the class.
SUBJ-IOBJ	Susan *appealed* to Peter.	SUBJ-CCOMP	Susan *believed* that Peter had found the book.
SUBJ-PRT	Susan *gave* up.	SUBJ-DOBJ-CCOMP	Susan *told* Betty that Peter had found the book.
SUBJ-DOBJ-PRT	Susan *picked* up the book.	SUBJ-DOBJ-OBJ2	Susan *gave* Betty a book.
SUBJ-PCOMP	Susan *thought* about whether she wanted to go.	SUBJ-IOBJ-XCOMP	Susan *appeared* to Peter to be worried.
SUBJ-IOBJ-PRT	Susan *gave* up on the project.		

Table 1: Fifteen SCFs with highest type frequency in our gold standard, with example sentences.

4 Methodology

4.1 Non-negative tensor factorization

Distributional co-occurrence data is usually represented in the form of a *matrix*. Matrices are perfectly suited for the representation of two-way co-occurrence data, but are unable to cope with multi-way co-occurrence data. We therefore make use of the generalization of a matrix, which is called a *tensor*. Tensor objects are able to encode co-occurrence data beyond two modes. Figure 1 shows a graphical comparison of a matrix and a tensor with three modes. Note that a tensor need not be restricted to three modes; in fact, our model requires tensors of up to 12 modes. Such tensors are difficult to represent visually, but the mathematical machinery remains unchanged.

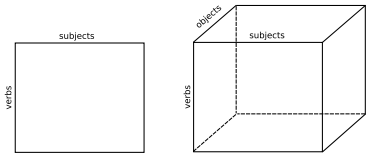


Figure 1: Matrix representation vs. tensor representation.

In order to create a succinct and generalized model of the extracted data, a statistical factorization technique called non-negative tensor factorization (NTF) is applied to the data. The NTF model is similar to parallel factor (PARAFAC) analysis – popular in areas such as psychology and bio-chemistry – with the constraint that all data needs to be non-negative (i.e. ≥ 0). PARAFAC is a multilinear analogue of the singular value decomposition (SVD), used e.g. in latent semantic analysis (Landauer and Dumais, 1997). The key idea is to minimize the sum of squares between the original tensor and the factorized model of the tensor. For an N-mode tensor $T \in \mathbb{R}^{D_1 \times D_2 \times \ldots \times D_N}$ this gives objective function (1), where k is the number of dimensions in the factorized model and $\circ$ denotes the outer product.

$$\min_{x_i \in \mathbb{R}^{D_1}, y_i \in \mathbb{R}^{D_2}, \ldots, z_i \in \mathbb{R}^{D_N}} \| T - \sum_{i=1}^{k} x_i \circ y_i \circ \ldots \circ z_i \|_F^2 \quad (1)$$

With non-negative tensor factorization, the non-negativity constraint is enforced, which yields a model with objective function (2).

$$\min_{x_i \in \mathbb{R}^{D_1}_{\geq 0}, y_i \in \mathbb{R}^{D_2}_{\geq 0}, \ldots, z_i \in \mathbb{R}^{D_N}_{\geq 0}} \| T - \sum_{i=1}^{k} x_i \circ y_i \circ \ldots \circ z_i \|_F^2 \quad (2)$$

The algorithm results in N matrices, indicating the loadings of each mode on the factorized dimensions. The model for the three-mode case is represented graphically in figure 2, visualizing the fact that the NTF decomposition consists of the summation over the outer products of N (in this case three) vectors.

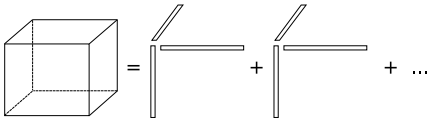


Figure 2: Graphical representation of the NTF as the sum of outer products.

Computationally, the NTF model is fitted by applying an alternating least-squares algorithm. In each iteration, two of the modes are fixed and the third one is fitted in a least squares sense. This process is repeated until convergence.[3]

4.2 Construction of verb-argument tensors

In order to discover SCFs and SPs, we construct a tensor that contains the multi-way co-occurrences of a verb and its different arguments.

4.2.1 Corpus data

We used a subset of the corpus of Korhonen et al. (2006), which consists of up to 10,000 sentences for each of approximately 6400 verbs, with data taken from five large British and American cross-domain corpora. To ensure sufficient data for each verb, we included verbs with at least 500 occurrences, yielding a total of 1993 verbs. The corpus data was tokenized, POS-tagged, lemmatized, and parsed with the RASP system (Briscoe and Carroll, 2002). RASP uses a tag-sequence grammar, and is unlexicalized, so that the parser's lexicon does not interfere with SCF acquisition. RASP produces output in the form of GRs. Passive sentences and those with clausal subjects were ignored.

4.2.2 Tensor construction

The corpus data is used to construct an N-mode tensor, where N represents the number of GRs. Each mode contains a different GR to the verb. Given the eight GRs from Section 3 plus the verb itself, this yields a 9-mode tensor (up to 12-mode when modifiers and split clausal modifiers are included; see Section 4.2.3).

For any particular verb instance (i.e. sentence), not every GR type will be instantiated. However, to model the multi-way co-occurrences in a tensor framework, each instance must have a feature for every mode to be incorporated into the tensor. Previous applications of non-negative tensor

[3]The algorithm has been implemented in MATLAB, using the Tensor Toolbox for sparse tensor calculations (Bader and Kolda, 2007).

factorization in NLP have not needed a representation for the non-instantiation of a mode. We introduce an empty, *void* (–) feature when a particular mode is not instantiated. For example, sentence (1) from Section 1 would be encoded as the tuple in (3):

(3) $\langle show_V, review_N, you_P, –, –, –, be_V, –, –\rangle$

indicating that the VERB, NCSUBJ, DOBJ, and CCOMP slots are filled with respectively $show_V$, $review_N$, you_P, *and* be_V, and that the remaining slots (IOBJ, OBJ2, PCOMP, XCOMP, PRT) are empty. (See Section 4.2.3 for alternative feature sets for each mode.)

Our final tensor then records how many times the tuple is attested in the corpus (i.e. how many times these particular features for the various grammatical relations occur together with the verb in question). The constructed tensor is then factorized to a limited number of latent dimensions, minimizing objective function (2). We normalize the factorization matrices to 1, to ensure a proper probability distribution.

Initially, we experimented with the number of latent dimensions of the factorization model (in the range 50–200). In further experiments, we retained the number of 150 dimensions, as this gave us the best results, and the model did not improve beyond 150 dimensions.

4.2.3 Feature sets

We constructed the feature sets for each mode in a number of different ways. Our base model uses the POS tag of the argument and no other features. We then experimented with a variety of additional features, based on linguistic intuitions about SCFS and SPS, as follows.

head The lexical head of the argument as well as the POS tag is used;

extpp PPS are extended to include the head of the PP's object, e.g. to_London_N (for the head models) or *to_N* (for the POS models) instead of simply *to*;

split both XCOMP and CCOMP are split up into two different modes to differentiate between null and lexicalized complementizers (e.g. for CCOMP, whether the complementizer is null or *that*);

mod modifiers (NCMOD) are included as an extra mode in the tensor.

Only the models with head features are relevant for SP acquisition. The head features also test how sensitive the learning of SCFS is to lexical-semantic as opposed to purely syntactic generalizations. The extended PP features provide additional lexical-semantic information. The clausal complement subtypes are available in the RASP output and offer a finer-grained syntactic analysis of these GRS. Finally, we used modifiers to test whether modifier co-occurrences, although (by definition) not part of the SCFS, might still be helpful in generalizing about subcategorization (i.e., maybe verbs taking similar frames also take similar modifiers). For each mode, we included the features that occurred with frequency ≥ 500 in the corpus, to maintain tractability.

For example, sentence (1) from Section 1 would be encoded as the tuple in (4) in the base POS-only model, and the tuple in (5) in the model with head and modifier features.

(4) $\langle show_V, N, P, –, –, –, V, –, –\rangle$

(5) $\langle show_V, review_N, you_P, –, –, –, be_V, –, –, comprehensively_R\rangle$

5 Experiment 1: SCF Induction

5.1 Evaluation method

SCF lexicons are traditionally evaluated against gold standards. We took the gold standard of Korhonen et al. (2006), which is a superset of SCFs in large dictionaries, and created a version using our eight basic GR types to define the SCFs. The resulting gold standard contains 183 general language verbs, with an average of 7.4 SCFs per verb. No attempt is made to distinguish between multiple senses of polysemous verbs; SCFs belonging to all senses are included for each lemma in the gold standard.

We evaluated the acquired SCF lexicons using type precision (percentage of SCF types that the system proposes which are correct), type recall (percentage of SCF types in the gold standard that the system proposes), and F-measure (the harmonic mean of type precision and recall).

We have two baselines. For baseline 1, we adopt the baseline of O'Donovan et al. (2005) which uniformly assigns to all verbs the two SCFs known to be most frequent in general language, transitive (SUBJ-DOBJ) and intransitive (SUBJ). This is a challenging baseline for SCF acquisition because of the Zipfian nature of SCF distributions: a small number of frequent SCFs are taken by the majority of verbs. For baseline 2, as described in Section 4.2.3, we use the base model with only POS features and none of the additional lexical or modifier features.

5.2 Mapping latent dimensions to SCFs

In order to evaluate this technique for SCF acquisition, we need to characterize each latent dimension according to its syntactic behaviour, i.e. map each dimension to a characteristic SCF.

Each latent dimension $\mathbf{z}$ is represented by a set of N vectors, indicating the loadings of each mode on $\mathbf{z}$. Because the loadings were normalized, each vector contains a probability distribution, over verbs or features. For a dimension $\mathbf{z}$ and a given mode (i.e. GR slot) we use the probability $p(\text{–}|\mathbf{z})$ of a void appearing in that slot to decide whether that slot is characteristically empty or filled for that dimension. For the verb mode, we use the probability $p(\mathbf{v}|\mathbf{z})$ to decide whether a verb $\mathbf{v}$ takes that dimension's characteristic SCF.

The mapping thus has two parameters. The first, θ_{verb}, represents the minimum $p(\mathbf{v}|\mathbf{z})$ for $\mathbf{v}$ to be assigned the characteristic SCF of $\mathbf{z}$. Based on early experiments, we chose to test three values for θ_{verb}, 0.001, 0.002, and 0.003.

The second parameter, θ_{void}, represents the maximum value of $p(\text{–}|\mathbf{z})$ at which the argument slot will be considered part of the SCF of $\mathbf{z}$. For example, if $p(\text{–}|\mathbf{z}) > \theta_{void}$ in the vector representing the DOBJ mode for $\mathbf{z}$, then the characteristic SCF of $\mathbf{z}$ does not include a direct object. We did not apply the θ_{void} threshold to subjects, but rather assumed that all characteristic SCFs include subjects; early experiments showed that subjects were otherwise sometimes erroneously excluded from the SCFs because the data contained high numbers of subjectless embedded clauses. For all other modes, we tested θ_{void} values from 0.1 to 0.8 in increments of 0.1.

The mapping process can be thought of as labeling the clusters produced by the tensor factorization. E.g. for a latent dimension $\mathbf{z}$ with a void value below θ_{void} for the DOBJ and IOBJ modes, its label is simply SUBJ-DOBJ-IOBJ. This label is assigned as an SCF to all the verbs with probabilities over θ_{verb} in $\mathbf{z}$.

If a dimension's characteristic SCF does not correspond to an SCF in the gold standard, that

Frame	# dims	Frame	# dims
SUBJ-DOBJ	29	SUBJ-XCOMP	17
SUBJ-DOBJ-IOBJ	9	SUBJ-DOBJ XCOMP	5
SUBJ	24	SUBJ-DOBJ-IOBJ-PRT	0
SUBJ-IOBJ	12	SUBJ-CCOMP	26
SUBJ-PRT	7	SUBJ-DOBJ-CCOMP	0
SUBJ-DOBJ-PRT	5	SUBJ-DOBJ-OBJ2	5
SUBJ-PCOMP	3	SUBJ-IOBJ-XCOMP	0
SUBJ-IOBJ-PRT	3		

Table 2: SCFs in order of type frequency in gold standard, with number of latent dimensions mapped to them (model features: POS, modifiers).

cluster is excluded from the evaluation. This typically happens with high values of θ_{void} because too many argument slots are simultaneously included in the SCF.

Note that multiple dimensions may be mapped to each SCF, because we chose the number of latent dimensions to be greater than the number of SCFs in the inventory. This decision allows the system to detect semantic structure in the data at a finer-grained level, which we hypothesized would improve overall accuracy on subcategorization acquisition, and to discover multiple lexical classes within a single argument slot. The relationship between number of dimensions mapping to an SCF and the frequency of the SCF is somewhat complex. To the extent that different verbs appear in different clusters, we expect that a larger number of dimensions mapping to an SCF roughly corresponds to higher type frequency of the SCF. However, some clusters contain more verbs than others; also, more clusters may indicate more semantic variability in argument slot fillers, without corresponding to higher frequency. A general relationship between type frequency and number of dimensions can be seen in Table 2, although note the high number of clusters mapped to the clausal complement frames SUBJ-XCOMP and SUBJ-CCOMP, possibly because these relations are semantically variable and used for adjuncts as well as arguments.

5.3 Tuning parameters

We used ten-fold cross-validation to tune the parameters θ_{verb} and θ_{void}, as well as to select the best feature combination (see Section 4.2.3). We randomly divided our test verbs into ten sets, each containing either 18 or 19 verbs. For each fold, we selected the parameters that gave the highest accuracy on the remaining nine-tenths of the verbs agaist the gold standard, and used those settings to acquire the lexicon for the 18 or 19 verbs in the fold.

For all ten folds, the best result was achieved with $\theta_{verb} = 0.001$ and $\theta_{void} = 0.4$, and with modifier features, but without extended PPs or split clause types. For seven of the folds, the best result was achieved with POS features, and for the other three with head features.

5.4 Results

Table 3 shows the results for our system after tuning with cross-validation. The parameters are: $\theta_{verb} = 0.001$, $\theta_{void} = 0.4$, POS and modifier features. Precision and recall are averaged over the ten folds. The standard deviation for precision was 4.3 and for recall 5.9. The final system achieves an F-measure of 68.7, well above the baseline 1 F-measure of 36.9, and nearly four

	P	R	F
Baseline 1	86.3	23.5	36.9
Baseline 2 (POS features)	53.1	83.3	64.8
Final system	61.0	78.5	68.7

Table 3: Results of cross-validation experiment. Precision and recall averaged over ten folds. F-score calculated as harmonic mean of the average P and R.

points better than the baseline 2 F-measure of 64.8. All of the improvement over baseline 2 is in precision, which shows that adding features beyond simple GR co-occurrences is beneficial to accurate SCF acquisition. Because of the Zipfian nature of SCF distributions, the system does not match the precision of baseline 1.

Direct comparison against previous unsupervised SCF acquisition methods on English was not possible because of the use of different data and frame inventories. However, best current methods involving handcrafted rules have reached a ceiling at an F-measure of about 70 (Korhonen et al., 2006; Preiss et al., 2007). Our results are promising considering the challenges of less supervised lexical acquisition.

5.5 Investigation of features

We also investigated the contribution of the different feature sets on the entire gold standard, using the values for θ_{verb} and θ_{void} which emerged from the cross-validation. The results of the different models are shown in Table 4 (note that the best result is slightly different from that in Table 3 because it is on the entire gold standard, not averaged over folds).

	Model							
	head	PP	split	mod	P	R	F	cov
1				•	61.4**,††	81.1**,††	**69.9**††	183
2	•			•	63.9**,††	76.4**,††	69.6††	183
3	•		•		67.2**,††	70.4**,††	68.8††	183
4		•	•		59.3††	80.9††	68.4††	183
5		•			58.7**,††	81.2**,††	68.2††	183
6		•		•	60.5*,††	77.9**,††	68.1††	183
7			•	•	58.7**,††	81.2**,††	68.1††	182
8		•	•	•	61.2**,††	76.0**,††	67.8††	183
9	•	•	•		**67.5****,††	67.7**,††	67.6††	183
10			•		56.1**,††	83.1**	67.0††	183
11	•	•			60.2††	74.3**,††	66.5†	182
12	•	•		•	61.8*,††	71.4*,††	66.3	183
13	•				59.8**,††	73.6**,††	66.0	183
14					53.1**	**83.3****	64.8*	183
15	•		•	•	65.1††	60.3**,††	62.6**,†	183
16	•	•	•	•	63.3††	52.6††	57.5††	181

Table 4: Results for each feature set, with 150 dimensions, $\theta_{verb} = 0.001$, $\theta_{void} = 0.4$. ** significant difference from next row with $p < 0.01$, * with $p < 0.05$. †† significant difference from baseline (row 14) with $p < 0.01$, † with $p < 0.05$.

The differences in F-measure between the top few models are rather small, but the models show wide variance in precision and recall. Using the head words of the arguments as features seems to favor precision (rows 2, 3, 9, 15, 16), while using POS tags favors recall. This is probably because evidence for different arguments is less sparse using POS tags, making less frequent frames easier to identify, but finer-grained distinctions more difficult. The highest F-scores are achieved with modifier features (rows 1, 2); however, these models strongly favor recall over precision, suggesting that the general applicability of modifiers to many verb classes interferes with accurate identification of SCFs. More balanced models have head features and split clausal complement types (row 3), or head features, extended PPs, and split clausal types (row 9), without losing out on F-score. This suggests that lexical-semantic features are valuable for SCF acquisition. Another trend is towards more accurate models with fewer additional features; individual features and pairs of features seem to provide the most improvement (rows 1-7) over the base model (row 14), but the model with all additional features (row 16) has markedly worse performance, which may indicate a data sparsity problem.

We carried out significance tests for the mentioned model differences using stratified shuffling (Yeh, 2000). These tests indicate that most of the models (rows 1-11) have significantly higher F-score than the baseline, and most show significant pairwise differences in precision and recall.

Parameter tuning with cross-validation resulted in a θ_{void} of 0.4 (though exploration of the models in Table 4 showed that some models performed better with even lower values). This means that the model only needs to assign a relatively low confidence score to the void feature to infer that a slot is not part of an SCF. This is probably because adjuncts and other noise in the data means that these slots are filled some of the time. We observed many cases of void probabilities nearly equal to 1 in various dimensions – most verbs never occur with an OBJ2, for example. However, void probabilities tend to be fairly low for CCOMP and XCOMP.

5.6 Qualitative evaluation

Table 5 shows the accuracy by SCF for the fifteen most frequent frames, using the final model that resulted from cross-validation. The system performs very well on a number of SCFs, especially the most frequent ones such as SUBJ-DOBJ, SUBJ-DOBJ-IOBJ, and SUBJ, but also on some SCFs involving the semantically important particle verbs, such as SUBJ-DOBJ-PRT and SUBJ-IOBJ-PRT. Precision is lower on frames involving clausal complements (XCOMP and CCOMP), possibly because these GRs are used frequently for adjuncts. Accuracy is also poor on SUBJ-PCOMP and SUBJ-DOBJ-OBJ2. These GRs are rarer and may be subject to parser errors (e.g. OBJ2).

6 Experiment 2: SP Induction

6.1 Introduction

Our second experiment looks at the model's ability to induce SPs. We investigate the model's ability to induce *multi-way* SPs, i.e. the preference of the model for the co-occurrence of a particular verb and all of its particular arguments *at the same time*.

The calculation of a SP value according to our NTF model is fairly straightforward. Recall that our model yields probabilities $p(\mathbf{v}|\mathbf{z})$, i.e. the probability of a verb given a latent dimension, and, for each argument to the verb, $p(\mathbf{g}|\mathbf{z})$, i.e. the probability of an argument given a latent dimension. The final SP value $SP(v, GR)$ for a particular verb v and a list of arguments GR then amounts to calculating the product of the probabilities of the verb and the various GRs given a particular latent dimension, and summing over all dimensions (equation 3).

Frame	P	R	F	Frame	P	R	F
SUBJ-DOBJ	95.4	98.8	97.0	SUBJ-XCOMP	44.0	98.6	60.9
SUBJ-DOBJ-IOBJ	89.6	88.5	89.0	SUBJ-DOBJ-XCOMP	45.9	79.4	58.1
SUBJ	82.7	98.7	90.0	SUBJ-DOBJ-IOBJ-PRT	0.0	0.0	0.0
SUBJ-IOBJ	80.6	91.5	85.7	SUBJ-CCOMP	35.9	100.0	52.8
SUBJ-PRT	75.2	87.1	80.7	SUBJ-DOBJ-CCOMP	33.3	71.1	45.4
SUBJ-DOBJ-PRT	72.8	83.0	77.6	SUBJ-DOBJ-OBJ2	20.0	90.3	32.8
SUBJ-PCOMP	56.9	45.7	50.7	SUBJ-IOBJ-XCOMP	0.0	0.0	0.0
SUBJ-IOBJ-PRT	71.9	83.1	77.1				

Table 5: Results by SCF for fifteen most frequent frames in gold standard with best-performing model.

$$SP(v, GR) = \sum_{i=1}^{k} p(v|z_i) \prod_{g \in GR} p(g|z_i) \quad (3)$$

We evaluate our method's ability to induce SPs using the lexicalized (HEAD) model that achieves the best score in our first experiment, i.e. model 2 in Table 4.

6.2 Evaluation method

To evaluate the results of the NTF model with regard to SPs, we make use of a pseudo-disambiguation task (similar to the one used by Rooth et al. (1999)). The task allows us to evaluate the generalization capabilities of the model. For a particular tuple (viz. a verb and its various arguments) that appears in a held-out test corpus, we generate random instances in which one or several arguments are substituted by random instantiations. We exhaustively substitute every individual argument, as well as the various random combinations.[4] For the sentence in (1), this yields instances like:

(6) $\langle show_V, rabbit_N, you_P, –, –, –, be_V, –, –\rangle$

(7) $\langle show_V, consumption_N, tunnel_N, –, –, –, dream_V, –, –\rangle$

We then calculate SP values according to our model, both for the corpus instance and the random instances. A tuple is considered correct if our model prefers the corpus instance over all random instances. Accuracy is then calculated by averaging over all instances that are part of the test corpus.

We compare our NTF model to a simple non-negative matrix factorization (NMF) model, comparable to the unsupervised model presented by Rooth et al. (1999). For this model, a matrix was constructed that contains the pairwise co-occurrence frequencies of verbs and their various arguments. As noted before, a matrix is only able to represent two modes; hence, the first mode consists of the verbs, while the second mode contains the concatenated list of the different argument features. We used the same number of features as with the NTF model, and also factorized to 150 dimensions. According to the NMF model, a tuple is considered correct if, for each argument to the verb, the model prefers the verb-argument pair containing the attested argument over the verb-argument pair containing the random substitute. As a baseline, we

[4] We do not substitute empty argument slots with lexical arguments; neither do we substitute filled arguments slots with *void* values. This experiment solely focuses on the induction of selectional preferences; the induction of SCFs is evaluated in experiment 1.

include an uninformed random model, which makes a random choice among the various possibilities.[5]

The models are evaluated using ten-fold cross-validation: the corpus is divided into 10 equal parts; in each fold, models are trained on nine tenths of the corpus, and tested on the remaining tenth.

6.3 Results

The results of the ten-fold cross-validation are shown in table 6. The NTF model clearly outperforms the matrix factorization model with regard to the reconstruction of SPS, with the NTF model reaching a score about 10% higher than its NMF counterpart. These results indicate that the use of multi-way data leads to a richer and more accurate representation of SPS. For comparison, (Van de Cruys, 2009) achieved accuracy of 90.89 on a three-way pseudo-disambiguation task, which is less complex than our eight-way task.

	accuracy (%)
baseline	29.21 ± .08
NMF	69.71 ± .28
NTF	**77.78** ± .17

Table 6: Selectional preference accuracy using ten-fold cross-validation (mean accuracy and standard deviation)

6.4 Qualitative evaluation

Additionally, we performed a qualitative evaluation of the 150 latent dimensions yielded by our NTF model. This evaluation shows that our model is indeed able to capture semantic information from the data. Recall from Section 5.2 that multiple dimensions map to a single SCF. Our cluster analysis shows that such dimensions reflect semantic information. Below are three example dimensions (denoted by the top 10 verbs with highest value on each dimension) that all map to a simple transitive SCF.

dim 29 buy, sell, use, collect, produce, handle, remove, purchase, obtain, eat

dim 38 kill, love, see, like, marry, know, meet, visit, help, say

dim 44 examine, identify, see, consider, assess, investigate, discuss, study, determine, explore

The three different transitive SCFS clearly exhibit different semantic properties. Dimension 29 seems to represent a general 'trading' dimension, in which the DOBJ argument contains inanimate objects, largely goods. The DOBJ argument has nouns such as *thing*, *material*, *food*,... as its top features. Dimension 38, on the other hand, is a transitive frame where the DOBJ argument takes animate objects. The last dimension 44 represents a transitive frame in which the DOBJ argument takes abstract objects.

Among the dimensions that map to the SUBJ-IO SCF, i.e. a single PP argument, there are also some interesting semantic and syntactic distinctions. Dimension 91 clearly represents a 'travel' cluster with a location complement; the IO slot for this dimension is mostly PPs lexicalized with *to*. Dimension 122 is a 'communication' cluster, and again most of the prepositions in the IO slot are *to*. Dimension 123 consists of verbs that occur with *at*, largely vision and non-verbal

[5] Note that the number of possibilities for both tensor and matrix model is exactly the same.

communication verbs. Finally, dimension 134 is interesting, because there is no clear semantic cohesion, but it represents cluster of verbs that take PP *for*. This indicates that the model is learning both semantic and syntactic regularities.

dim 91 go, come, return, move, walk, get, run, rush, travel, fly

dim 122 talk, speak, listen, write, belong, happen, appeal, come, say, lie

dim 123 look, stare, smile, laugh, shout, gaze, glance, glare, grin, scream

dim 134 wait, pay, look, care, work, ask, vote, call, prepare, apply

The results presented here indicate that our model is able to capture syntactic as well as semantic properties. On a coarse-grained level, our model is able to induce a verb's different SCF frames. When we zoom in to the level of individual clusters, we notice that these clusters are often semantically cohesive, expressing the selectional preferences of the verb's argument slots. The ability to capture both syntax and semantics is an important advantage of our method.

7 Conclusion

We have presented a novel method for joint unsupervised SCF and SP acquisition which allows the incorporation of a range of features (syntactic, lexical and semantic) in the acquisition process. Although SCFs and SPs are closely related and can complement each other, to the best of our knowledge, no previous work has proposed a joint model for them.

Applying NTF to the multi-way co-occurrence tensor of verbs and their arguments, we are able to cluster verbs from a large corpus according to their syntactic and semantic behaviour. The SCF lexicon that emerges from the clusters yields an F-score of 68.7 against a gold standard, outperforming lexicons produced by our baseline methods. This performance is promising for a largely unsupervised method. The model yields an accuracy of 77.8 on a new pseudo-disambiguation evaluation for SPs, in which all arguments of the verb are considered at once, clearly outperforming a matrix factorization model. Our qualitative evaluation reveals that the method is indeed capable of learning rich lexical information about both syntactic and semantic aspects of verb behaviour in corpus data.

In the future, we plan to improve our approach in several directions. In addition to improving the detection of low accuracy SCFs through the use of lexical features that may help to distinguish arguments from adjuncts in clausal complements, we plan to improve precision by using e.g. statistical filtering. We also plan to extend the model to acquire finer-grained SCFs for English. This will involve e.g. refining SCFs with lexicalized elements and including semantically-based SCFs in the inventory, making use of the factorization method's ability to induce latent structure, as demonstrated by the SP evaluation. Finally, we intend to improve our SP acquisition through the use of a more extensive feature set.

A key advantage of this approach is that it is able to combine syntactic SCF and semantic SP acquisition. In the future, we plan to explore the joint induction of verb syntax and semantics in greater depth and look into modelling additional information about semantic verb classes which tend to capture similar SCF and SP behaviour. This could facilitate inducing a more comprehensive lexical resource that supplements the SCFs and SPs with a verb classification – in the style of VerbNet (Kipper-Schuler, 2005) – providing generalizations that can be useful for a wider range of NLP tasks.

Acknowledgements

The work in this paper was funded by the Royal Society (UK), the Isaac Newton Trust (Cambridge, UK), EPSRC grant EP/G051070/1 (UK) and EU grant 7FP-ITC-248064 'PANACEA'.

References

Abend, O., Reichart, R., and Rappoport, A. (2009). Unsupervised argument identification for semantic role labeling. In *Proceedings of ACL*.

Altamirano, I. R. and Alonso i Alemany, L. (2010). IRASubcat, a highly customizable, language independent tool for the acquisition of verbal subcategorization information from corpus. In *Proceedings of the NAACL HLT 2010 Young Investigators Workshop on Computational Approaches to Languages of the Americas*, pages 84–91, Los Angeles, CA.

Arun, A. and Keller, F. (2005). Lexicalization in crosslinguistic probabilistic parsing: The case of french. In *Proceedings of ACL*, Ann Arbor, Michigan.

Bader, B. W. and Kolda, T. G. (2007). Matlab tensor toolbox version 2.2. `http://csmr.ca.sandia.gov/~tgkolda/TensorToolbox/`.

Basili, R., Cao, D. D., Marocco, P., and Pennacchiotti, M. (2007). Learning selectional preferences for entailment or paraphrasing rules. In *Proceedings of RANLP 2007*, Borovets, Bulgaria.

Bhagat, R., Pantel, P., and Hovy, E. (2007). Ledir: An unsupervised algorithm for learning directionality of inference rules. In *Proceedings of Conference on Empirical Methods in Natural Language Processing (EMNLP-07)*, page 161–170, Prague, Czech Republic.

Bharati, A., Venkatapathy, S., and Reddy, P. (2005). Inferring semantic roles using subcategorization frames and maximum entropy model. In *Proceedings of CoNLL*, pages 165–168, Ann Arbor.

Blei, D., Ng, A., and Jordan, M. (2003). Latent dirichlet allocation. *The Journal of Machine Learning Research*, 3:993–1022.

Briscoe, T. and Carroll, J. (1997). Automatic extraction of subcategorization from corpora. In *Proceedings of the fifth conference on Applied natural language processing*, pages 356–363. Association for Computational Linguistics.

Briscoe, T. and Carroll, J. (2002). Robust accurate statistical annotation of general text. In *Proceedings of the 3rd International Conference on Language Resources and Evaluation*, pages 1499–1504.

Carroll, G. and Rooth, M. (1996). Valence induction with a head-lexicalized pcfg. In *Proceedings of the 3rd Conference on Empirical Methods in Natural Language Processing (EMNLP 3)*, pages 36–45.

Carroll, J. and Fang, A. (2004). The automatic acquisition of verb subcategorisations and their impact on the performance of an HPSG parser. In *Proceedings of the 1st International Joint Conference on Natural Language Processing (IJCNLP)*, pages 107–114, Sanya City, China.

Chesley, P. and Salmon-Alt, S. (2006). Automatic extraction of subcategorization frames for french. In *Proceedings of the Language Resources and Evaluation Conference (LREC)*, pages 253–258, Genoa, Italy.

Cholakov, K. and van Noord, G. (2010). Using unknown word techniques to learn known words. In *Proceedings of EMNLP*, pages 902–912, Massachusetts.

Dinu, G. and Lapata, M. (2010). Measuring distributional similarity in context. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 1162–1172. Association for Computational Linguistics.

Dębowski, Ł. (2009). Valence extraction using EM selection and co-occurrence matrices. *Language resources and evaluation*, 43(4):301–327.

Erk, K. (2007). A simple, similarity-based model for selectional preferences. In *Proceedings of ACL 2007*, Prague, Czech Republic.

Gildea, D. and Jurafsky, D. (2002). Automatic labeling of semantic roles. *Computational Linguistics*, page 245–288.

Hajič, J., Čmejrek, M., Dorr, B., Ding, Y., Eisner, J., Gildea, D., Koo, T., Parton, K., Penn, G., Radev, D., and Rambow, O. (2002). Natural language generation in the context of machine translation. Technical report, Center for Language and Speech Processing, Johns Hopkins University, Baltimore. Summer Workshop Final Report.

Han, C., Lavoie, B., Palmer, M., Rambow, O., Kittredge, R., Korelsky, T., and Kim, M. (2000). Handling structural divergences and recovering dropped arguments in a korean/english machine translation system. In *Proceedings of the AMTA*.

Ienco, D., Villata, S., and Bosco, C. (2008). Automatic extraction of subcategorization frames for italian. In *Proceedings of the Sixth International Conference on Language Resources and Evaluation (LREC'08)*, Marrakech, Morocco.

Kawahara, D. and Kurohashi, S. (2010). Acquiring reliable predicate-argument structures from raw corpora for case frame compilation. In *Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC'10)*, Valletta, Malta.

Kipper-Schuler, K. (2005). *VerbNet: A broad-coverage, comprehensive verb lexicon*. PhD thesis, University of Pennsylvania, Philadelphia, PA.

Kolda, T. G. and Bader, B. W. (2009). Tensor decompositions and applications. *SIAM Review*, 51(3):455–500.

Korhonen, A. (2002). Semantically motivated subcategorization acquisition. In *Proceedings of the ACL-02 workshop on Unsupervised lexical acquisition-Volume 9*, pages 51–58. Association for Computational Linguistics.

Korhonen, A., Krymolowski, Y., and Briscoe, T. (2006). A large subcategorization lexicon for natural language processing applications. In *Proceedings of LREC*, Genoa, Italy.

Landauer, T. and Dumais, S. (1997). A solution to Plato's problem: The Latent Semantic Analysis theory of the acquisition, induction, and representation of knowledge. *Psychology Review*, 104:211–240.

Lenci, A., McGillivray, B., Montemagni, S., and Pirrelli, V. (2008). Unsupervised acquisition of verb subcategorization frames from shallow-parsed corpora. In *Proceedings of the Sixth International Conference on Language Resources and Evaluation (LREC'08)*, Marrakech, Morocco.

Lippincott, T., Korhonen, A., and Ó Séaghdha, D. (2012). Learning syntactic verb frames using graphical models. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (ACL 2012)*, Jeju, Korea.

Messiant, C. (2008). ASSCI: A subcategorization frames acquisition system for French verbs. In *Proceedings of the Association for Computational Linguistics (ACL, Student Research Workshop)*, pages 55–60, Columbus, Ohio.

Moschitti, A. and Basili, R. (2005). Verb subcategorization kernels for automatic semantic labeling. In *Proceedings of the ACL-SIGLEX Workshop on Deep Lexical Acquisition*, pages 10–17, Ann Arbor.

O'Donovan, R., Burke, M., Cahill, A., van Genabith, J., and Way, A. (2005). Large-scale induction and evaluation of lexical resources from the penn-ii and penn-iii treebanks. *Computational Linguistics*, 31:328–365.

Ó Séaghdha, D. (2010). Latent variable models of selectional preference. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics (ACL)*, Uppsala, Sweden.

Preiss, J., Briscoe, T., and Korhonen, A. (2007). A system for large-scale acquisition of verbal, nominal and adjectival subcategorization frames from corpora. In *Proceedings of ACL*, Prague, Czech Republic.

Reisinger, J. and Mooney, R. (2011). Crosscutting models of lexical semantics. In *Proceedings of EMNLP-11*, Edinburgh, UK.

Resnik, P. (1997). Selectional preference and sense disambiguation. In *Proc. of the ACL SIGLEX Workshop on Tagging Text with Lexical Semantics: Why, What, and How?*

Ritter, A. and Etzioni, O. (2010). A latent dirichlet allocation method for selectional preferences. In *In Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics (ACL*.

Rooth, M., Riezler, S., Prescher, D., Carroll, G., and Beil, F. (1999). Inducing a semantically annotated lexicon via em-based clustering. In *37th Annual Meeting of the ACL*.

Scheible, C. (2010). An evaluation of predicate argument clustering using pseudo-disambiguation. In *Proceedings of LREC*.

Schulte im Walde, S. (2006). Experiments on the automatic induction of german semantic verb classes. *Computational Linguistics*, 32(2):159–194.

Schulte im Walde, S., Hying, C., Scheible, C., and Schmid, H. (2008). Combining EM training and the MDL principle for an automatic verb classification incorporating selectional preferences. In *Proceedings of the ACL*, pages 496–504, Columbus, OH.

Seaghdha, D. O. and Korhonen, A. (2011). Probabilistic models of similarity in syntactic context. In *Proceedings of EMNLP-11*, Edinburgh, UK.

Shashua, A. and Hazan, T. (2005). Non-negative tensor factorization with applications to statistics and computer vision. In *Proceedings of the 22nd international conference on Machine learning*, pages 792–799. ACM.

Sun, L. and Korhonen, A. (2011). Hierarchical verb clustering using graph factorization. In *Proceedings of EMNLP*, pages 1023–1033, Edinburgh, Scotland.

Thater, S., Furstenau, H., and Pinkal, M. (2010). Contextualizing semantic representations using syntactically enriched vector models. In *Proceedings of ACL-10*, Uppsala, Sweden.

Van de Cruys, T. (2009). A non-negative tensor factorization model for selectional preference induction. In *Proceedings of the workshop on Geometric Models for Natural Language Semantics (GEMS)*, pages 83–90, Athens, Greece.

Yeh, A. (2000). More accurate tests for the statistical significance of result differences. In *Proceedings of the 18th conference on Computational linguistics*, pages 947–953, Saarbrücken, Germany.

Zapirain, B., Agirre, E., and Marquex, L. (2009). Generalizing over lexical features: Selectional preferences for semantic role classification. In *Proceedings of ACL-IJCNLP-09*, Singapore.

Zhou, G., Zhao, J., Liu, K., and Cai, L. (2011). Exploiting web-derived selectional preference to improve statistical dependency parsing. In *Proceedings of ACL-11*, Portland, OR.

Sub-corpora Sampling with an Application to Bilingual Lexicon Extraction

Ivan VULIĆ *Marie-Francine MOENS*
Department of Computer Science
KU Leuven
Celestijnenlaan 200A
Leuven, Belgium
{ivan.vulic,marie-francine.moens}@cs.kuleuven.be

ABSTRACT
We propose a novel associative approach for bilingual word lexicon extraction (BLE) from parallel corpora that relies on the paradigm of data reduction instead of data augmentation. The key insight of the approach is the effective usage of sub-corpora sampling and properties of low-frequency words in the task of lexicon induction, particularly in a setting where only limited parallel data are available. Word translation pairs are extracted from many smaller sub-corpora (sampled from the original corpus) according to several frequency-based criteria of similarity. We prove the validity of our data sampling approach, and show that this method outperforms IBM Model 1 and associative methods based on similarity scores and hypothesis testing in terms of precision and F-measure in the task of lexicon extraction. Additionally, we show that our sampling-based method can learn correct word translations from fewer data.

TITLE AND ABSTRACT IN ANOTHER LANGUAGE (CROATIAN)

Uzorkovanje Potkorpusa uz Primjenu u Ekstrakciji Dvojezičnih Rječnika

U radu se predlaže nov asocijativan pristup ekstrakciji dvojezičnih rječnika iz usporednih korpusa koji se oslanja na paradigmu smanjivanja količine podataka umjesto njezinog povećavanja. Ključna je ideja pristupa učinkovita uporaba uzorkovanja potkorpusa te svojstava niskofrekventnih riječi u zadatku indukcije rječnika, posebice u situacijama kada je na raspolaganju ograničen skup usporednih podataka. Prijevodni parovi riječi ekstrahirani su iz većeg broja manjih potkorpusa (uzorkovanih iz izvornog korpusa) temeljem nekoliko frekvencijski utemeljenih kriterija sličnosti. U radu je pokazana ispravnost našeg pristupa temeljenog na uzorkovanju potkorpusa. Pokazano je da ovaj postupak u smislu F-mjere na zadatku ekstrakcije leksikona nadmašuje IBM-ov Model 1 te asocijativne postupke temeljene na ocjenama sličnosti i testiranju hipoteze. Također je pokazano da naš postupak temeljen na uzorkovanju može naučiti ispravne prijevode riječi iz manjih količina podataka.

KEYWORDS: bilingual lexicon extraction, empirical word translation, sub-corpora sampling, data reduction, low-frequency words.

KEYWORDS IN CROATIAN: ekstrakcija dvojezičnih rječnika, empirijsko prevođenje riječi, uzorkovanje potkorpusa, smanjivanje količine podataka, niskofrekventne riječi.

Proceedings of COLING 2012: Technical Papers, pages 2721–2738,
COLING 2012, Mumbai, December 2012.

1 Introduction

Bilingual word lexicons serve as an invaluable and indispensable source of knowledge for both end users (as an aid for translators or other language specialists) and many natural language processing tasks, such as dictionary-based cross-language information retrieval (Carbonell et al., 1997; Levow et al., 2005) and statistical machine translation (Och and Ney, 2003).

In order to construct high quality bilingual lexicons for various domains, it is necessary to build such lexicons manually by hand or extract them automatically from parallel corpora. Compiling such lexicons manually is often a labor-intensive and time-consuming task, whereas parallel corpora either do not exist or are of limited size for most language pairs. Therefore the focus of the researchers has turned towards bilingual lexicon extraction (BLE) from comparable corpora (Rapp, 1995; Fung and Yee, 1998; Rapp, 1999; Diab and Finch, 2000; Fung and Cheung, 2004; Morin et al., 2007; Haghighi et al., 2008; Laroche and Langlais, 2010; Andrade et al., 2010; Shezaf and Rappoport, 2010; Vulić et al., 2011; Prochasson and Fung, 2011; Vulić and Moens, 2012; Tamura et al., 2012). However, such lexicons contain a great deal of noise and, moreover, the methods for BLE from comparable corpora typically rely on seed lexicons which are again hand-built or extracted from parallel corpora.

With respect to that observation, numerous systems for various applications trained on parallel or comparable data almost exclusively rely on knowledge from bilingual lexicons extracted from parallel texts. These lexicons are usually acquired from word translation probabilities of the IBM alignment models (Brown et al., 1993; Och and Ney, 2003) or obtained by associative methods such as the log-likelihood score or the Dice coefficient. They are then used in systems for extracting parallel sentences from non-parallel corpora (Fung and Cheung, 2004; Munteanu and Marcu, 2005), bilingual sentence alignment (Moore, 2002), estimating phrase translation probabilities (Venugopal et al., 2003), extracting parallel sub-sentential fragments from non-parallel corpora (Munteanu and Marcu, 2006), word-level confidence estimation (Ueffing and Ney, 2007), sub-sentential alignment for terminology extraction (Lefever et al., 2009), cross-lingual text classification and plagiarism detection (Pinto et al., 2009) and others.

High accuracy of automatically constructed bilingual word lexicons is the top priority for these systems. Church and Mercer (1993) advocate a simple solution of collecting more data in order to utilize statistical and stochastic methods in a more effective way. However, these systems are typically faced with only limited parallel data for many language pairs and domains (Resnik and Smith, 2003).

In order to tackle these issues, we propose a novel approach built upon the idea of *data reduction* instead of *data augmentation*. The method is directed towards extraction of only *highly reliable translation pairs* from *parallel data of limited size*. It is based on the idea of *sub-corpora sampling* from the original corpus. For instance, given an initial corpus $\mathscr{C}$ of 4 data items $\{I_1, I_2, I_3, I_4\}$, the construction of, say, a sub-corpus $SC = \{I_2, I_4\}$ may be observed as: (1) sampling items $I_2, I_4 \in \mathscr{C}$ for SC (hence the term sub-corpora sampling) or (2) removing data items I_1, I_3 from the original corpus $\mathscr{C}$, so that $SC = \mathscr{C} - \{I_1, I_3\}$ (hence the term data reduction). By reducing the size of the initial corpus, we typically decrease frequencies of the words in a newly formed sub-corpus. This simplifies the establishment of potential translation candidates, since that is now reduced to a problem of establishing reliable translational equivalence between low-frequency words. We explain the method for establishing translational equivalence based on the absolute frequency distributions of words in a sub-corpus. We exploit it in the construction of the algorithm for BLE. Moreover, each word exhibits a different distribution over items in each

newly built sub-corpus, and it is different from the fixed distribution in the original corpus. It allows us to identify different potential translation candidates in different sub-corpora and then form word translation tables by combining these evidences acquired from different sub-corpora. The key strength of the proposed algorithm is that it takes the entire initial corpus into account, regardless of its size, and at the same time it also benefits from the sampling of a vast number of different subsets/sub-corpora sampled from that initial corpus, and the evidences of potential word translation pairs coming from these sub-corpora.

In the remainder of the paper, we show that: (1) Bilingual lexicon extraction benefits from the concept of data reduction and sub-corpora sampling - the key intuitions, assumptions and the construction of the algorithm are provided in Section 2; (2) The proposed algorithm for BLE removes a lot of noise from the bilingual word lexicons by harvesting only the most accurate translation candidates, and it outscores other standard models for BLE from parallel data; (3) Due to the concept of data reduction, the proposed algorithm does not suffer from a problem of indirect associations; (4) Most importantly, the proposed algorithm outperforms other models for BLE when dealing with parallel data of limited size. The results are presented in Section 4. Finally, Section 5 lists conclusions and possible paths of future work.

2 Learning Translation Pairs Using Sub-Corpora Sampling

Section 1 has already provided a general intuition behind our method for mining translational candidates from aligned corpora. Now, we provide an in-depth description and analysis of our algorithm for bilingual word lexicon extraction. First, we explain the key reasoning that led us to our approach that relies on *data sampling*. Second, we provide the criteria for extracting translation candidates that purely rely on their distributional features, but do not employ any similarity-based measure or hypothesis testing for word association, and finally, we present our algorithm for BLE that processes words of all frequencies in an uniform way.

2.1 Why Sampling Sub-corpora?

The foundation of this work is built upon the so-called *Zipfian phenomenon* which states that, regardless of the size of a corpus, most of the distinct words occur only a small number of times. For instance, Moore (2004b) measures that in the first 500,000 English sentences taken from the Canadian Hansards data (Mihalcea and Pedersen, 2003), one finds 52,921 distinct word types, of which 60.5% occur five or fewer times, and, moreover, 32.8% occur only once. A general solution to mitigate the problem of low-frequency words is by augmenting the amount of input training data. However, that approach leads to a *chicken and egg problem* - adding more data will increase frequencies of the words already present in the corpus, and, accordingly, solve the issue of the low-frequency words, but at the same time, it will introduce many extra words, where some of them were previously out-of-vocabulary. Most of these new words will now be low-frequency words - again we observe the very same Zipfian phenomenon, and the problem of low-frequency words is still present. Therefore, we have decided to take an opposite path, where "removing" data from the initial corpus (that actually means sampling a sub-corpus with less data items from the original large corpus) and properties of low-frequency words (Moore, 2004b; Prochasson and Fung, 2011) should actually help us detect correct cross-lingual word associations. By reducing the corpus size, we also decrease frequencies of the words in the corpus. In an extreme case, when the reduced corpus consists of only one sentence, almost all words in that "corpus" will occur only once or twice. Intuitively, for words with higher frequencies, one needs to remove more data, i.e., to sample a sub-corpus of smaller size, to

bring the words down to only a few occurrences in the sub-corpus. We will show that it is easier to establish translational equivalence for low-frequency words.

2.2 Criteria for Extraction of Translation Pairs

Given is a source language S, a target language T, and a corpus $\mathscr{C}$ of N aligned item pairs $\mathscr{C} = \{(I_1^S, I_1^T), (I_2^S, I_2^T), \ldots, (I_N^S, I_N^T)\}$, where, depending on the corpus type, item pairs may be sentences, paragraphs, chunks, documents, etc. For parallel corpora, the item pairs are pairs of sentences. The goal is to extract potential translation candidates from the item-aligned set using only *internal distributional evidences*. Internal evidences, according to Kay and Röscheisen (1993), represent information derived only from the given corpora themselves. Our criteria for establishing translational equivalence between words are derived from this trivial case:

Imagine the scenario where a source word w_1^S occurs only once on the source side of the corpus $\mathscr{C}$, in a source item I_j^S. There is a target word w_2^T occurring in a target item I_j^T (which is aligned to I_j^S) and the word w_2^T also occurs only once on the target side of the corpus $\mathscr{C}$. Additionally, there does not exist another source word w_a^S such that it occurs only once on the source side of the corpus and, at the same time, exactly in the item I_j^S, and there does not exist another target word w_b^T that occurs only once on the target side of the corpus and exactly in the item I_j^T. Our key assumption is that the words w_1^S and w_2^T should then be listed as translation candidates. We can further generalize the intuition, that is, two words are extracted as translation candidates if they both satisfy the entire set of features $\mathscr{F}$, and there are no other words that satisfy this set of features.[1]

The set $\mathscr{F}$ may include various clues as features, but in our work we opt only for the internal, language-independent features that are related to the distributions of words over corpora. A source word w_1^S and a target word w_2^T are listed as potential translation candidates if they fulfil the following criteria:

1. The overall frequency of w_1^S on the source side of the corpus is equal to the overall frequency of w_2^T on the target side of the corpus.
2. The overall frequency of both words is above some minimum frequency threshold M_f.
3. w_1^S and w_2^T occur only in aligned item pairs, and with exactly the same frequency.
4. The number of aligned item pairs in which the words occur is above some minimum M_i.
5. There is no source word w_a^S such that the pair (w_a^S, w_2^T) satisfies all the previous conditions, and there is no target word w_b^T such that the pair (w_1^S, w_b^T) satisfies all the previous conditions.[1]

For instance, if the French word *pluie* occurs 4 times in the whole corpus, 2 times in item I_j^S, 1 time in item I_k^S, and 1 time in item I_l^S, and there is the English word *rain* that also occurs 4 times in total, 2 times in item I_j^T, 1 time in item I_k^T, and 1 time in item I_l^T, and there are no other words with the same frequency distribution in the corpus, we claim *(pluie, rain)* to be a pair of translation candidates.

In our work, we have opted for the listed criteria/constraints, but we are free to adjust or add more criteria if we want to boost a certain behavior of the model, that is, if we want to focus

[1] This specifies one-to-one alignment constraint, but more relaxed criteria are also possible. For instance, we could allow 2 or more target words to have the same features as a source word and then distribute partial link counts over all target candidates.

more on accuracy or on coverage of the lexicon. By imposing, for instance, stricter thresholds for M_f or M_i (e.g., accepting only candidates that occur in at least two items), we can direct the algorithm for lexicon extraction towards higher accuracy, and, vice versa, by relaxing the thresholds, we boost the coverage of the lexicon.

In summary, the proposed criteria for extraction of translation candidates are not biased towards high-frequency or low-frequency words, as they treat all words the same, trying to find potential candidates according to the defined set of features. However, in practice, the majority of the matched candidates will be low-frequency words.

2.3 The Algorithm for Lexicon Extraction

By employing the aforementioned criteria for extraction of translation candidates on the initial corpus $\mathscr{C}$, we are able to extract only a limited number of translation pairs, since distributional evidences for the large corpus $\mathscr{C}$ are fixed and unchangeable. But by sampling data from $\mathscr{C}$, we actually build a new corpus, a sub-corpus $SC \subset \mathscr{C}$ of size $K < N$, which now has a changed set of distributional evidences, which may lead to extracting additional translation candidates. The process of data reduction may be observed as a process of *sampling*, i.e., we randomly pick a subset of item pairs from $\mathscr{C}$, and build a new sub-corpus SC. We can then repeat the process, sample another sub-corpus and try to detect more translation candidate pairs.

Having the large corpus $\mathscr{C}$ of a finite size N, the number of different sub-corpora is huge, but finite. The exact number of different sub-corpora that can be sampled from $\mathscr{C}$ is $M_{\mathscr{C}} = \sum_{K=1}^{N} \binom{N}{K}$. Since we are clearly unable to process all the possible sub-corpora, we need to design a smart strategy to: (1) cover the entire initial corpus and (2) detect translation pairs for both high-frequency and low-frequency words.

2.3.1 One Sampling Round with Fixed Sub-Corpora Size

Let us fix the size of sub-corpora to some value K. We want to assure that every item pair from $\mathscr{C}$ is taken into account in at least one sub-corpus of size K. Additionally, we want to be able to repeat the procedure and obtain more different sub-corpora of the same size. The procedure is as follows:

1. Initialize: Detect the number of sub-corpora for this round: $\lfloor \frac{N}{K} \rfloor$.
2. Randomly shuffle the item pairs in $\mathscr{C}$ to obtain a permutation of the item pairs in $\mathscr{C}$.
3. Split $\mathscr{C}$ into sub-corpora of equal size K as follows:
   - For $i = 1, \ldots, \lfloor \frac{N}{K} \rfloor - 1$, assign the item pairs from position $(i-1) \cdot K + 1$ until position $i \cdot K$ to the sub-corpus SC_i.
   - Assign the remaining item pairs from position $(\lfloor \frac{N}{K} \rfloor - 1) \cdot K + 1$ until the end (position N) to the sub-corpus $SC_{\lfloor \frac{N}{K} \rfloor}$.

We build a set of $\lfloor \frac{N}{K} \rfloor - 1$ sub-corpora of size K and one sub-corpus of size $K + N \bmod K$, while, at the same time, we ensure that the complete original corpus $\mathscr{C}$ is covered. We will call the described procedure the *sampling round*. If we want to repeat the procedure and acquire another set of sub-corpora of the same size, we simply go back to Step 2 of the procedure and perform another sampling round.

2.3.2 The Final Algorithm: SampLEX

Now, we have everything set for the construction of the algorithm. In order to capture words with different frequencies, we need to vary the sub-corpora size K. With respect to the Zipf's law (Prochasson and Fung, 2011), we have decided to vary the values of K from N down to 1, where K is divided by 2 in each step of the loop (see the final algorithm). In that way, we ensure that all the words occur as low-frequency words in at least some sub-corpora of various sizes. Again, if we want to reduce frequencies of high-frequency words, we need samples of smaller sizes, so such words will typically learn its translation candidates from sampled sub-corpora consisting of only a few sentences. One pass of the algorithm from the values N to 1 is called an *iteration*.

We can detect potential translation candidates in many different sub-corpora (of various sizes). Additionally, we should assign more weight to translation pairs that fulfil the strict criteria in sub-corpora of larger size K. For instance, if we detect that two words have identical frequency distributions and have fulfilled all the criteria from Subsection 2.2 in a sub-corpus consisting of a few millions items, that evidence should be more important than detecting that the two words could be extracted from a sub-corpus comprising only a few sentences. Thus, for each potential translation pair t_{ij} we assign a corresponding overall score $c_{t_{ij}}$. If we detect that the two words that form the translation pair t_{ij} could be extracted from a sub-corpus of size K, we update the score $c_{t_{ij}} := c_{t_{ij}} + 1 \cdot weight_K$, where $weight_K = \lfloor \frac{N}{K} \rfloor$. This way we assign more importance when the pairs are extracted from larger sub-corpora. For instance, if we detect that two words from the potential translation pair t_{ij} are extracted as translation candidates from the original corpus $\mathscr{C}$, then $K = N$ and $c_{t_{ij}} := c_{t_{ij}} + 1$.

The final algorithm is as follows:

1. **Input:** The initial large corpus $\mathscr{C}$ of size N.
2. **Initialize:** (1) Define the criteria for extraction of translation candidates; (2) Initialize an empty lexicon L. Each entry in the lexicon L will have the following form: $(t_{ij}, c_{t_{ij}})$, where t_{ij} denotes the extracted translation pair consisting of a source word w_i^S and a target word w_j^T, while $c_{t_{ij}}$ is a variable that denotes the score for the translation pair t_{ij}.
3. **Set** initial sub-corpora size: $K := N$.
4. **Perform** one *sampling round* with the current sub-corpora size set to K (Subsection 2.3.1). We obtain $\lfloor \frac{N}{K} \rfloor$ different sub-corpora: $SC_1, \ldots, SC_{\lfloor \frac{N}{K} \rfloor}$, all of size K except the last one (its size is always $K + N \bmod K$).
5. **Extract** translation pairs from all sub-corpora obtained in Step 4. If a translation pair t_{ij} is already present in the lexicon L, update the score $c_{t_{ij}} := c_{t_{ij}} + 1 \cdot weight_K$ for that pair. Otherwise, add the translation t_{ij} to L and set its current score $c_{t_{ij}} := 1 \cdot weight_K$.
6. **Set** new sub-corpora size: $K := \lfloor K/2 \rfloor$.
7. **If** $K > 0$, **go** to Step 4. Otherwise, we have reached the end of one *iteration* and we need to **check** the stopping criteria (**go** to Step 8).
8. **Check** the stopping criteria: **if** no new translation pairs were extracted after the end of one whole iteration **or** we have reached the maximum or the predefined number of iterations **or** timeout, **go** to Step 9. Otherwise, **go** to Step 3.
9. **Output:** The lexicon L.

We will call this procedure the **SampLEX** algorithm. The proposed algorithm exhibits only one possible strategy for mining translation pairs from sub-corpora. For instance, we could

opt for another strategy when deciding how to change the size of sub-corpora, skip already processed sub-corpora, remodel the criteria for extraction from Section 2.2, change stopping criteria, or employ a procedure for the sub-corpora sampling different from the one presented in Subsection 2.3.1. However, our main goal is to propose a general framework for lexicon extraction when the data sampling approach is employed, where other researchers could design their own algorithms built upon the same idea.

2.3.3 Properties of the Algorithm

Reducing corpora size provides several benefits. First, establishing associations between translation candidates is much easier when we deal with low-frequency words - we reduce our problem to a *binary decision problem*. According to the specified criteria for extraction, two words are simply considered to be a translation pair, or they are not. By employing the criteria that rely on raw frequency counts as distributional evidences, we remove the need of an association measure based on hypothesis testing such as the G^2 statistic (Dunning, 1993; Agresti, 2002) or a similarity-based measure such as the Dice coefficient (Dice, 1945), which are often unreliable when dealing with low-frequency words (Manning and Schütze, 1999).

The *SampLEX* algorithm is *symmetric* and *non-directional*. The final output of the algorithm provides translation pairs along with their counts obtained after training. We can easily transform them into word translation probabilities to build word translation tables similar to those of IBM Model 1. Since the algorithm is symmetric, we can obtain both source-to-target and target-to-source word translation probabilities after the algorithm run is completed:

$$P(w_2^T|w_1^S) = \frac{c_{t_{12}}}{\sum_j c_{t_{1j}}} \qquad P(w_1^S|w_2^T) = \frac{c_{t_{12}}}{\sum_j c_{t_{j2}}} \tag{1}$$

Surprisingly, another modeling advantage lies in *randomness* when selecting sub-corpora. Namely, if we detect that two words constantly co-occur in aligned items randomly sampled from the large corpus, regardless of the surrounding context, it actually strengthens the confidence that those two words really constitute a translation pair. During the sampling procedure, sentences are moved from their "natural" surrounding of other sentences (the context in this case) and the new sub-corpus is built by randomly taking sentences from the entire corpus. If the same translational equivalence between the two words is encountered in more different sub-corpora, it also raises the *significance* of that equivalence. Also, by building sub-corpora of smaller sizes from the original large corpus, we perform an implicit *disambiguation* - a word occurring only once or twice in a small sub-corpus cannot bear more meanings in that sub-corpus, although it might have more meanings in the large superset corpus.

3 Experimental Setup

In this section, we present datasets used for training, training setup of the SampLEX method and state-of-the-art models for bilingual lexicon extraction from parallel data often used in real-life applications.

3.1 Training

3.1.1 Training Collections

We work with Europarl data (Koehn, 2005) for Dutch-English and Italian-English language pairs, retrieved from the website[2] of the OPUS project (Tiedemann, 2009). We use subsets of the corpora, comprising the first 300,000 sentence pairs. For Dutch-English, there are 76,762 unique Dutch words, and 37,138 unique English words. For Italian-English, there are 68,710 unique Italian words and 37,391 unique English words. The unbalance between the number of unique vocabulary words is mostly due to a richer morphological system in Italian and the noun compounding phenomenon in Dutch.

Since we also want to test and evaluate the behavior of our system in a setting where only limited parallel data are present, we construct additional subsets of Europarl data consisting of the first 2,000, 10,000 and 50,000 sentence pairs from the corpora.

3.1.2 Training Setup of the SampLEX Method

Parameter values are set to the same values for all training datasets. We set $M_f = M_i = 0$, which means that all words that occur in a sub-corpus at least once may be extracted. By setting some higher thresholds M_f and M_i, we could move the algorithm towards extracting lexicons of higher accuracy, but lower coverage. We stop our training procedure for *SampLEX* after 1000 iterations for all corpora. The SampLEX algorithm converges quickly - many translations are found in the first few iterations. However, having more iterations implies obtaining more different evidences from different sub-corpora and assigning more significance for the extracted candidates (see Subsection 2.3.3). Therefore, we have decided to use 1000 iterations for safety. Other stopping criteria are also possible (see Step 8 in Subsection 2.3.2).

3.2 State-of-the-Art Models for BLE

In order to evaluate the performance of our *SampLEX* algorithm for bilingual lexicon extraction, we compare it with other models that constitute state-of-the-art for BLE, and are often used in real-life applications (see Section 1).

3.2.1 IBM Model 1

Our first baseline is IBM Model 1 (Brown et al., 1993) for word alignment, which is a purely lexical model, i.e, the only set of parameters employed by the model are word translation probabilities. We omit the exact generative story for IBM Model 1, but the curious reader may find all the details in (Brown et al., 1993) or (Och and Ney, 2003). Word translation probability $P(w_2^T|w_1^S)$ denotes a probability that a source word w_1^S generates a target word w_2^T. These probabilities can then be used to decide upon translational equivalence between words and to build bilingual lexicons from parallel texts.[3] That makes it comparable to our *SampLEX* model, which can also output word translation probabilities (Equation 1). IBM Model 1 is used in many systems as a primary tool for bilingual lexicon extraction from parallel data (e.g., Venugopal et al. (2003), Munteanu and Marcu (2005), Munteanu and Marcu (2006), Lefever et al. (2009)). We use standard GIZA++ settings and train IBM Model 1 with 5 iterations

[2] http://opus.lingfil.uu.se/Europarl3.php

[3] We have also tried to use word translation probabilities from the higher order IBM Models, but we have not detected any major difference in results on the task of bilingual word lexicon extraction.

(**IBM1-i5**) and 20 iterations (**IBM1-i20**) of the EM algorithm, as often found in the literature (Och and Ney, 2003; Moore, 2004a).

3.2.2 The Dice Coefficient

Another baseline model is a similarity-based model relying on the Dice coefficient (**DICE**):

$$DICE(w_1^S, w_2^T) = \frac{2 \cdot C(w_1^S, w_2^T)}{C(w_1^S) + C(w_2^T)} \tag{2}$$

where $C(w_1^S, w_2^T)$ denotes the co-occurrence count of words w_1^S and w_2^T in the aligned items from the corpus. $C(w_1^S)$ and $C(w_2^T)$ denote the count of w_1^S on the source side of the corpus, and the count of w_2^T on the target side of the corpus, respectively. The Dice coefficient was used as an associative method for word alignment by Och and Ney (2003), Tiedemann (2003) used it as one associative clue for his clue-based word alignment, and Melamed (2000) used it to measure the strength of translational equivalence.

3.2.3 Log-Likelihood Ratio

Another associative model that we use is based on the log-likelihood-ratio (**LLR**), that is derived from the G^2 statistic (Dunning, 1993). LLR is a more appropriate hypothesis testing method for detecting word associations from limited data than the χ^2 test (Manning and Schütze, 1999) and was previously used as an effective tool for automatically constructing bilingual lexicons (Melamed, 2000; Moore, 2001; Munteanu and Marcu, 2006). Its definition is easily explained on the basis of a contigency table (Kilgarriff, 2001; Padó and Lapata, 2007), which is a four-cell matrix for each pair of words (w_1^S, w_2^T) (see Table 1).

	w_1^S	$\neg w_1^S$
w_2^T	k	l
$\neg w_2^T$	m	n

Table 1: The contigency table for a pair of words (w_1^S, w_2^T).

The contingency table records that source word w_1^S and target word w_2^T co-occur in k aligned item/sentences pairs, and w_1^S occurs in m aligned pairs in which w_2^T is not present. Similarly, w_2^T occurs in l aligned pairs in which w_1^S is not present, and n is the number of aligned pairs that involve neither w_1^S nor w_2^T. The final formula for the log-likelihood ratio is then defined as:

$$\begin{aligned} LLR(w_1^S, w_2^T) = G^2(k,l,m,n) = 2(&k\log k + l\log l + m\log m + n\log n \\ &-(k+l)\log(k+l) - (k+m)\log(k+m) \\ &-(l+n)\log(l+n) - (m+n)\log(m+n) \\ &+(k+l+m+n)\log(k+l+m+n)) \end{aligned} \tag{3}$$

High LLR scores can indicate either a positive association or a negative one (Moore, 2004b). Since we expect translation pairs to be positively associated, we impose an additional constraint: $P(w_1^S, w_2^T) > P(w_1^S) \cdot P(w_2^T)$, where $P(w_1^S, w_2^T) = \frac{k}{k+l+m+n}$, $P(w_1^S) = \frac{k+m}{k+l+m+n}$ and $P(w_2^T) = \frac{k+l}{k+l+m+n}$. This constraint keeps only positively associated words in the lists of potential translation candidates.

3.3 Evaluation Methodology

3.3.1 Lists of Ground Truth Translation Pairs

In order to evaluate the BLE models, we have designed a set of ground truth translations - we have randomly sampled a set of Dutch content words that occur in the full corpus comprising 300, 000 sentences. Following that, we have used the *Google Translate* tool plus an additional annotator to translate those words to English. The annotator has manually revised the lists and has kept only words that have their corresponding translation in the English vocabulary. In order to build a one-to-one ground truth dataset of translations, only one possible translation has been annotated as correct. In case when more than 1 translation is possible, the annotator has marked as correct the translation that occurs more frequently in the English Europarl data. Finally, we have come up with a set of 1001 ground truth one-to-one translation pairs. We have followed the same procedure for Italian-English and have also constructed a set of 1001 ground truth translation pairs.[4]

3.3.2 Evaluation Metrics

All the methods under consideration actually retrieve ranked lists of translation candidates. Let us keep only the first translation candidate from each ranked list, and build a non-probabilistic lexicon of one-to-one word translations: L_e. Assuming that we now have a set G of ground truth one-to-one word translation pairs, we can evaluate the quality of our lexicon with respect to the ground truth set G. We use standard precision, recall and F-Measure ($\beta = 1$) scores as our evaluation metrics:

$$Prec_{L_e,G} = \frac{|L_e \cap G|}{|L_e|} \qquad Rec_{L_e,G} = \frac{|L_e \cap G|}{|G|} \qquad F_{L_e,G} = (1+\beta^2)\frac{Prec_{L_e,G} \cdot Rec_{L_e,G}}{\beta^2 \cdot Prec_{L_e,G} + Rec_{L_e,G}}$$

Since sometimes a word has more than one correct translation (e.g., Dutch word *verklaring* can be translated as both *statement* and *declaration*), and the current evaluation setting cannot capture that phenomenon, we also evaluate the quality of the lexicon in a more lenient setting, where, instead of performing the hard cut-off, i.e., instead of keeping only the top candidate from the ranked list, we keep the ranked list of all the candidates from the list and calculate the *mean reciprocal rank* (MRR) (Voorhees, 1999). For a source word w_i^S, $rank(w_i^S)$ denotes the rank of its correct translation (as provided by the set of ground truth translation pairs) within the retrieved list of potential translation candidates. MRR of the lexicon is then defined by the following formula:

$$MRR_{L_e,G} = \frac{1}{|L_e|} \sum_{w_i^S \in L_e} \frac{1}{rank(w_i^S)} \tag{4}$$

4 Results and Discussion

We conduct several experiments to measure the quality of the lexicon constructed using the *SampLEX* algorithm: (1) we evaluate the lexicon obtained by SampLEX using the full corpus of 300, 000 sentences, and compare its accuracy with the accuracy of baseline systems from Section 3.2 trained on the same corpus, (2) after performing the error analysis, we carry out another set of experiments that prove that the *SampLEX* algorithm, due to its modeling

[4] We will make the datasets publicly available.

properties, alleviates the problem of *indirect associations* and, finally, (3) we test our lexicon in a setting where only limited parallel data are available and show that the *SampLEX*-based lexicon outperforms other bilingual word lexicons in that setting in terms of quality provided by the F-measure and precision scores.

4.1 Experiment I: Testing the Quality of the Lexicon in Terms of Precision

Unlike our baseline state-of-the-art systems for BLE, the *SampLEX* algorithm does not assure the full coverage of the source vocabulary, as it does not necessarily build ranked lists of translation candidates for all the words observed during training. However, our claim is that translation pairs obtained by *SampLEX* are of higher quality than those obtained by the baseline systems. Therefore, with this experiment we want to answer the following question: "Are translation pairs obtained by the SampLEX algorithm really more accurate than translation pairs obtained by other methods?". In order to answer that question, we calculate the precision and MRR scores on our ground truth datasets for Italian-English and Dutch-English, where all the BLE methods have been trained on the full 300,000 datasets. The obtained scores are presented in Tables 2 and 3.

	Dutch-English				
	IBM1-i5	**IBM1-i20**	**DICE**	**LLR**	**SampLEX**
Prec(300k)	0.7113	0.7023	0.6963	0.7662	**0.8221**
MRR(300k)	0.8196	0.8045	0.7767	0.8542	**0.9069**

Table 2: Precision and MRR scores for all models trained on the first 300,000 sentences of Dutch-English Europarl data, and evaluated on the sets of 1001 ground truth translation pairs for Dutch-English.

	Italian-English				
	IBM1-i5	**IBM1-i20**	**DICE**	**LLR**	**SampLEX**
Prec(300k)	0.7912	0.7752	0.7932	0.8361	**0.8771**
MRR(300k)	0.8781	0.8588	0.8494	0.8945	**0.9250**

Table 3: Precision and MRR scores for all models trained on the first 300,000 sentences of Italian-English Europarl data, and evaluated on the sets of 1001 ground truth translation pairs for Italian-English.

As previously shown by Moore (2004a), LLR serves as a better associative method than the Dice coefficient for the word alignment task. We obtain the same finding for bilingual lexicon extraction. Additionally, the model based on LLR is also better than IBM Model 1 when applied for BLE. Munteanu and Marcu (2006) drew the same conclusion, and they used the LLR-based lexicon in their system when a higher precision of the lexicon was paramount. However, the results reveal that the quality of the lexicon obtained by the *SampLEX* algorithm is superior to the LLR-lexicon *in terms of precision* and, consequently, to all other evaluated lexicons.

4.2 Experiment II: Investigating Indirect Associations

When examining the results, we have detected that one advantage of our SampLEX algorithm is due to its mitigating the phenomenon of the so-called indirect associations. Indirect associations,

as defined by Melamed (2000), are associations between words that have a tendency to co-occur much more often than expected by chance, but are not mutual translations. Lexicon extraction models unaware of the indirect associations tend to give translational preference to higher-frequency words. Considering the fact that one key assumption of our model is sub-corpora sampling that causes decreasing frequencies of words in the obtained sub-corpora from which translation pairs are learned, our model should successfully mitigate the problem of indirect associations. Indeed, during the error analysis, we have detected that both IBM Model 1 and LLR provide a wrong translation of the Dutch word *beschouwen (consider)*, since both models retrieve the English word *as* as the first translation candidate (due to a very high frequency of the collocation *consider as*). Other examples of the same type include the Dutch word *integreren (integrate)* which is translated as *into*, *betwijfelen (doubt)* which is translated as *whether*, or an Italian example of the verb *entrare (enter)* which is translated as *into*. Our BLE model, on the other hand, provides correct translations for all these examples. Dagan et al. (1993) noted that collocates often tend to cause confusion among algorithms for bilingual lexicon extraction. More examples include the Dutch word *opinie (opinion)*, translated as *public* by IBM Model 1 and LLR (due to a high frequency of the collocation *public opinion*), the Dutch word *cirkels (circles)*, translated as *concentric*, or the Italian word *pensionabile (pensionable)*, translated as *age*. All these examples are again correctly translated by our model for lexicon extraction.

In order to test the hypothesis that our lexicon extraction model does not suffer from the problem of learning indirect associations, we have conducted a small experiment. For the purpose of the evaluation, we have constructed a small dataset of 219 Italian verbs in first person plural of the present tense. We have also constructed the set of ground truth translations in the same way as in Subsection 3.3.1. These verbs are easy to extract because they all have the same suffix *-iamo* (e.g., the verb *respiriamo*, meaning *(we) breathe*). If the problem of indirect associations for a lexicon extraction method is prominent, the English word *we* will appear as the first translation for many of these verbs, instead of the word that really bears the content of the verb (e.g., *breathe*). Table 4 shows precision and MRR scores for the lexicon extraction models evaluated on this toy dataset. As expected, due to its modeling property related to the

	IBM1-i5	**DICE**	**LLR**	**SampLEX**
Prec(300k)	0.4475	0.4201	0.6119	**0.8584**
MRR(300k)	0.5575	0.5108	0.7300	**0.9140**

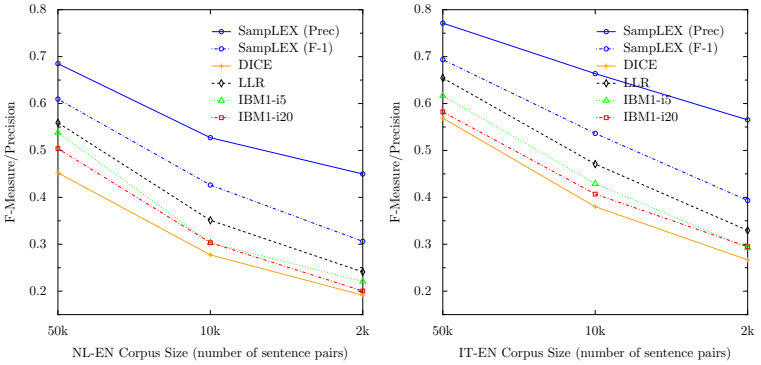
Table 4: Precision and MRR scores on our evaluation set consisting of Italian *-iamo* verbs (present tense, first person plural).

reduction of word frequencies, our BLE model does not suffer from the problem of indirect associations like other models. That property eventually has a positive impact on precision and MRR scores and the overall quality of the lexicons obtained by our SampLEX algorithm.

4.3 Experiment III: Experiments with a Limited Amount of Parallel Data

In a real-life situation, one often possesses only limited parallel data (e.g., terminology texts from special, very narrow domains and sub-domains). With this final set of experiments we test the performance of all the models for lexicon extraction in such a setting with limited parallel data. To simulate the shortage of data, we have extracted three additional corpora of smaller sizes by selecting the first 2,000, 10,000 and 50,000 sentence pairs from our Dutch-English and Italian-English Europarl data. From our initial ground truth set (Subsection 3.3.1), we

have only kept the words that occur at least once in the respective corpora as ground truth for evaluations (e.g., there are 444 words in the ground truth dataset for the corpus consisting of the first Dutch-English 2,000 sentence pairs, and 931 words for the corpus consisting of the first 50,000 Dutch-English sentence pairs). Our question is now: "Are lexicons extracted by SampLEX really of better quality than lexicons obtained by other methods when dealing with parallel corpora of limited size?" As mentioned before, the *SampLEX* algorithm does not have a property to provide results in a form of ranked lists for the entire source vocabulary, but we claim that *SampLEX* is directed towards extracting only highly reliable and precise candidates which, consequently, leads to lexicons of a higher quality. That claim is again supported by the findings presented in Figure 1(a) for Dutch-English, and in Figure 1(b) for Italian-English.



(a) Dutch to English (b) Italian to English

Figure 1: Precision and F-Measure scores over parallel corpora of different size (2k, 10k and 50k aligned sentence pairs). Since *SampLEX* does not necessarily obtain the lists of translations for all words in a vocabulary, its precision scores are different than its F-measure scores. For all other models within this evaluation setting, it is valid: *Precision=Recall=F-measure*.

We have also performed an additional experiment to test whether the translation candidates for Dutch and Italian words that happen to be retrieved by the *SampLEX* algorithm still display better overall precision and MRR scores than the translation candidates for the same Dutch and Italian words obtained by the other methods. If that is not true, we could use *SampLEX* only to extract source words for which a translation might be found, but the particular translation for each extracted word could then be obtained by some other method. However, it is not the case, as the results in Tables 5 and 6 reveal. As noted in the literature (Manning and Schütze, 1999), we observe that, of all the baseline models for BLE, LLR suffers the least from data sparsity, but still performs worse than our method.

Since *SampLEX* is built on the concept of data sampling, the criteria for extracting translation candidates and the whole training process inherently remain the same when working with parallel corpora of limited size. However, it is natural that the results decrease when the size of the large corpus $\mathscr{C}$ decreases. The more data we possess, the more sub-corpora we can sample, which finally provides better chances to extract correct translation pairs. We could say

	Dutch-English				
	IBM1-i5	**IBM1-i20**	**DICE**	**LLR**	**SampLEX**
Prec(2k)	0.3668	0.3624	0.3319	0.4323	**0.4498**
MRR(2k)	0.4206	0.4199	0.3968	0.4498	**0.4836**
Prec(10k)	0.4266	0.4306	0.3682	0.4889	**0.5272**
MRR(10k)	0.5071	0.5039	0.4513	0.5587	**0.5848**
Prec(50k)	0.6180	0.5952	0.5295	0.6621	**0.6850**
MRR(50k)	0.7067	0.6901	0.6183	0.7182	**0.7429**

Table 5: Precision and MRR scores for all models trained on the subsets of different sizes (2k, 10k and 50k sentences) from Dutch-English Europarl data. Only candidates retrieved by *SampLEX* have been taken into account for this evaluation.

	Italian-English				
	IBM1-i5	**IBM1-i20**	**DICE**	**LLR**	**SampLEX**
Prec(2k)	0.5087	0.5174	0.4348	0.5521	**0.5652**
MRR(2k)	0.5798	0.5778	0.4897	**0.6113**	0.6079
Prec(10k)	0.5978	0.5846	0.5011	0.6461	**0.6637**
MRR(10k)	0.6556	0.6489	0.5709	0.6914	**0.7014**
Prec(50k)	0.7129	0.6966	0.6381	0.7578	**0.7714**
MRR(50k)	0.7926	0.7847	0.7186	0.8064	**0.8278**

Table 6: Precision and MRR scores for all models trained on the subsets of different sizes (2k, 10k and 50k sentences) from Italian-English Europarl data. Only candidates retrieved by *SampLEX* have been taken into account for this evaluation.

that *SampLEX* takes the best of both worlds - it benefits from the idea of data reduction, yet it provides better scores when more input data are available.

5 Conclusions and Future Work

In this paper, we have proposed a statistical framework for the construction of a bilingual word lexicon built upon the idea of sampling many smaller sub-corpora from an initial larger item-aligned corpus.

The *SampLEX* algorithm for bilingual lexicon extraction presented in the paper is directed towards extraction of only highly reliable word translation pairs. After comparisons with other models for BLE from parallel data, we have proved that *SampLEX* builds word lexicons of higher accuracy and overall quality as revealed by the F-measure and precision scores, which is especially important in a setting where only a limited amount of parallel data is available. The proposed framework allows for many further experimentations and possible applications. The description of the framework provided in the paper is generic - it is language-independent and applicable to any corpus that provides some sort of alignment (at the sentence, paragraph or document level). In future work, we plan to design algorithms for mining word translations from comparable corpora based on the similar idea of sub-corpora sampling.

Acknowledgments

The research has been carried out in the framework of the TermWise Knowledge Platform (IOF-KP/09/001) funded by the Industrial Research Fund, KU Leuven, Belgium.

References

Agresti, A. (2002). *Categorical Data Analysis, 2nd Edition*. Wiley.

Andrade, D., Nasukawa, T., and Tsujii, J. (2010). Robust measurement and comparison of context similarity for finding translation pairs. In *Proceedings of the 23rd International Conference on Computational Linguistics (COLING)*, pp. 19–27.

Brown, P. F., Pietra, V. J. D., Pietra, S. A. D., and Mercer, R. L. (1993). The mathematics of statistical machine translation: parameter estimation. *Computational Linguistics*, 19(2):263–311.

Carbonell, J. G., Yang, J. G., Frederking, R. E., Brown, R. D., Geng, Y., Lee, D., Frederking, Y., E, R., Geng, R. D., and Yang, Y. (1997). Translingual information retrieval: A comparative evaluation. In *Proceedings of the 15th International Joint Conference on Artificial Intelligence (IJCAI)*, pp. 708–714.

Church, K. W. and Mercer, R. L. (1993). Introduction to the special issue on computational linguistics using large corpora. *Computational Linguistics*, 19(1):1–24.

Dagan, I., Church, K. W., and Gale, W. A. (1993). Robust bilingual word alignment for machine aided translation. In *Proceedings of the Workshop on Very Large Corpora*, pp. 1–8.

Diab, M. T. and Finch, S. (2000). A statistical translation model using comparable corpora. In *Proceedings of the 6th Triennial Conference on Recherche d'Information Assistée par Ordinateur (RIAO)*, pp. 1500–1508.

Dice, L. R. (1945). Measures of the amount of ecologic association between species. *Ecology*, 26(3):297–302.

Dunning, T. (1993). Accurate methods for the statistics of surprise and coincidence. *Computational Linguistics*, 19(1):61–74.

Fung, P. and Cheung, P. (2004). Mining very-non-parallel corpora: Parallel sentence and lexicon extraction via bootstrapping and EM. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 57–63.

Fung, P. and Yee, L. Y. (1998). An IR approach for translating new words from nonparallel, comparable texts. In *Proceedings of the 17th International Conference on Computational Linguistics (COLING)*, pp. 414–420.

Haghighi, A., Liang, P., Berg-Kirkpatrick, T., and Klein, D. (2008). Learning bilingual lexicons from monolingual corpora. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics (ACL)*, pp. 771–779.

Kay, M. and Röscheisen, M. (1993). Text-translation alignment. *Computational Linguistics*, 19(1):121–142.

Kilgarriff, A. (2001). Comparing corpora. *International Journal of Corpus Linguistics*, 6(1):1–37.

Koehn, P. (2005). Europarl: A parallel corpus for statistical machine translation. In *Proceedings of the Machine Translation Summit 2005*, pp. 79–86.

Laroche, A. and Langlais, P. (2010). Revisiting context-based projection methods for term-translation spotting in comparable corpora. In *Proceedings of the 23rd International Conference on Computational Linguistics (COLING)*, pp. 617–625.

Lefever, E., Macken, L., and Hoste, V. (2009). Language-independent bilingual terminology extraction from a multilingual parallel corpus. In *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics (EACL)*, pp. 496–504.

Levow, G.-A., Oard, D. W., and Resnik, P. (2005). Dictionary-based techniques for cross-language information retrieval. *Information Processing and Management*, 41:523–547.

Manning, C. D. and Schütze, H. (1999). *Foundations of Statistical Natural Language Processing*. MIT Press, Cambridge, MA, USA.

Melamed, I. D. (2000). Models of translational equivalence among words. *Computational Linguistics*, 26:221–249.

Mihalcea, R. and Pedersen, T. (2003). An evaluation exercise for word alignment. In *Proceedings of the HLT-NAACL 2003 Workshop on Building and Using Parallel Texts: Data Driven Machine Translation and Beyond*, pp. 1–10.

Moore, R. C. (2001). Towards a simple and accurate statistical approach to learning translation relationships among words. In *Proceedings of the Workshop on Data-Driven Methods in Machine Translation*, pp. 1–8.

Moore, R. C. (2002). Fast and accurate sentence alignment of bilingual corpora. In *Proceedings of the 5th Conference of the Association for Machine Translation in the Americas on Machine Translation: From Research to Real Users (AMTA)*, pp. 135–144.

Moore, R. C. (2004a). Improving IBM word-alignment Model 1. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics (ACL)*, pp. 518–525.

Moore, R. C. (2004b). On log-likelihood-ratios and the significance of rare events. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 333–340.

Morin, E., Daille, B., Takeuchi, K., and Kageura, K. (2007). Bilingual terminology mining - using brain, not brawn comparable corpora. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics (ACL)*, pp. 664–671.

Munteanu, D. S. and Marcu, D. (2005). Improving machine translation performance by exploiting non-parallel corpora. *Computational Linguistics*, 31:477–504.

Munteanu, D. S. and Marcu, D. (2006). Extracting parallel sub-sentential fragments from non-parallel corpora. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th Annual Meeting of the Association for Computational Linguistics (COLING-ACL)*, pp. 81–88.

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

Padó, S. and Lapata, M. (2007). Dependency-based construction of semantic space models. *Computational Linguistics*, 33(2):161–199.

Pinto, D., Civera, J., Barrón-Cedeòo, A., Juan, A., and Rosso, P. (2009). A statistical approach to crosslingual natural language tasks. *Journal of Algorithms*, 64(1):51–60.

Prochasson, E. and Fung, P. (2011). Rare word translation extraction from aligned comparable documents. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL-HLT)*, pp. 1327–1335.

Rapp, R. (1995). Identifying word translations in non-parallel texts. In *Proceedings of the 33rd Annual Meeting of the Association for Computational Linguistics (ACL)*, pp. 320–322.

Rapp, R. (1999). Automatic identification of word translations from unrelated English and German corpora. In *Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics (ACL)*, pp. 519–526.

Resnik, P. and Smith, N. A. (2003). The Web as a parallel corpus. *Computational Linguistics*, 29:349–380.

Shezaf, D. and Rappoport, A. (2010). Bilingual lexicon generation using non-aligned signatures. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics (ACL)*, pp. 98–107.

Tamura, A., Watanabe, T., and Sumita, E. (2012). Bilingual lexicon extraction from comparable corpora using label propagation. In *Proceedings of the Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pp. 24–36.

Tiedemann, J. (2003). Combining clues for word alignment. In *Proceedings of the 10th Conference of the European Chapter of the Association for Computational Linguistics (EACL)*, pp. 339–346.

Tiedemann, J. (2009). News from OPUS - A collection of multilingual parallel corpora with tools and interfaces. In *Recent Advances in Natural Language Processing (RANLP)*, pp. 237–248.

Ueffing, N. and Ney, H. (2007). Word-level confidence estimation for machine translation. *Computational Linguistics*, 33(1):9–40.

Venugopal, A., Vogel, S., and Waibel, A. (2003). Effective phrase translation extraction from alignment models. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics (ACL)*, pp. 319–326.

Voorhees, E. M. (1999). The TREC-8 question answering track report. In *Proceedings of the Eighth TExt Retrieval Conference (TREC-8)*.

Vulić, I., De Smet, W., and Moens, M.-F. (2011). Identifying word translations from comparable corpora using latent topic models. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL-HLT)*, pp. 479–484.

Vulić, I. and Moens, M.-F. (2012). Detecting highly confident word translations from comparable corpora without any prior knowledge. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics (EACL)*, pp. 449–459.

The Utility of Discourse Structure in Identifying Resolved Threads in Technical User Forums

Li Wang[1,2] *Su Nam Kim*[3] *Timothy Baldwin*[1,2]
(1) Dept. of Computing and Information Systems, The University of Melbourne
(2) NICTA Victoria Research Laboratory
(3) Faculty of Information Technology, Monash University
li@liwang.info, sunamkim@gmail.com, tb@ldwin.net

ABSTRACT
Online discussion forums are a valuable means for users to resolve specific information needs, both interactively for the participants and statically for users who search/browse over historical thread data. However, the complex structure of forum threads can make it difficult for users to extract relevant information. Automatically identifying whether the problem in a thread has been solved or not can help direct users to threads where the original problem has been solved, hence enhancing their prospects of solving their particular problem. In this paper, we investigate the task of Solvedness classification by exploiting the discourse structure of forum threads. Experimental results show that simple features derived from thread discourse structure can greatly boost the accuracy of Solvedness classification, which has been shown to be very difficult in previous research.

KEYWORDS: Discourse Structure, Web User Forum, Social Media, Dialogue Act.

Proceedings of COLING 2012: Technical Papers, pages 2739–2756,
COLING 2012, Mumbai, December 2012.

1 Introduction

Web user forums (or simply "forums") are online platforms for people to discuss information and obtain information via a text-based threaded discourse, generally in a pre-determined domain (e.g. IT support or DSLR cameras). With the advent of Web 2.0, there has been an explosion of web authorship in this area, and forums are now widely used in various areas such as customer support, community development, interactive reporting and online eduction. In addition to providing the means to interactively participate in discussions or obtain/provide answers to questions, the vast volumes of data contained in forums make them a valuable resource for "support sharing", i.e. looking over records of past user interactions to potentially find an immediately applicable solution to a current problem. On the one hand, more and more answers to questions over a wide range of domains are becoming available on forums; on the other hand, it is becoming harder and harder to extract and access relevant information due to the sheer scale and diversity of the data.

In the domain of troubleshooting-oriented forums, one potential way to enhance information access and support sharing is to automatically identify threads where the original information need has been resolved. By filtering out threads which do not contain a valid answer, we can focus the attention of users on threads which have a greater chance of containing the required solution. Baldwin et al. (2007) explore this task of Solvedness classification, and find that it is an extremely difficult problem. Figure 1 shows an example thread, made up of 5 posts from 3 distinct participants, from the ILIAD (Improved Linux Information Access by Data Mining) dataset of Baldwin et al. (2007). In this thread, Post1 and Post3 are both from the thread's initiator UserA. Post1 asks a question, and Post3 asks for more information about an answer provided by UserB in Post2. In response to Post3, UserB adds more information to his/her original answer, and Post5 provides another independent answer. In threads like this, it is important to identify whether the problem is solved or not, and also where solution(s) are likely to be found.

This research proposes to use information derived from thread discourse structure (Kim et al., 2010b; Wang et al., 2011) to help predict Solvedness of threads, without validating the answers provided in the threads. The discourse structure of the thread is modelled as a rooted Directed Acyclic Graph (DAG), and each post in the thread is represented as a node in this DAG. The reply-to relations between posts are then denoted as direct edges (Links) between nodes in the DAG, and the type of a reply-to relation is defined as Dialogue Act (DA). The Link between two connected posts (i.e. having a reply-to relation) is represented as the distance between the two posts in their chronological ordering. In the annotated version of the example ILIAD thread, as is shown in Figure 1, UserA initiates the thread with a question (Dialogue Act = Question-question: (Kim et al., 2010b)) in the first post, by asking a question. In response, UserB provides an answer (Dialogue Act = Answer-answer). Then, UserA confirms more details about the answer provided (Dialogue Act = Answer-confirmation). UserB responds to UserA to add more information about his/her previous answer (Dialogue Act = Answer-add). Finally, UserC proposes an independent answer again to the original question (Dialogue Act = Answer-answer).

Specifically, we explore features extracted from the thread discourse structure which can be used to help classify the Solvedness of threads. We experiment with both gold-standard and automatically predicted discourse structure, and find that thread discourse structure (which in no way evaluates the correctness of each post) can, indeed, boost thread classification accuracy,

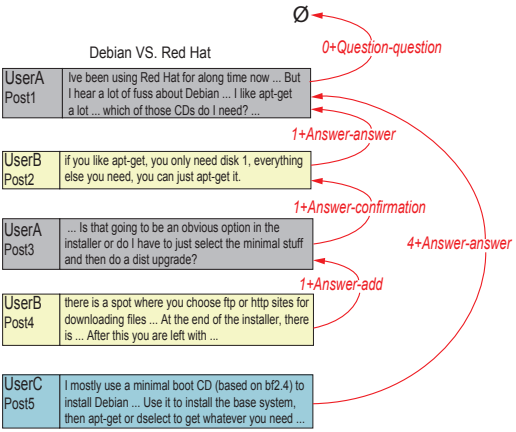


Figure 1: A snippeted ILIAD thread with annotated discourse structure.

achieving state-of-the-art results over the task. We also investigate the correlation between thread discourse structure prediction F-score and thread Solvedness classification accuracy, and demonstrate a positive correlation. Finally, we show that focusing on improving the F-score over certain dialogue acts is able to boost Solvedness classification.

2 Related Work

As far as we are aware, there is very little NLP work that is specifically targeted at the thread-level analysis of web user forum data. The most closely-related work is that performed by Baldwin et al. (2007), on which this work is directly based. Baldwin et al. (2007) focused on three specific characteristics detected from forum threads, namely *Task orientation* (i.e. whether a thread focuses on solving a problem), *Completeness* (i.e. whether the initial post provides enough information of the problem) and *Solvedness* (i.e. whether the problem is solved). Three classification tasks were identified base on these three characteristics, and experiments were carried out using a range of classification and regression methods. Baldwin et al. (2007) explored not only bag-of-words features, but also another 18 lexical and contextual features from distinct partitions of the thread, namely *initial posts*, *first response post*, *all responses posts* and *final post from the initiator*. 250 threads with various topics from a Linux-related forum and mailing list were annotated for use in the experiments in the paper. While the experimental results illustrated the difficulties in performing the three tasks automatically, the experiments also implied that their approach could be employed to rank threads based on their characteristics.

Research on thread discourse structure analysis and classification over user forums has gained in momentum in recent years. Fortuna et al. (2007) defined 5 post-level dialogue acts to describe the levels of agreement (i.e. agreement, disagreement, insult) and identify questions and answers (i.e. question and answer) in forum posts. Xi et al. (2004) defined 5 prevalent types of post-level dialogue acts in forum threads. This set of dialogue acts was then adapted

and extended by Kim et al. (2010b) to describe possible types of posts in troubleshooting-oriented online forums. Specifically, Kim et al. (2010b) devised a post-level dialogue act set and annotated a set of threads taken from CNET.[1] In this work, they proposed a set of novel features, which they applied to the separate tasks of post link classification and dialogue act classification. They later applied the same basic methodology to dialogue act classification over one-on-one live chat data with provided message dependencies (Kim et al., 2010a), demonstrating the generalisability of the original method. In both cases, however, they tackled only a single task, either link classification (optionally given dialogue act tags) or dialogue act classification, but never the two together.

Wang et al. (2011) delved into the task of thread discourse structure parsing further. They used the same features as Kim et al. (2010b), but different parsing approaches. Specifically, Wang et al. (2011) approached thread discourse structure parsing as a joint link and dialogue act classification task, by using CRFSGD (Bottou, 2011) and MaltParser (Nivre et al., 2007). They also demonstrated that the methods they use for thread discourse structure parsing are able to perform equally well over partial threads as complete threads, by experimenting with "in situ" classification of evolving threads.

There is also research focusing on particular types of dialogue acts, such as question–answer pairs in emails (Shrestha and McKeown, 2004) and forum threads (Cong et al., 2008), question–context–answer in forum threads (Cong et al., 2008; Ding et al., 2008; Cao et al., 2009), initiation–response pairs (e.g. question–answer, assessment–agreement, and blame–denial) in forum threads (Wang and Rosé, 2010), as well as request and commitment in emails (Lampert et al., 2007, 2008a,b, 2010).

Thread discourse structure can be used to facilitate different tasks in web user forums. For example, threading information has been shown to enhance retrieval effectiveness for post-level retrieval (Xi et al., 2004; Seo et al., 2009), thread-level retrieval (Seo et al., 2009; Elsas and Carbonell, 2009), sentence-level shallow information extraction (Sondhi et al., 2010), and near-duplicate thread detection (Muthmann et al., 2009). Moreover Wang and Rosé (2010) demonstrated that initiation–response pairs (e.g. question–answer, assessment–agreement, and blame–denial) from online forums have the potential to enhance thread summarisation and automatically generate knowledge bases for Community Question Answering (cQA) services such as Yahoo! Answers. Furthermore, Kim et al. (2006) showed that dialogue acts can be used to classify student online discussions in web-enhanced courses. Specifically, they use dialogue acts to identify discussion threads that may have unanswered questions and need the attention of an instructor.

These previous research efforts suggest that the thread structural representation used in Wang et al. (2011), which includes both linking structure and the dialogue act associated with each link, could potentially provide even greater leverage in these tasks. We specifically target the thread-level task of Solvedness classification as it is conceptually the most difficult of the three classification tasks proposed by Baldwin et al. (2007), and intuitively, it is the task which stands to gain the most from discourse parsing.

[1] http://www.csse.unimelb.edu.au/research/lt/resources/conll2010-thread/

3 Data Description

We use ILIAD (Improved Linux Information Access by Data Mining) data set created by Baldwin et al. (2007), which contains threads crawled from Linuxquestions[2] and Debian mailing lists.[3] The ILIAD dataset is made up of 250 threads, which are annotated with three Boolean thread-level labels, namely *Task Orientation*, *Completeness* and *Solvedness*. This paper only explores the classification task of Solvedness, which addresses the question of "is there a documented solution to the original problem described by the thread initiator in the thread (including the possibility of URLs pointing off to solutions elsewhere on that same forum or generally on the web)?" A detailed description of the other tasks and the dataset is presented in Baldwin et al. (2007).

To explore the task of using discourse structure to predict the Solvedness of a thread, we annotated the ILIAD threads for discourse structure based on the dialogue act set proposed by Kim et al. (2010b). The dialogue act set is made up of 5 super-categories: Question, Answer, Resolution, Reproduction and Other. The Question category contains 4 sub-classes: question, add, confirmation and correction. Similarly, the Answer category contains 5 sub-classes: answer, add, confirmation, correction and objection. For example, the label Question-add signifies the Question superclass and add subclass, i.e. addition of extra information to a question. For full details of the original dialogue act tagset, see Kim et al. (2010b).

The original dialogue act set was developed primarily over troubleshooting-oriented threads (i.e. the CNET dataset described in Section 2), however there are non-troubleshooting threads present in the ILIAD dataset (hence the *Task Orientation* thread classification task is addressed in Baldwin et al. (2007)). After manual analysis of the ILIAD data, we identified that the dialogue act tagset was largely transferable in its original state, but needed the addition of the information sub-class to the Question super-category (i.e. Question-information). Question-information is used on posts in threads which are not troubleshooting-oriented and only provide information (e.g. on developer mailing lists to report on a bug fix). We also relaxed the definition of Resolution slightly to accommodate non-troubleshooting threads. For example, in one thread the initiator requests an update to a wiki page, and this update is confirmed later by a non-initiator. In this case, this non-initiator's post is labelled as Resolution. In the original definition, Resolution can only be used on posts from the initiator of the thread.

The modified dialogue acts (DAs) used to annotate the ILIAD dataset for discourse structure are described in Table 1. The annotation was performed by two annotators. The main annotator annotated all 250 threads (containing 1158 posts), and the secondary annotator independently annotated 26 randomly-selected threads (containing 113 posts) for quality assurance purposes. During annotation, annotators first annotate the Links between posts in a thread, and then identify the type of each link (DA). The κ values for agreement between the two annotators are 0.64 for combined Link and DA tagging, 0.79 for just the Links and 0.68 for just the DAs.

While both the ILIAD and CNET datasets are mainly troubleshooting-oriented and technical, they come from different domains. Therefore, we expect the DA and Link distributions of them to be different. However, to our surprise, the distributions of both DAs and Links in the two datasets are remarkably similar, supporting the suggestion that the DA label set has cross-domain applicability.

[2] http://www.linuxquestions.org
[3] http://lists.debian.org/completeindex.html

Super-category	Sub-class	Description
Question	question	the post contains a new and independent question.
	add	the post provides additional information or asks a follow-up question, regarding a previous question.
	confirmation	the post confirms details or errors in a question.
	correction	the post corrects errors in a question.
	information*	the post is in a non-troubleshooting thread, and only provides information.
Answer	answer	the post proposes an answer to a question.
	add	the post provides additional information to an answer.
	confirmation	the post confirms details or errors in an answer.
	correction	the post corrects errors in an answer.
	objection	the post objects to an answer.
Resolution	—	a user confirms that an answer works.*
Reproduction	—	a non-initiator asks a similar question, or confirms that an answer should work.
Other	—	the post does not belong to any of the above classes.

Table 1: The Dialogue Act (DA) set used for annotating ILIAD dataset. ("*" signifies a difference over the original DA proposed by Kim et al. (2010b))

Regarding the Solvedness label for ILIAD dataset, the original thread-level annotations were done by three annotators are on a five-point scale, with 1 indicating high confidence in Solvedness for a given thread and 5 indicating low confidence (Baldwin et al., 2007). These annotations were aggregated by taking the simple mean across the three annotators and discretising into binary classes, with 2.5 as the breakpoint. Out of the 250 threads in the ILIAD dataset, 28 threads had a score of 2.5 and were discarded in the original paper. In the interests of comparability with the original research, we experiment over this reduced dataset (denoted ILIAD_{222}), but question the theoretical soundness of removing these threads from the dataset, so additionally experiment with the full dataset (denoted ILIAD).

4 Discourse Structure Parsing for Thread Solvedness Classification

Baldwin et al. (2007) showed that the task of automatic Solvedness classification on ILIAD_{222} is extremely hard. This is because the annotation was often based on expert knowledge of Linux, and a great deal of information not explicitly mentioned in the thread. Take the thread in Figure 1 for example: although two independent answers are provided in Post2 and Post5, it is almost impossible to identify whether there is a correct solution unless the whole thread is understood at a technical level.

Although predicting Solvedness is challenging, we believe that the use of thread discourse structure should assist in the task. As a first step, we need to do thread discourse structure parsing, which includes predicting both the linkings (Links) between posts and the type (DA) of each link.

Discourse structure parsing, as discussed in Wang et al. (2011), can be addressed in several ways. If a structured classification approach, such as Conditional Random Fields (CRFs: Lafferty et al. (2001)), is used, we can either classify the Link and DA separately and compose them afterwards (denoted as Composition), or we can classify the combined Link and DA (e.g. treat 0+Question-question as a single label) directly (denoted as Combine). Another approach is

Feature Category	Feature Name	Description
DA-only	LastPostDA	The DA of the last post in the thread.
	LastNonInitDA	The DA of the last post from a non-initiator in the thread.
	HasResolution	Whether the thread contains a Resolution post.
LinkDA-based	LastPairDA	The DA pair for the deepest post pair in the thread tree. In the case of ties, the pair containing the latest post is chosen.
	LastSubthreadDA	The sequence of DAs in the longest subthread in the thread tree. In case of ties, the sequence containing the latest post is chosen.

Table 2: Thread discourse structure features used for Solvedness classification.

to treat discourse structure parsing as a dependency parsing problem. Dependency parsing (Kübler et al., 2009) is the task of automatically predicting the dependency structure of a token sequence, in the form of binary asymmetric dependency relations with dependency types. The joint classification task of Link and DA is a natural fit for dependency parsing, in that the task is intrinsically one of inferring labelled dependencies between posts.

For discourse structure parsing, all experiments were carried out based on 10-fold cross-validation, stratifying at the thread level to ensure that all posts from a given thread occur in a single fold. The results are evaluated using post-level micro-averaged F-score ($\beta = 1$). All three discourse structure parsing methods were tested in our experiments, by using CRFSGD (Bottou, 2011) and MaltParser (Nivre et al., 2007). As for features, we experimented with all the features proposed by Wang et al. (2011), including Initiator, Position, TitSim, PostSim, Punct and UserProf, as well as many of our own features. We found that using CRFSGD with a simple Initiator (i.e. whether a post's author is the initiator of the thread) feature and the Combine approach achieves the highest Link and DA joint (LinkDA) classification F-score of 0.626. This is significantly better[4] than a strong heuristic baseline which classifies all first posts as 0+Question-question and all subsequent posts as 1+Answer-answer, which achieves a joint classification F-score of 0.420.

We also explored the possibility of domain adaptation, by using threads from the CNET dataset of Kim et al. (2010b) to augment the ILIAD thread discourse structure parsing. However, we were unable to achieve any significant improvements.

When using the thread discourse structure (i.e. Link and DA) for Solvedness prediction, one natural question is "could we simply use Resolution to identify solved thread?" While Resolution is a clear identifier of solved threads with 100% precision, only 8% of the threads contain Resolution posts, and yet 80.4% of the threads are labelled as solved. Therefore, by only using Resolution, a classifier could not do better than a majority class baseline. Instead, we propose to use a combination of discourse structure features to address the Solvedness classification problem. Table 2 displays all the discourse structure features[5] used in this paper, which can be grouped into two categories: (1) those based on only the DAs (DA-only); and (2) those based

[4]All statistical significance results in the paper are based on randomised estimation (Yeh, 2000), at a significance level of $p < 0.05$.

[5]We have experimented with many "discourse structure" features and non-"discourse structure" features, and these are particularly useful and interesting.

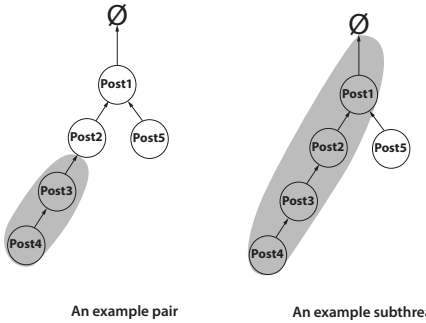


Figure 2: Examples of a pair and a subthread in a thread.

on Link and DA (LinkDA-based). When using the Link information, we rely on the notions of "pairing" and "subthreading". A pair is defined to be the combination of a post with the parent post it links to (noting that a given post can participate as child in multiple "pairs" as it can link to multiple posts), and a subthread contains all posts in a given path from a leaf node to the root node following the link structure. Figure 2 shows an example of each, based on the sample thread from Figure 1. As the `LastPostDA` is Answer-answer from Post5 (the final post), the `LastNonInitDA` for the thread is Answer-answer, `HasResolution` is 0 (as there are no Resolution posts), `LastPairDA` is Answer-add/Answer-confirmation from the pairing of Post4 and Post3, and `LastSubthreadDA` is Answer-add/Answer-confirmation/Answer-answer/Question-question from the subthread Post4/Post3/Post2/Post1.

5 Solvedness Classification

Baldwin et al. (2007) experimented with various learners from three machine learning software packages, namely LIBSVM (Chang and Lin, 2011), TiMBL (Daelemans et al., 2010) and Weka (Hall et al., 2009), and found that LIBSVM performs superiorly on the Solvedness classification task. Therefore, LIBSVM is used for Solvedness classification in this research.

In our initial experiments, we experimented with different kernel functions for LIBSVM, including linear, polynomial, radial basis function (RBF) and sigmoid kernels, and found the linear kernel to outperform other kernels. Therefore, LIBSVM with linear kernel is used throughout our experiments. We approach the Solvedness classification task by firstly following the procedure of Baldwin et al. (2007), where ILIAD_{222} is used. Subsequently, we carry out experiments over the full 250-thread ILIAD dataset. In both cases, various combinations of the features introduced in Section 4 are used. To generate these features, both the gold-standard LinkDAs and the automatically predicted ones are used.

All our Solvedness classification experiments were carried out based on stratified 10-fold cross-validation. The results are evaluated using classification accuracy (ACC). As our baselines, we use a majority classifier (ZeroR), as well as the best Solvedness classifier provided by Baldwin et al. (2007) (ADCS). As mentioned earlier, randomised estimation (Yeh, 2000) (at a significance level of $p < 0.05$) is used throughout the paper for statistical significance testing.

Feature Category	System/feature(s)	ACC_{gold}	ACC_{auto}
Baseline	ZeroR	.779	
	ADCS	.788	
DA-only	`LastPostDA`	.833*	.775
	`LastNonInitDA`	.766	**.792**
	`HasResolution`	.779	.779
	`LastPostDA` +`LastNonInitDA`	.834*	.779
	`LastPostDA` +`HasResolution`	**.883***	.775
	`LastNonInitDA` +`HasResolution`	.874*	**.792**
	`AllDAFeat`	**.883***	.779
LinkDA-based	`LastPairDA`	.851*	**.792**
	`LastSubthreadDA`	.833*	.779
	`AllLinkDAFeat`	.833*	**.792**
	`AllDAFeat` +`AllLinkDAFeat`	.865*	**.792**

Table 3: Results over ILIAD$_{222}$, using discourse structure features from the gold-standard and also the discourse parsing model ("*" signifies a significantly better result than both baselines; the best result in each column is indicated in **boldface**).

5.1 Experiments over ILIAD$_{222}$

Table 3 presents the results from experiments over ILIAD$_{222}$, using the thread discourse structure features generated from both the gold-standard (ACC_{gold}) LinkDAs, and automatically predicted ones (ACC_{auto}). The automatically predicted discourse structure of the whole ILIAD$_{222}$ dataset is obtained by aggregating the discourse structure predictions from each fold of the 10-fold cross-validation experiments described in Section 4. The combination of all DA-only features (i.e. `LastPostDA`, `LastNonInitDA` and `HasResolution`) is denoted `AllDAFeat`, and the combination of all LinkDA-based features (i.e. `LastPairDA` and `LastSubthreadDA`) is denoted `AllLinkDAFeat`. Results which are significantly better than both baseline results are signified by "*", and the best result(s) in each column are presented in **boldface**.

Looking first at the ACC_{gold} results in Table 3 we can see that, not surprisingly, `HasResolution` by itself does not have any effect on the prediction (see our comments in Section 4). Moreover, while `LastPostDA` leads to a significant improvement, `LastNonInitDA` does not have a significant effect. More interestingly, the combination of `LastPostDA` or `LastNonInitDA` with `HasResolution` leads to further improvements. This is because the classifiers trained on `LastPostDA` or `LastNonInitDA` are aggressive and misclassify many solved threads as unsolved, which `HasResolution` can correct.

The ACC_{gold} column also shows both the potential and shortcomings of LinkDA-based features — i.e. while both `LastPairDA` and `LastSubthreadDA` lead to significantly better results in isolation, combining them does not lead to further improvements. Moreover, combining all the features (i.e. `AllDAFeat` +`AllLinkDAFeat`) leads to a drop in results compared to just using `AllDAFeat`. We hypothesise that there are a number of reasons for this. Firstly, the `LastPairDA`, `LastSubthreadDA` and DA-based features have dependencies between each other, in that they all draw on the same set of DAs. While they are closely related, the classifiers do not have any access into the internals of the features to leverage them, causing the learner to overfit the training data. Secondly, while `LastPairDA` and `LastSubthreadDA` lead to low results in isolation, this is almost certainly because of the sparse nature (`LastPairDA` and

Feature Category	System/feature(s)	ACC_{gold}	ACC_{auto}
Baseline	ZeroR	.804	
	ADCS	.804	
DA-only	LastPostDA	.784	.780
	LastNonInitDA	.792	.788
	HasResolution	.804	.804
	LastPostDA +LastNonInitDA	.848*	.776
	LastPostDA +HasResolution	.864*	.780
	LastNonInitDA +HasResolution	.872*	.788
	AllDAFeat	**.884***	.776
LinkDA-based	LastPairDA	.832	**.816**
	LastSubthreadDA	.832	.792
	AllLinkDAFeat	.824	.792
	AllDAFeat +AllLinkDAFeat	.852*	.792

Table 4: Results over ILIAD, using discourse structure features from the gold-standard and also the discourse parsing model ("*" signifies a significantly better result than both baselines; the best result in each column is indicated in **boldface**).

LastSubthreadDA have 72 and 135 distinct values, respectively), much moreso than the DA-based features. When combined with the other features, however, some of these features are found to have utility.

Looking next to the ACC_{auto} results in Table 3, we can see that we surpass the two baselines, delivering on the promise of discourse parsing aiding in Solvedness classification. The results drop appreciably relative to those achieved with the gold-standard labels, and in fact the improvements over the baselines aren't statistically significant. This is perhaps not surprising, however, given than the F-score for discourse parsing was a modest 0.626, meaning that errors will propagate through to the thread-level classification.

While these results are certainly encouraging, and were worthwhile in terms of establishing the superiority of our method when discourse parsing features are used, we always had reservations about the ILIAD_{222} dataset, due to the most contentious instances having been removed from the dataset. In introducing these instances back into the dataset and labelling them as solved, the task becomes both more realistic and more challenging, including the ZeroR baseline rising up further. In the next section, we reapply our methods to the ILIAD dataset.

5.2 Experiments over ILIAD

We carry out the same experiments done in Section 5.1 over the whole ILIAD dataset, and present the results in Table 4. Again, the results which are significantly better than both baseline results are signified by "*", and the best result in each column is presented in **boldface**.

From Table 4 we can see a similar trend to that in Section 5.1, with our method improving over both baselines when we use either gold-standard or automatically-predicted features. However, there are some notable differences. Looking first at the ACC_{gold} column, firstly, none of LastPostDA, LastNonInitDA and HasResolution led to any improvement in isolation. However, the combination of these three features led to results that are significantly better than the baselines, with AllDAFeat achieving the best result of 0.883. Secondly, neither LastPairDA nor LastSubthreadDA has a significant impact on results, and their combination

DA	LastPostDA	LastNonInitDA	HasResolution	AllDAFeat
Question-add	0.999	0.811	—	0.999
Question-confirmation	0.881	0.991	—	0.961
Question-information	0.918	—	—	0.918
Answer-answer	0.500	0.513	—	0.498
Answer-add	0.461	0.550	—	0.489
Answer-confirmation	0.918	—	—	0.918
Answer-objection	0.918	1.000	—	0.954
Reproduction	1.000	1.000	—	1.000
Resolution	0.332	0.918	0.229	0.237
Other	0.971	0.934	—	0.952

Table 5: Entropy of each DA against the Solvedness class distribution for every DA-only feature and AllDAFeat features.

(i.e. AllLinkDAFeat) also does not outperform the baselines significantly. Looking next to ACC_{auto}, we achieve the best results with LastPairDA once again, surpassing the baselines but not at a level of statistical significance. Overall, while it is clear that the Solvedness classification task becomes harder when we experiment with the full ILIAD dataset, we were able to reproduce the overall results from Section 5.1.

6 Results Analysis and Simulation

Examining the differences between the results for ACC_{gold} and ACC_{auto} in Section 5.1 and Section 5.2 leads us to suspect that if the F-score of the thread discourse parsing could be boosted, we would be able to achieve better Solvedness classification accuracy. Furthermore, because the most effective discourse structure features, i.e. LastPostDA, LastNonInitDA and HasResolution, only make use of a subset of the DAs, we anticipate that if we can improve the F-score over certain DAs, we will be able to significantly boost our Solvedness classification accuracy.

To test these hypotheses, firstly, we examine the entropy (presented in Table 5) of every DA against the Solvedness class distribution for each DA-only feature (i.e. LastPostDA, LastNonInitDA and HasResolution) and the combination of all DA-only features (i.e. AllDAFeat). From Table 5, we can see that Answer-answer, Answer-add and Resolution have relatively low entropy compared to the rest of the DAs. Therefore, it seems that these three DAs can contribute more in Solvedness classification.[6]

Secondly, we conducted simulation experiments to examine the potential relation between DA classification and Solvedness classification. The simulation starts with a seed DA classification result (SeedResults), based on CRFSGD and the Initiator feature. This seed DA classification achieves a F-score of 0.651, significantly better than a strong heuristic baseline (i.e. 0.515) which classifies all first posts as Question-question and all subsequent posts as Answer-answer. Then, an arbitrary higher goal (e.g. 0.8) is set and an artificial classification result (ArtificialResults) is created by randomly correcting errors in the output of the discourse parsing model. The corrections are made evenly across all DA labels, relative to the original error rates for each DA. Next, a simulator is used to predict the labels of each instance, by randomly selecting from the

[6]Note that this entropy analysis can only capture the association between a single DA and the Solvedness class, and we are not able to capture more subtle feature interactions.

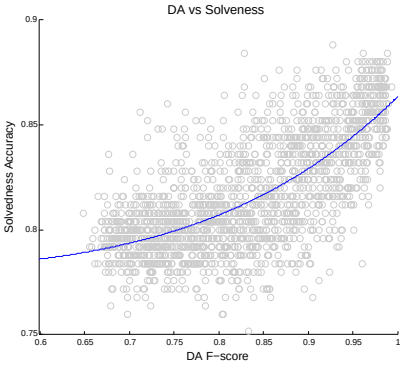


Figure 3: Simulation over all DAs (AllDA).

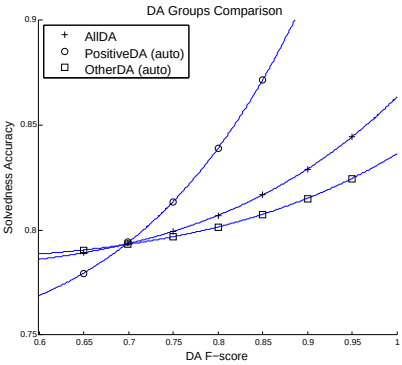


Figure 4: Simulation over automatically-generated DA groups (PositiveDA$_{auto}$ and OtherDA$_{auto}$).

labels returned by SeedResults and ArtificialResults with equal chance. In order to generate enough simulated results, we pick 20 goal F-score figures between 0.651 and 1.0, and run the simulator 100 times for each of these figures. Finally, we use these 2000 simulated discourse structure predictions to classify Solvedness using `AllDAFeat` features, and plot each pair of discourse structure F-score and Solvedness accuracy in a scatter plot. We also try to fit a series of simple polynomial models of the form $y = ax^n + b$ ($n \in \{1, 2, 3, 4, 5\}$)[7] to the plot. We find that the model for $y = ax^5 + b$ provides the best fit with the data, although the differences in the range $n \in \{2, 3, 4, 5\}$ are negligible. Figure 3 shows the graph, along with the curve of best fit for the function $y = ax^5 + b$.

From Figure 3 we can see that there is a clear correlation between the F-score of DA classification and the accuracy of Solvedness classification, and that the impact of DA classification on Solvedness classification is, in fact, accentuated for higher F-scores. Theoretically, therefore, by improving the DA classification F-score, the Solvedness classification accuracy will increase accordingly.

The entropy analysis showed that not all DAs have the same utility for the task of Solvedness classification — i.e. some DAs are more important (lower entropy) than others. We select the three DAs (i.e. Answer-answer, Answer-add and Resolution) with lowest entropy values from Table 5, because these DAs seem to be the most effective across the three feature types (i.e. `LastPostDA`, `LastNonInitDA` and `HasResolution`). Then, we carry out an analogous simulation over this set of automatically-selected DAs (PositiveDA$_{auto}$). Additionally, we conducted a simulation over the 8 non-selected DAs (OtherDA$_{auto}$).[8] Once again, a line of best fit for $y = ax^5 + b$ is generated for the resulting simulations. The curves of best fit are shown in Figure 4, along with the original curve of best fit for all DAs (AllDA) from Figure 3.

[7] Choosing $n > 5$ does not result in better fit with the data.

[8] Question-correction and Answer-correction are never used in annotating the discourse structure of ILIAD dataset.

DA Group	DA	F-score
PositiveDA$_{auto}$	Answer-answer	0.782
	Answer-add	0.641
	Resolution	0.514
OtherDA$_{auto}$	Question-question	0.992
	Question-add	0.678
	Question-confirmation	0
	Question-information	0
	Answer-confirmation	0
	Answer-objection	0
	Reproduction	0
	Other	0

Table 6: Micro-averaged DA classification F-scores per DA over ILIAD

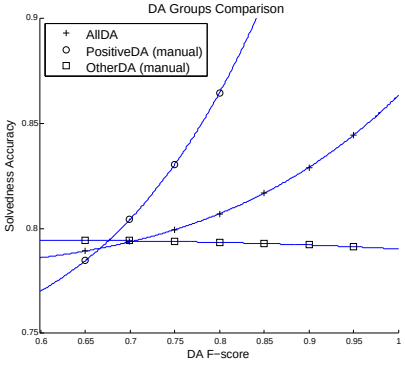


Figure 5: Simulation over manually-created DA groups (PositiveDA$_{manual}$ and OtherDA$_{manual}$).

From Figure 4 we can see that, as suspected, the PositiveDA$_{auto}$ group is much more important than the OtherDA$_{auto}$ group for Solvedness classification. Therefore, to improve Solvedness classification, we should focus our attention on improving the DA classification F-score for DAs such as Answer-answer, Answer-add and Resolution. Table 6 shows the micro-averaged F-scores of DA classification, calculated per DA. When we do a breakdown of the results for the discourse parsing model, we can see that there is definitely room for improvement with Answer-answer, Answer-add and Resolution. Moreover, Answer-answer and Answer-add are the most-frequent and third most-frequent DAs in the ILIAD dataset, respectively, appearing 354 and 147 times. Therefore, there appears to be considerable scope for improvement.

While the identification of the more important DAs can be done automatically as shown above, we also attempted to select them in a more ad hoc way, based on our understanding and analysis of the data set. Intuitively, if a thread's last post or the last post from a non-initiator is Question-confirmation, Question-information, Answer-confirmation or Answer-objection, this thread is more likely to be unresolved. At the same time, we can observe that the micro-

average F-scores for all these DAs are 0, that is the model never predicts a post to be one of these DA types correctly. To explore the utility of these additional DAs, we conducted an additional simulation experiment including Question-confirmation, Question-information, Answer-confirmation, Answer-objection and Resolution in PositiveDA$_{manual}$, and relegating the other 6 DAs to OtherDA$_{manual}$. The results for these manually-created groupings are shown in Figure 5.

From Figure 5 we can see that the improvements in discourse parsing over the manually-chosen PositiveDA$_{manual}$ will lead to even greater improvements over Solvedness prediction than before, if only we can get the models to make predictions using them. Perhaps even more surprising is that our simulations predict that improvements over OtherDA$_{manual}$ stand to *degrade* Solvedness classification. These findings set the direction for future work on improving the F-score of the discourse parser.

7 Conclusions and Future Work

In this research, we explore the task of Solvedness classification, that is the automatic prediction of whether the information need on the part of the initiator of a thread has been resolved or not, by parsing thread discourse structure in the form of a rooted directed acyclic graph over posts, with edges labelled with dialogue acts. While Solvedness classification has been shown to be very difficult in previous research (Baldwin et al., 2007), we achieve significantly better results using gold-standard discourse structure. We are also able to attain improvements in Solvedness classification accuracy using automatically-predicted thread discourse structure, although not at a level of statistical significance. However, simulations suggest that as we improve the F-score of thread discourse structure parsing, the Solvedness classification accuracy will increase disproportionately. Additionally, we showed that a particular subset of DAs is crucial to Solvedness classification accuracy, and that if we can improve the F-score of our discourse structure predictions over these DAs, we stand to make large gains in Solvedness classification accuracy.

In future work, we plan to firstly investigate ways to improve the discourse parser F-score over the PositiveDA$_{auto}$ and PositiveDA$_{manual}$ sets of DAs. Moreover, we plan to delve further into feature engineering, looking at other means of capturing thread discourse structure. Additionally, although our preliminary experiments on using CNET annotated threads to help ILIAD discourse structure parsing were not positive, it would be interesting to investigate the effect of using CNET threads in predicting specific DAs for the ILIAD dataset.

Acknowledgements

NICTA is funded by the Australian government as represented by Department of Broadband, Communication and Digital Economy, and the Australian Research Council through the ICT Centre of Excellence programme.

References

Baldwin, T., Martinez, D., and Penman, R. B. (2007). Automatic thread classification for Linux user forum information access. In *Proceedings of the 12th Australasian Document Computing Symposium (ADCS 2007)*, pages 72–79, Melbourne, Australia.

Bottou, L. (2011). CRFSGD software. `http://leon.bottou.org/projects/sgd`.

Cao, X., Cong, G., Cui, B., Jensen, C. S., and Zhang, C. (2009). The use of categorization information in language models for question retrieval. In *Proceedings of the 18th ACM Conference on Information and Knowledge Management (CIKM 2009)*, pages 265–274, Hong Kong, China.

Chang, C.-C. and Lin, C.-J. (2011). LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2(3):27:1–27:27. Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.

Cong, G., Wang, L., Lin, C.-Y., Song, Y.-I., and Sun, Y. (2008). Finding question-answer pairs from online forums. In *Proceedings of 31st International ACM-SIGIR Conference on Research and Development in Information Retrieval (SIGIR'08)*, pages 467–474, Singapore.

Daelemans, W., Zavrel, J., Van der Sloot, K., and Van den Bosch, A. (2010). TiMBL: Tilburg memory based learner, version 6.3, API guide. *ILK Research Group Technical Report Series*, 03(10). Software available at `http://ilk.uvt.nl/timbl/`.

Ding, S., Cong, G., Lin, C.-Y., and Zhu, X. (2008). Using conditional random fields to extract context and answers of questions from online forums. In *Proceedings of the 46th Annual Meeting of the ACL: HLT (ACL 2008)*, pages 710–718, Columbus, USA.

Elsas, J. L. and Carbonell, J. G. (2009). It pays to be picky: An evaluation of thread retrieval in online forums. In *Proceedings of 32nd International ACM-SIGIR Conference on Research and Development in Information Retrieval (SIGIR'09)*, pages 714–715, Boston, USA.

Fortuna, B., Rodrigues, E. M., and Milic-Frayling, N. (2007). Improving the classification of newsgroup messages through social network analysis. In *Proceedings of the 16th ACM Conference on Information and Knowledge Management (CIKM 2007)*, pages 877–880, Lisbon, Portugal.

Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., and Witten, I. H. (2009). The WEKA data mining software: An update. *ACM SIGKDD Explorations Newsletter*, 11(1):10–18.

Kim, J., Chern, G., Feng, D., Shaw, E., and Hovy, E. (2006). Mining and assessing discussions on the web through speech act analysis. In *Proceedings of the ISWC'06 Workshop on Web Content Mining with Human Language Technologies*, Athens, USA.

Kim, S. N., Cavedon, L., and Baldwin, T. (2010a). Classifying dialogue acts in one-on-one live chats. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing (EMNLP 2010)*, pages 862–871, Boston, USA.

Kim, S. N., Wang, L., and Baldwin, T. (2010b). Tagging and linking web forum posts. In *Proceedings of the 14th Conference on Computational Natural Language Learning (CoNLL-2010)*, pages 192–202, Uppsala, Sweden.

Kübler, S., McDonald, R., and Nivre, J. (2009). Dependency parsing. *Synthesis Lectures on Human Language Technologies*, 2(1):1–127.

Lafferty, J., McCallum, A., and Pereira, F. (2001). Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the 18th International Conference on Machine Learning*, pages 282–289, Williamstown, USA.

Lampert, A., Dale, R., and Paris, C. (2008a). The nature of requests and commitments in email messages. In *Proceedings of the AAAI 2008 Workshop on Enhanced Messaging*, pages 42–47, Chicago, USA.

Lampert, A., Dale, R., and Paris, C. (2008b). Requests and commitments in email are more complex than you think: Eight reasons to be cautious. In *Proceedings of the Australasian Language Technology Association Workshop 2008 (ALTA 2008)*, pages 64–72, Hobart, Australia.

Lampert, A., Dale, R., and Paris, C. (2010). Detecting emails containing requests for action. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL HLT 2010)*, pages 984–992, Los Angeles, California.

Lampert, A., Paris, C., and Dale, R. (2007). Can requests-for-action and commitments-to-act be reliably identified in email messages? In *Proceedings of the Twelfth Australasian Document Computing Symposium (ADCS 2007)*, pages 48–55, Melbourne, Australia.

Muthmann, K., Barczyński, W. M., Brauer, F., and Löser, A. (2009). Near-duplicate detection for web-forums. In *Proceedings of the 2009 International Database Engineering & Applications Symposium (IDEAS 2009)*, pages 142–151, Cetraro, Italy.

Nivre, J., Hall, J., Nilsson, J., Chanev, A., Eryigit, G., Kübler, S., Marinov, S., and Marsi, E. (2007). MaltParser: A language-independent system for data-driven dependency parsing. *Natural Language Engineering*, 13(02):95–135.

Seo, J., Croft, W. B., and Smith, D. A. (2009). Online community search using thread structure. In *Proceedings of the 18th ACM Conference on Information and Knowledge Management (CIKM 2009)*, pages 1907–1910, Hong Kong, China.

Shrestha, L. and McKeown, K. (2004). Detection of question-answer pairs in email conversations. In *Proceedings of the 20th International Conference on Computational Linguistics (COLING 2004)*, pages 889–895, Geneva, Switzerland.

Sondhi, P., Gupta, M., Zhai, C., and Hockenmaier, J. (2010). Shallow information extraction from medical forum data. In *Proceedings of the 23rd International Conference on Computational Linguistics (COLING 2010), Posters Volume*, pages 1158–1166, Beijing, China.

Wang, L., Lui, M., Kim, S. N., Nivre, J., and Baldwin, T. (2011). Predicting thread discourse structure over technical web forums. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 13–25, Edinburgh, UK.

Wang, Y.-C. and Rosé, C. P. (2010). Making conversational structure explicit: identification of initiation-response pairs within online discussions. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL HLT 2010)*, pages 673–676.

Xi, W., Lind, J., and Brill, E. (2004). Learning effective ranking functions for newsgroup search. In *Proceedings of 27th International ACM-SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2004)*, pages 394–401, Sheffield, UK.

Yeh, A. (2000). More accurate tests for the statistical significance of result differences. In *Proceedings of the 18th International Conference on Computational Linguistics (COLING 2000)*, pages 947–953, Saarbrücken, Germany.

Implicit Discourse Relation Recognition by Selecting Typical Training Examples

Xun Wang[1] *Sujian Li*[1*] *Jiwei Li*[1] *Wenjie Li*[2]
(1) Key Laboratory of Computational Linguistics, Peking University, Ministry of Education, CHINA
(2) Department of Computing, Hong Kong Polytechnic University
{xunwang,lisujian,bdlijiwei}@pku.edu.cn, cswjli@comp.polyu.edu.hk

ABSTRACT

Implicit discourse relation recognition is a challenging task in the natural language processing field, but important to many applications such as question answering, summarization and so on. Previous research used either artificially created implicit discourse relations with connectives removed from explicit relations or annotated implicit relations as training data to detect the possible implicit relations, and do not further discern which examples are fit to be training data. This paper is the first time to apply a different typical/atypical perspective to select the most suitable discourse relation examples as training data. To differentiate typical and atypical examples for each discourse relation, a novel single centroid clustering algorithm is proposed. With this typical/atypical distinction, we aim to recognize those easily identified discourse relations more precisely so as to promote the performance of the implicit relation recognition. The experimental results verify that the proposed new method outperforms the state-of-the-art methods.

KEYWORDS: Discourse relation recognition, single centroid clustering, implicit discourse relation.

* Corresponding author.

Proceedings of COLING 2012: Technical Papers, pages 2757–2772,
COLING 2012, Mumbai, December 2012.

1 Introduction

It is widely agreed that sentences/clauses are usually not understood in isolation, but in relation to their neighbouring sentences/clauses. The task of discourse relation recognition is to identify and label the relations between sentences/clauses, which is fundamental to many natural language processing applications such as question answering, automatic summarization and so on.

Discourse relations, such as comparison and causal relations, can be divided into explicit and implicit relations by the presence or absence of discourse connectives (e.g., *but*, *because* et. al.). Previous study indicates that the presence of discourse markers can greatly help relation recognition and the most general senses (i.e., comparison, contingency, temporal and expansion) can be disambiguated with 93% accuracy based solely on the discourse connectives (Pitler et al., 2008). On the other hand, the absence of explicit textual cues makes it very difficult to identify the implicit discourse relations. Thus, recently discourse relation recognition research puts more efforts to meet the challenges in implicit discourse relation recognition.

Existing work mainly focused on exploiting various linguistic features to learn the implicit discourse relation classifiers based on the training data collected (Wellner, Pustejovsky and Havasi, 2006; Pitler, Louis and Nenkova, 2009; Lin, Kan and Ng 2009; Wang, Su and Tan, 2010). Most useful linguistic features (such as word pairs) are extracted from the local context, which is usually determined as **Argument 1 (Arg1,** the first sentence/clause**)** plus **Argument 2** (**Arg2**, the second sentence/clause). Like the other related work in the literature, in this paper, we focus on the recognition of local implicit discourse relations, i.e. only the two arguments are examined.

To collect training data, the state-of-the-art methods normally start from the artificial/real perspective and simply make use of the implicit relations either derived from explicit or manual annotations. Marcu and Echihabi (2002); Sporleder and Lascarides (2008) created artificial implicit relations as training data by removing discourse connectives from the explicit relation examples. The advantage of these methods is that a large number of (artificial) implicit relation examples could be used as training data, saving the labor extensive and time-consuming annotation work. However, the experimental results in Sporleder and Lascarides (2008) showed that training on a large artificial data set is not necessarily a good strategy. Lin, Kan and Ng (2009) also pointed out that an artificially implicit relation corpus may exhibit marked differences from a natively implicit one. Also surprising is the fact that the results were not as good as expected when the classifiers are trained by using the manually annotated real implicit relations, though better than the results based on the artificial implicit relations.

Then the following questions come to our minds. Do all the real natively implicit relation examples provide useful hints for training the classifiers? Is it a reasonable choice if the training data is over-restricted to the annotated implicit relation examples even when the quantity of these data is limited and their annotation demands a high cost? Can a part of, if not all of, the artificial implicit relations created from the explicit relations be picked out to train an implicit relation classifier? In short, can we obtain more effective training examples at less cost?

With the above consideration, we argue that an effective training set is composed of typical examples, which have distinct characteristics to signify their discourse relations. These typical examples, however, can be either the natively implicit relations or the created implicit relations with connectives removed from the explicit relations. Using the typical examples as training data,

an implicit relation classifier with higher discrimination power can be built according to the linguistic features in the two arguments.

We provide three *Comparison* relation examples from the Penn Discourse TreeBank (PDTB) v2.0 (Prasad et al., 2008) which is widely used in the research of relation recognition as follows to illustrate what the possible typical examples are like.

(1) **Arg 1**: 44 North Koreans oppose the plan,
Arg 2: (while) South Koreans, Japanese and Taiwanese accept it or are neutral.
(2) **Arg 1**: In such situations, you cannot write rules in advance.
Arg 2: you can only make sure the President takes the responsibility.
(3) **Arg 1**: Columbia Savings is a major holder of so-called junk bonds.
Arg 2: New federal legislation requires that all thrifts divest themselves of such speculative securities over a period of years.

Here, the first one is an artificial implicit relation with the connective (i.e. "*while*") deleted while the second and third examples are natively implicit. The first and second ones are possibly typical because they have distinguishable linguistic features (such as: oppose/accept, cannot /can) to verify their relations. In contrast, it is hard to find significant characteristics in the third one to determine its discourse relation. The trained implicit relation classifier would possibly suffer a decline in performance if a lot of examples like the third one are included in the training set.

Based on the analysis above, we for the first time propose to select training data for implicit discourse relation recognition from a new typical/atypical perspective other than from an artificial/real perspective. Identifying the typical examples from both artificially created and real implicit discourse relations is the focus of this work. Assuming that the typical examples of a discourse relation are usually connected through the similar features, Yarowsky's algorithm (1995), as one of the first bootstrapping algorithms, gives us the following inspiration: given a small set of *seed* typical discourse relation examples, more typical examples are added iteratively by identifying the significant features of the seed set. In this paper, a training data selection approach named single centroid clustering (SCC) is proposed to acquire the typical examples for each relation. With the typical examples in the training set, the task of implicit relation recognition is cast to a classification problem. The experimental results show that the training set selected in such a way can improve the performance of an implicit relation classifier.

The rest of this paper is organized as follows. Section 2 introduces the related work. Section 3 describes our framework of implicit relation recognition, and introduces the types of features involved. Section 4 proposes the single centroid clustering algorithm that selects the typical examples iteratively. Section 5 presents the experimental results. Section 6 concludes our work.

2 Related Work

2.1 Implicit discourse relation recognition

So far, the existing research which used statistical models to recognize implicit discourse relations mainly falls into two categories according to whether the data annotation is required.

One research line tried to use the large quantity of unannotated explicit relations as a training set, which are roughly identified by discourse connectives and then converted to artificial implicit relations through removing the discourse connectives. Among the pioneer work was the one

presented by Marcu and Echihabi (2002) who applied massive amounts of unannotated explicit relations and lexical features to train the Naïve Bayes classifier for both explicit and implicit discourse relation recognition. Following the same idea, Saito, Yamamoto and Sekine (2006) conducted the experiments with the combination of cross-argument word pairs and phrasal patterns as features on Japanese sentences. Blair-Goldensohn（2007）further extended the work of Marcu and Echihabi (2002) by involving syntactic filtering and topic segmentation. Another interesting work is that of Zhou et al. (2010), which predicted discourse connectives between arguments via a language model. Then the generated connectives plus other linguistic features were combined in a supervised framework to determine the implicit discourse relation.

However, Sporleder and Lascarides (2008) discovered that the models of Marcu and Echihabi (2002) did not perform well on implicit relations recognition with artificially created relations as training data and concluded that removing discourse markers may lead to a meaning shift in the examples. Sporleder and Lascarides (2008) promoted the other research line that used the human-annotated training data. The development of various discourse banks also made the use of human-annotated data feasible. Based on Rhetorical Structure Theory Discourse Treebank (RST-DT) (Carlson et al. 2001), Soricut and Marcu (2003) developed two probabilistic models to identify elementary discourse units and generate discourse trees at the sentence level. Further Hernault et al. (2010); Feng and Hirst (2012) explore various features for discourse tree building on RST-DT. With the Discourse Graphbank (Wolf and Gibson, 2005), Wellner et al.(2006) integrated multiple knowledge sources to produce syntactic and lexical semantic features, which were then used to automatically identify and classify explicit and implicit discourse relations. Especially after the release of the second version of the PDTB v2.0 (Prasad et al., 2008), more research began to take the advantage of the annotated implicit relations for training purpose and were dedicated to exploiting various linguistic features in the supervised framework (Pitler, Louis and Nenkova, 2009; Lin, Kan and Ng, 2009; Wang, Su and Tan, 2010). Lin, Kan and Ng (2009) conducted a thorough performance analysis for four classes of features including contextual relations, constituent parse features, dependency parse features and cross-argument lexical pairs, while Pitler et al. (2009) applied several linguistically informed features, such as word polarity, verb classes, and word pairs. Wang, Su and Tan (2010) adopted the tree kernel approach to mine more structure information and got better results. These efforts of feature selection have achieved better performance though not that satisfying. The quality of training data are partly responsible for the difficulty of improving the performance of implicit relation recognition.

To better recognize the implicit discourse relations, we propose to review the annotated discourse corpora available at hand, identify and choose typical relation examples as training data for supervised learning. To the best of our knowledge, this paper is the first time to re-think the training data and implicit relation recognition from a novel perspective.

2.2 Rhetoric Discourse Treebank and Penn Discourse Treebank

As for the available discourse corpora, due to the space limitation we mainly introduce the two widely used discourse corpora - the Penn Discourse TreeBank (PDTB) and Rhetorical Structure Theory Discourse Treebank (RST-DT), which provide a common platform for researchers to develop discourse-centric systems.

The PDTB focuses on encoding discourse relations with the discourse connectives, adopting a lexically grounded approach for the annotation. For each pair of adjacent sentences within the same paragraph, annotators selected the explicit or implicit discourse connective which best

expressed the relation between the sentences. Then, the annotations can be seen as being of a predicate-argument structure, where a discourse connective is treated as a predicate taking a pair of adjacent sentences as its arguments. Thus, this discourse connective grounded approach exposes a clearly defined level of discourse structure. In PDTB, a hierarchy of relation tags is provided for the relation annotation. In our experiments, we only use the top level of the annotations, which is composed of four major relation classes: *Temporal*, *Contingency*, *Comparison* and *Expansion*. These four core relations allow us to be theory-neutral, since they are almost included in all discourse theories, sometimes under different names.

RST-DT is manually annotated under the Rhetoric Structure Theory framework (Mann and Thompson, 1988). In this corpus the rhetoric relations are labelled hierarchically between non-overlapping adjacent text spans which range from elementary discourse units (EDU, the minimal building blocks of a discourse tree) to paragraph. A total of 110 different relations were used for the tagging of the RST corpus (RST-DT, 2002). The final inventory of the relations is data driven and can be partitioned into 18 classes, from which we still select four classes including *Temporal*, *Contrast*, *Cause*, and *Background* to verify our method. These four relations spanning over individual sentences are collected to keep consistent with the discourse relations from PDTB.

So far, most of the previous works experimented on one corpus only. With the aim to verify the portability of our methods, we examine two corpora in this paper.

3 The Learning Framework for Implicit Discourse Relation Recognition

In this paper, the problem of implicit relation recognition is approached in the supervised learning framework. Figure 1 illustrates the architecture of our system.

The first and most important step is to collect the training data. As stated in Section 1, on the one hand, not all the annotated implicit relations contain significant features to distinguish themselves from the other relation types. On the other hand, we expect to pick out the suitable examples of the artificial implicit relations and strengthen their influence on the training process. We argue that the examples suitable to be training data are generally the typical ones having distinct linguistic features to signify their discourse relations, yet they can be of real implicit relations (denoted as *IM* data) or artificially implicit relations with connectives removed from explicit relations (denoted as *EX* data).

To select typical examples, for each discourse relation type the original artificial/real partition (denoted as EX_i/IM_i) is converted to a novel typical/atypical partition (denoted as A_i/B_i), which is obtained automatically by the proposed *single centroid clustering* (SCC) algorithm. Starting from an initial seed set, SCC iteratively refines typical examples and removes atypical examples if necessary. This algorithm is detailed in Section 4.

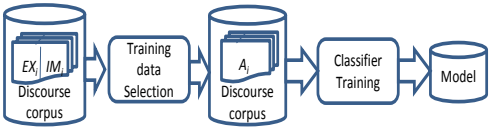


FIGURE 1 – System architecture for implicit discourse relation recognition

Assume there are n discourse relations. Let $Y=\{R_0, R_1, \ldots, R_n\}$ where R_i represents the ith typical relation type and R_0 denotes the atypical case. After the conversion from artificial/real partition to

typical/atypical partition, we get the $A_i/B_i(1 \le i \le n)$ and assign the relation label $R_i(1 \le i \le n)$ to each typical example in the set A_i. Each example in the union $A_0 = \cup B_i(1 \le i \le n)$ is labelled as R_0. Then the set of ordered pairs $<A_i, R_i>$ $(0 \le i \le n)$ can be used to train an implicit relation classifier for labelling $R_i(1 \le i \le n)$. Both clustering and classification require representing the annotated argument pairs with feature vectors. We introduce the feature selection in subsection 3.1.

3.1 Feature Selection

Various linguistic features have been experimented for recognizing implicit discourse relations in previous studies (Marcu and Echihabi, 2002; Pitler, Louis and Nenkova, 2009; Lin et al., 2009). Learning from them, we consider the following 7 types of features.

Polarity: The polarity of each sentiment word is tagged as positive, negative or neutral according to Multi-perspective Question Answering Opinion Corpus (Wilson et al., 2005). Note that the sentiment words preceded by negated words would be assigned an opposite tag. For example, "good" would be assigned as positive while “not good” is negative. Negated neutral is ignored. The occurrence of negative, positive and neutral polarities in each argument and their cross product are used as features.

Inquirer tags: General Inquirer lexicon (Stone et al., 1966) divides each word into fine-grained semantic categories described by the inquirer tags. From all the categories, we select 21 pairs of complementary categories, such as: *Rise* versus *Fall*, or *Pleasure* versus *Pain*, etc. The occurrence of each complementary category pair in the two arguments are used as features.

Modality: The presence of modal words including their various tenses and abbreviations in both arguments and their cross product are used as features.

SameWord: This type of feature represents whether a noun or a verb simultaneously occurs in both arguments. The intuition of using this feature is similar to that of the ***Verbs*** feature in (Pitler et al., 2009), for indicating the semantic association of the two arguments.

FirstLastFirst3: The first word, the last word, the first three words of each argument, the pair of the two first words and the pair of the two last words in the two arguments are used as features.

CrossWordPairs: The words in each argument compose one set. This type of features indicates the word pairs from the cross product of the two sets.

IntraWordPairs: The word pairs that occur in the same argument.

Since the length of the two arguments is relatively short, it is quite common that a feature is observed only once if it is present. Hence each feature is assigned a binary value to indicate whether it is present or absent. Assuming d features are extracted, each example is represented with a d-dimension binary feature vector.

4 Single Centroid Clustering for Training Example Selection

4.1 Overview

A good training set usually exhibits the property that most of its items have distinct features to differentiate the instances in the different classes. To precisely classify implicit discourse relations, the typical examples which have significant linguistic features except discourse connectives for identifying their relations are fit to be included in the training set. In this section,

we introduce the Single Centroid Clustering (SCC) algorithm which picks out the typical examples for each discourse relation from both *EX* and *IM* data.

Algorithm 1: Single-Centroid Clustering algorithm

Input: For relation i, artificial implicit relation set EX_i, real implicit relation set IM_i,
Output: Typical relation example set A_i, Atypical relation example set B_i

1. Initialize A_i: A_i= seed set of typical examples;
2. $B_i = EX_i \cup IM_i - A_i$
3. Compute the centroid CA_i for A_i
4. While stopping criterion has not been met
5. For each example e_j in A_i:
6. If dist $(e_j, CA_i) > T_{dis}^{(i)}$:
7. $A_i = A_i - \{e_j\}$; $B_i = B_i \cup \{e_j\}$
8. For each example e_j in B_i:
9. If dist$(e_j, CA_i) <= T_{dis}^{(i)}$:
10. $A_i = A_i \cup \{e_j\}$; $B_i = B_i - \{e_j\}$
11. Compute the centroid CA_i for A_i
12. End While

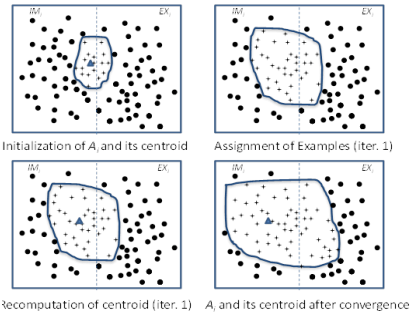


FIGURE 2 – Illustration of the Single Centroid Clustering algorithm

The principle underlying SCC is similar to that of the Yarowsky algorithm (1995), which has been successfully applied to the Word Sense Disambiguation (WSD). Yarowsky augmented the seed sets of each sense based on two powerful constraints, namely one-sense-per-collocation and one-sense-per-discourse. In our SCC algorithm, the features introduced in Section 3.1 are used to obtain the constraints of augmenting the seed sets and pick out those typical examples for each discourse relation. The SCC algorithm, as shown in Algorithm 1, consists of two loops. The "outer loop" can be regarded as a supervised learning process. In particular, based on the current available typical examples, SCC computes for each relation the centroid that judges which features are significant. The "inner loop" uses the current centroid of a relation to re-assign all the examples of the relation as either typical or atypical.

Figure 2 illustrates a snapshot of SCC on relation R_i, with dots and crosses representing the data in the A_i and B_i sets respectively. The closed curve in the left-top graph represents the seed set of typical examples. The closed curves in the other three graphs represent the intermediate and final results of the typical examples sets. The solid triangles in the middle of the closed curves denote the centroids computed based on typical examples. When SCC reaches its stable state, the final typical example set is passed to the classification models as training data. Take the three sentence pairs in Section 1 for example, the ideal output from SCC should include the first and the second examples in the typical set of the Comparison relation.

4.2 Implementation Details

4.2.1 Seed Set Construction

For each relation $R_i(1 \le i \le n)$, we can identify a relatively small number of typical examples as the seed set either manually or automatically. Similar to the Yarowsky algorithm (1995), to avoid the laborious procedure, through observation we manually lay down some simple rules to identify the distinct features for each relation from the 7 feature types and then select those containing the distinct features from the corresponding relation examples to compose of the seed set. The rules for identifying distinct features are illustrated in Table 1. Taking the *Comparison* relation for example, rule (1) identify the features of "Arg 1 is positive and Arg 2 is negative" and "Arg 1 is negative and Arg 2 is positive" which are from the **Polarity** feature type. Rule (2) can identify the features which are related to the words *seldom*, *back*, etc. according to the feature types of **FirstLastFirst3**, **CrossWordPairs**, and **IntraWordPairs**. Other strategies of selecting typical example seed set and the experimental comparisons are provided in Subsection 5.3.

Class	Description of Rules
Comparison	(1) A pair of opposite polarity tags is identified respectively in Arg 1 and Arg 2. (2) Arg 1 or Arg 2 contains the words including *seldom, back, yet, only.*
Contingency	(1) Opposite polarity tags are identified respectively in Arg 1 and Arg 2. (2) Arg 1 or Arg 2 contains the words including *draw, as, result.*
Temporal	Arg 1 or Arg 2 contains the words including *following, last, first, second.*
Expansion	Arg 1 and Arg 2 contain the same noun words or verb words.

TABLE 1 – Rules for selecting the seed set of typical examples.

4.2.2 Centroid Computation

A_i can be seen as the iteratively refined typical set. Suppose A_i is composed of $|A_i|$ examples $\{e_1^{(i)}, e_2^{(i)}, \ldots, e_{|A_i|}^{(i)}\}$, each example $e_j^{(i)}$ $(1 \le j \le |A_i|)$ is represented by a d-dimension feature vector $(e_{j1}^{(i)}, e_{j2}^{(i)}, \ldots e_{jd}^{(i)})$. In the d-dimensional Boolean space, the centroid CA_i is also represented by a d-dimension binary feature vector $(c_1^{(i)}, c_2^{(i)}, \ldots, c_d^{(i)})$, where $c_k^{(i)}$ is the value in the k^{th} dimension. We define $c_k^{(i)}$ as:

$$c_k^{(i)} = \begin{cases} 1, & \frac{\sum_j e_{jk}^{(i)}}{|A_i|} > T_c^{(i)} \\ 0, & otherwise \end{cases} \quad (1)$$

where $T_c^{(i)}$ is the percentage threshold corresponding to R_i. $c_k^{(i)}$ is assigned to 1 if the k^{th} feature occurs more than a certain percentage (i.e. $T_c^{(i)}$) of the examples that belong to the typical set A_i.

In this way, the centroid values actually reflect which features are significant to the corresponding discourse relation. Normally, centroid is used to compute the "average" of all objects in a certain space, and it should be noted that the computation of *centroid* in a Boolean space here does not strictly observe the "average" form.

4.2.3 Distance Metric

For each relation R_i, we exclude atypical examples from A_i or select typical ones into A_i by computing the distance between discourse relation examples and the centroid of CA_i. Assuming the example e is represented by the feature vector $(e_1, e_2, \ldots, e_d)$, the distance between e and CA_i is defined as follows.

$$dist(e, CA_i) = \sum_k w_k \cdot |c_k^{(i)} - e_k| \quad (2)$$

$$w_k = \begin{cases} \frac{|A_i| \cdot T_c - \sum_k e_{jk}^{(i)}}{|A_i|}, c_k^{(i)} - e_k = -1 \\ \frac{\sum_k e_{jk}^{(i)}}{|A_i|}, \ otherwise \end{cases} \quad (3)$$

where $|c_k^{(i)} - e_k|$ reflects whether the example e has a different value from CA_i in the k-th dimension and $w_k (1 \leq k \leq d)$ is used to measure the influence of the difference in the k-th dimension on the distance between e and CA_i. Here, w_k is determined according to the frequency of the k-th feature occurring in all examples of a discourse relation. The distance between an example e and the centroid CA_i denotes the representativeness or to say the typicality of the example e to the relation R_i. The smaller the distance value of an example, the more typical the example is.

A distance threshold $T_{dis}^{(i)}$ is set to control which examples should be selected into the typical set of R_i. The examples with distance less than $T_{dis}^{(i)}$ are possibly re-assigned to the typical set A_i. $T_{dis}^{(i)}$ is defined depending on the maximum distance and the minimum distance between the examples and the centroid CA_i, i.e.,

$$T_{dis}^{(i)} = min_e dist(e, CA_i) + p^{(i)} (\max_e dist(e, CA_i) - \min_e dist(e, CA_i)) \quad (4)$$

where $p^{(i)}$ is a control parameter within the interval (0,1) for R_i. If $p^{(i)}$ is set 0, $T_{dis}^{(i)}$ equals to the minimum distance, meaning that no examples can be included into the typical set. On the other extreme, if $p^{(i)}$ is 1, $T_{dis}^{(i)}$ equals to the maximum distance, it allows all the examples to be selected. The value of $p^{(i)}$ is also tuned to assure that typical examples can be well selected in each iteration.

5 Experiments and Evaluation

5.1 Experiment Set-up

The experiments and evaluations are conducted on the PDTB and RST-DT corpus, which contains 2519 and 385 Wall Street Journal articles respectively. PDTB is mainly used to evaluate and analyse recognition performance of our methods. RST-DT is used to verify the portability.

Following the work of Pitler, Louis and Nenkova (2009), the sections 2-20 of PDTB are used for training, the sections 0-1 for development and the sections 21-22 for test. As for the discourse relations, we adopt the top level of PDTB's annotations, which is composed of four major relation classes: *Temporal*, *Contingency*, *Comparison* and *Expansion*. Though PDTB allows each

sentence pair to be annotated with more than one relation, we only extract the first relation labelled for each sentence pair here. Table 2 shows the number of each relations in PDTB.

Class	Training		Test	Develop.
	EX	*IM*	Implicit	Implicit
Comparison	4209	1894	146	191
Contingency	2505	3281	277	287
Temporal	2633	665	67	54
Expansion	4770	6792	556	651
Total	14117	12632	1046	1183

TABLE 2 – Discourse relation distribution in PDTB.

According to the 7 types of features introduced in Section 3.1, in total 4022 features are extracted. Then each sentence pair is represented as a 4022-dimension binary feature vector. The two classifiers, i.e., the Naïve Bayes and Decision Tree classifiers are implemented with MALLET[1]. Two metrics, i.e., accuracy and F_1 measure, are used to evaluate the performance:

$$\text{Acc} = \frac{TruePositive + TrueNegative}{All} \quad \text{and} \quad F_1(R_i) = \frac{2 \cdot Precision \cdot Recall}{Precision + Recall}$$

where *precision* and *recall* are two most common criteria to evaluate information retrieval and information extraction systems.

Four sets of experiments are designed (1) to tune the two thresholds $T_c^{(i)}$ and $T_{dis}^{(i)}$ in SCC; (2) to compare different strategies of selecting seed sets for SCC; (3) to compare the performance of various training sets on different classifiers; (4) to verify the portability of our methods.

5.2 Threshold Tuning in SCC

SCC aims at selecting typical examples for training discourse classifiers. Since it is difficult to directly evaluate the quality of a training set, we evaluate the training set outputted by SCC via the classification performance of a Decision Tree classifier. For each discourse relation R_i, SCC involves two main thresholds. $T_c^{(i)}$ determines which features are significant to the relation R_i, and $T_{dis}^{(i)}$ defines the borderline between the typical examples and the atypical ones. It is hard to find a global optimized solution for the combination of these two factors. So we apply a gradient search strategy. As in formula (4), $p^{(i)}$ is the only determining factor of $T_{dis}^{(i)}$. At first we set $p^{(i)}$ the value of 0.5, and different values of $T_c^{(i)}$ ranging from 0.05 to 0.35 are examined. Then, given that $T_c^{(i)}$ is set to the value with the best performance, we conduct experiments to find an appropriate value for $p^{(i)}$.

We ran four binary classifiers to distinguish each discourse relation (*Comp.*, *Cont.*, *Temp.*, and *Expa.* for short) from the others. For each relation, we include equal number of positive and negative examples in the training data. The positive examples are selected from the typical set of the relation while the negative examples are randomly chosen from the atypical set of the same relation or the other discourse relations. We use all the 1183 implicit relations in the development set, which is representative of the natural distribution of implicit discourse relations. Table 3 lists the F_1 and accuracy (within parentheses) of the implicit relation classifiers.

[1] www.mallet.cs.umass.edu.

T_c	*Comp*. vs. other	*Cont*. vs. other	*Temp*. vs. other	*Expa*. vs. other
0.05	23.2 (54.2)	39.0 (24.3)	12.8 (54.5)	0 (45.0)
0.10	23.9 (40.4)	41.9 (26.5)	12.8 (46.4)	64.4 (53.6)
0.15	**27.9 (39.4)**	38.6 (24.6)	12.5 (43.6)	66.2 (54.6)
0.20	27.6 (24.1)	39.9 (37.4)	12.4 (47.1)	68.3 (55.4)
0.25	17.4 (75.1)	42.1 (29.1)	**13.4 (45.5)**	**71.0 (55.0)**
0.30	18.4 (73.0)	**42.3 (28.3)**	11.7 (45.2)	55.2 (50.0)
0.35	17.0 (74.9)	39.1 (24.8)	12.6 (43.0)	55.2 (50.0)

TABLE 3 – F_1 (Acc) with varying $T_c^{(i)}$ values (p=0.5).

Table 3 shows that the value of $T_c^{(i)}$ directly influences the quality of the generated training set. When $T_c^{(i)}$ is assigned a smaller value, more features will satisfy the percentage requirement. That means more features will be reflected in the centroid and it will cause the distance between an example and the centroid is closer to one another. Then when $T_{dis}^{(i)}$ is fixed, more examples will enter into the typical set. Oppositely, when $T_c^{(i)}$ is assigned a larger value, it is more difficult for a feature to satisfy the percentage requirement. Then less number of features is reflected in the centroid. Notice that in general cases when the value of $T_c^{(i)}$ is larger than 0.35, the generated centroid closely approaches to the zero vector and thus does not work in the typical example selection. According to the best F_1 of each relation, we set the $T_c^{(i)}$ values to **0.15**, **0.3**, **0.25** and **0.25** for *Comp*., *Cont*., *Temp*., and *Expa*. respectively.

$p^{(i)}$	*Comp*. vs. other ($T_c^{(i)}$=0.15)	Cont. vs. other ($T_c^{(i)}$=0.3)	Temp. vs. other ($T_c^{(i)}$=0.25)	Expa. vs. other ($T_c^{(i)}$=0.25)
0.1	23.1(42.0)	0(75.7)	6.0(37.2)	0(45.0)
0.2	25.6(45.5)	0(75.7)	8.8(38.1)	0(45.0)
0.3	24.3(66.4)	29.8(58.7)	9.6(52.0)	0(45.0)
0.4	26.0(38.0)	39.0(24.3)	13.0(28.6)	0(45.0)
0.5	**27.9(39.4)**	**42.3(28.3)**	**13.4(45.5)**	**71.0(55.0)**
0.6	23.4(74.0)	39.0(24.3)	13.7(44.9)	62.5(52.3)
0.7	1.8(81.9)	39.0(24.3)	13.3(42.8)	57.8(54.1)
0.8	1.8(81.9)	38.9(25.1)	11.7(35.5)	0(45.0)
0.9	1.8(81.9)	38.1(24.5)	11.6(37.5)	0(45.0)

TABLE 4 – F_1 (Acc) with varying $p^{(i)}$ values ($T_c^{(i)}$ is fixed).

Next, with the tuned $T_c^{(i)}$ values, we inspect the performance of SCC with different $T_{dis}^{(i)}$ by tuning the value of $p^{(i)}$. Table 4 illustrates that almost all the classification reach their best performance at around $p^{(i)}$=0.5 where the threshold is the average of the minimum and maximum distances of the examples to the corresponding centroid. Then, in the following experiments, we set the $T_c^{(i)}$ values to **0.15**, **0.3**, **0.25** and **0.25** for *Comp*., *Cont*., *Temp*., and *Expa*., and all values of $p^{(i)}$ to 0.5.

At the same time, we observe the constituents of the best training data set generated by SCC for each relation. Table 5 illustrates the distributions of the final training set. From this table we can see that both the *IM* examples and *EX* examples contribute to the final typical example sets which is composed of 6753 artificial examples and 7816 real ones , According to Table 2 and Table 5, about 61.8 percent (7816/12632) of the *IM* examples and 47.8 percent (6753/14117) of the *EX* examples are typical. In the cases where the explicit discourse markers are absent, normally

richer linguistic features are involved to indicate the implicit discourse relations. For this reason, the real implicit examples tend to be typical.

	From EX_i	From IM_i	Total
$A_{Comp.}$	1293	852	2145
$A_{Cont.}$	1717	1418	3135
A_{Temp}	2090	404	2494
$A_{Expa.}$	1653	5142	6795
Total	6753	7816	14569

TABLE 5 – Constituents of the final typical sets.

5.3 Influence of Initial Seed Sets

The SCC algorithm begins with a seed set of typical examples that are picked out from the training data according to the manually summarized rules (denoted as the manual strategy) in section 4.2.1. The seed sets are generally composed of 1-5% of the corresponding relations.

stragegy	*Comp.* vs. other	*Cont.* vs. other	*Temp.* vs. other	*Expa.* vs. other
Manual	**27.9(39.4)**	**42.3(28.3)**	**13.7(44.9)**	**71.0(55.0)**
IM_seed	22.5(57.6)	39.8(47.4)	9.5(48.5)	50.9(47.4)
*EX*_seed	20.2(62.6)	39.0(24.3)	7.9(45.0)	55.2(50.0)
Random	19.1(75.7)	37.8(29.8)	8.0(27.9)	53.6(45.2)

TABLE 6 – F_1 (Acc) with different seed sets on Dev. Data.

stragegy	*Comp.* vs. other	*Cont.* vs. other	*Temp.* vs. other	*Expa.* vs. other
Manual	**28.5(62.0)**	**48.5(49.4)**	**14.7(69.0)**	**71.1(57.3)**
IM_seed	26.4(60.7)	41.9(26.5)	12.0(35.8)	52.6(49.2)
*EX*_seed	21.2(63.0)	41.9(35.4)	11.7(52.4)	54.6(50.1)
Random	22.2(47.1)	36.3(48.8)	11.1(40.2)	52.6(49.2)

TABLE 7 – F_1 (Acc) with different seed sets on Test Data.

For comparison purpose, we also examine the other three automatic seed set selection strategies on both development and test data. The results are shown in Table 6 and Table 7. We select the *IM* and *EX* data as seed set separately, denoted as *IM_seed* and *EX_seed* strategy respectively. With the Random strategy, we randomly select 10% of examples from the *EX* and *IM* data as the seed set for each relation. Both Table 6 and Table 7 show the superiority of the manual strategy over the other three. SCC to some extent is sensitive to the initialization of the typical set and could achieve a better performance with a better seed set of typical examples.

5.4 Evaluation of Implicit Relation Classifiers

We build four binary classifiers (*Comp.* vs Other, *Cont.* vs Other, *Temp.* vs Other, and *Expa.* vs Other) for relation labelling, and implement a 4-way classifier directly using the typical examples. All the 1046 implicit relations in the test data are used to compare our algorithm with the others.

Table 8 summarizes the performance implemented by Decision Tree (DT) and Naïve Bayes (NB) classifiers trained on different training sets in comparison with the state-of-the-art performance presented in Pitler et al. (2009), which solely uses the *IM* data to examine the influence of several linguistic features on implicit relation prediction. The second and third rows respectively show

Pitler's best results using single feature (Pitler-1) and combined features (Pitler-2), which are evaluated by a Naïve Bayes classifier. The IM_i, EX_i, EX_i+IM_i rows refer to our results of directly taking the *IM* data, the *EX* data, and both the *EX* and *IM* data as the training set respectively. Notice that all implementation of the IM_i method but feature selection is the same as Pitler's, though the performance of the IM_i method is far below Pitler's best results. This means feature selection is a key to promoting the performance.

		Comp. vs. Other	*Cont.* vs. Other	*Temp.* vs. Other	*Expa.* vs. Other	4-way
NB	Pitler-1	21.0(52.6)	36.7(62.4)	15.9(61.2)	71.3(59.2)	(65.4)
	Pitler-2	22.0(56.6)	47.1(67.3)	16.8(63.5)	76.4(63.6)	--
	IM_i	6.7(81.4)	41.9(28.0)	13.4(30.7)	44.4(51.9)	(51.3)
	EX_i	18.7(74.3)	40.1(27.6)	12.4(48.6)	8.2(46.6)	(34.1)
	EX_i+IM_i	14.0(76.5)	41.9(27.0)	12.7(44.8)	27.5(47.5)	(42.3)
	SCC	**24.3(58.3)**	**43.1(65.2)**	**18.0(92.2)**	**68.6(52.4)**	**(68.3)**
DT	IM_i	11.6(41.5)	38.7(40.5)	14.3(76.1)	38.8(44.7)	(53.5)
	EX_i	18.9(70.5)	41.9(26.5)	12.1(8.2)	0(46.8)	(42.6)
	EX_i+IM_i	14.0(76.5)	41.9(26.5)	9.0(67.3)	0(46.8)	(51.4)
	SCC	**28.5(62.0)**	**48.5(49.4)**	**14.7(69.0)**	**71.1(57.3)**	**(72.2)**

TABLE 8 – Performance comparison on PDTB.

SCC means using the training set which is composed of typical examples. Since the typical examples are picked out by SCC due to their distinct features, it is more suitable for the DT classifier to acquire the classifying rules according to the distinct features. That is why the performance of the DT classifier is better than that of the NB classifier in Table 8. The performance of both the DT and NB classifiers trained by typical examples are comparable to Pitler-1 and Pitler-2, though feature selection is not concerned in our systems. This table also shows that using typical examples as training data is more effective than using either IM_i, EX_i, or both IM_i and EX_i data as training set. For detecting the comparison relation with the DT classifier, the training set output by SCC significantly outperforms IM_i, by as much as about 17% absolute improvement in F_1-scores (i.e., 28.5 vs. 11.6). It is also observed that the performance of using IM_i as training set is comparable to that of using EX_i. This conforms to our assumption that typical examples contributes to the classification performance, while the final typical example set is composed of almost the same percent of the *IM* data and *EX* data according to Table 5.

According to the typical/atypical distribution in the training data, the test data should be composed of about 61.8% of typical ones and 38.2% of atypical ones. Since we do not preprocess the test data, the typical examples and the atypical ones in the test data are identified for their relations simultaneously. We observe the 4-way classification results with the DT classifier and find that most examples correctly identified are typical while the wrongly identified examples are usually atypical. For example, the third example in Section 1 is identified as *Expansion*.

5.5 Evaluation of Portability

To verify the portability of our method on RST-DT, we divide the whole RST-DT data into 347 training articles and 38 test articles. Different from PDTB, RST-DT includes about 18 relation types (RST-DT, 2002). To avoid data sparseness, we choose 4 relations that include a sufficient amount of examples. They are *Temporal*, *Contrast*, *Cause* and *Background*, and to some extent they are consistent with the 4 discourse relation types of PDTB. At the same time, we collect all

the 4 discourse relations spanning over individual sentences. Table 9 illustrates the relation distribution. For the 4 relations, we set $T_c^{(i)} = 0.25$ and $p^{(i)}=0.5$, SCC outputs the typical and atypical sets and their sizes are also given in the table.

Class	Training		Test	SCC	
	EXi	***IMi***	Implicit	A_i	B_i
Contrast	972	578	311	610	940
Background	701	677	330	660	718
Cause	304	785	535	846	243
Temporal	466	462	244	590	338

TABLE 9 – Relation distribution on RST.

Here, we evaluate the performance of SCC with the Decision Tree classifier. We compare it with the three baselines: real implicit examples (IM_i), artificial implicit examples (EX_i) or all the examples as training data (IM_i+EX_i). Table 10 shows that SCC can promote the performance with statistical significance (i.e., p-value[2]<0.1) on F_1. In addition, F_1 of *Contrast vs Other* (31.6) outperforms that of *Comparison vs Other* (28.5) on PDTB. It is the same for *Temporal*. According to our analysis, the reason is that the relations of RST-DT are fine-grained and it is relatively easy for SCC to obtain typical examples.

	Contrast vs. Other	*Background* vs. Other	*Cause* vs. Other	*Temporal* vs. Other
SCC	**31.6 (43.3)**	**38.3 (31.1)**	**54.8 (37.7)**	**31.2 (38.2)**
IM_i	27.6 (56.1)	34.8 (30.4)	35.6 (54.4)	29.2 (17.1)
EX_i	24.0 (64.9)	34.0 (41.5)	30.6 (57.1)	17.6 (67.0)
IM_i+EX_i	20.8(62.4)	34.9 (30.5)	32.2 (56.6)	27.2 (43.8)

TABLE 10 – F_1 (Acc) comparison on RST-DT.

Conclusions

In this paper, we for the first time present the typical/atypical perspective to select the most suitable training examples for implicit discourse relation recognition. A novel single centroid clustering algorithm is proposed to differentiate typical and atypical examples for each discourse relation. The experimental results show that the performance of the implicit relation classifiers with the typical examples selected as the training set are comparable to the best state-of-the-art methods on PDTB v2.0. In addition, the experiments on RST-DT show statistically significant improvements over the baselines and demonstrate the portability of our method. We will further explore more linguistic features and employ our approach on finer grained relation types. In SCC, we want to further investigate other distance formula. We also hope to explore the effective way to make use of the unlabelled discourse data.

Acknowledgments

The research work described in this paper has been partially supported by NSFC grants (No.61273278 and No.90920011), NSSFC grant (No: 10CYY023), National Key Technology R&D Program (No: 2011BAH10B04-03), and National High Technology R&D Program (No. 2012AA011101). We also thank the three anonymous reviewers for their helpful comments.

[2]Paired t-test is performed to compare the difference between SCC and *IM*, or between SCC and *IM+EX*. The *p*-values are 0.057 and 0.059 respectively.

References

Carlson, L., Marcu, D. and Okurowski, M.E. (2003). Building a discourse-tagged corpus in the framework of rhetorical structure theory. In Janvan Kuppelvelt and Ronnie Smith, editors, *Current Directions in Discourse and Dialogue*. Kluwer Academic Publishers.

Christiann, V. W. and Barnard., E. (2006). Data characteristics that determine classifier performance. *In Proceedings of the Sixteenth Annual Symposium of the Pattern Recognition Association of South Africa*, pp. 166–171, Parys, South Africa.

Feng, V. W. and Hirst, G., (2012). Text-level discourse parsing with rich linguistic features, *In Proc.of ACL'12*, pages 60-68.

Hernault, H., Prendinger, H., David, A., and Mitsuru I. (2010). HILDA: a discourse parser using support vector machine classification. *Dialogue and Discourse*, 1(3):1-33.

Lin, Z., Kan, M.–Y., and Ng, H. T. (2009). Recognizing implicit discourse relations in the Penn discourse treebank, *In Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, Singapore.

Mann, W. C. and Thompson, S. A. (1988). Rhetorical structure theory: towards a functional theory of text organization. *Text*, 8(3):243-281.

Marcu, D. and Echihabi, A. (2002). An unsupervised approach to recognizing discourse relations. *In Proc. of ACL 2002*, pages 368-375.

Pitler, E., Raghupathy, M., Mehta, H., Nenkova, A., Lee, A. and Joshi, A. (2008). Easily identifiable discourse relations. *In Proc. of the 22nd International Conference on Computational Linguistics (COLING08)*. pages 85-88.

Pitler, E., Louis, A. and Nenkova, A. (2009). Automatic sense prediction for implicit discourse relations in text, *In Proc. of the 47th ACL*. pages 683-691.

Prasad, R., Dinesh, N., Lee, A., Miltsakaki, E., Robaldo, L., Joshi, A. and Webber, B. (2008). The Penn discourse treebank 2.0. *In Proc. of the 6th International Conference on Language Resources and Evaluation (LREC)*. Marrakech, Morocco.

RST_DT. (2002). RST Discourse Treebank. Linguistic Data Consortium, http://www.ldc.upenn.edu/Catalog/CatalogEntry.jsp?catalogId=LDC2002T07

Saito, M., Yamamoto, K. and Sekine, S. (2006). Using phrasal patterns to identify discourse relations. In Proc. of the HLTCNA Chapter of the ACL. pages 133-136.

Sasha B.-G. (2007). Long-Answer Question answering and rhetorical-semantic relations. Ph. D. thesis, Columbia Unversity.

Soricut, R. and Marcu, D. (2003). Sentence level discourse parsing using syntactic and lexical information. *In Proc. of HLT/NAACL 2003*. pages 149-156.

Sporleder, C. and Lascarides. A. (2008). Using automatically labelled examples to classify rhetorical relations: An Assessment. *Natural Language Engineering*, 14:369-416.

Wang, W., Su, J. and Tan C. L. (2010). Kernel based discourse relation recognition with temporal ordering information, *In Proc. of ACL'10*. pages 710-719.

Wilson, T., Wiebe, J. and Hoffmann, P. (2005). Recognizing contextual polarity in phrase-level sentiment analysis. *In Proc. of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, pp. 347-354.

Wellner, B., Pustejovsky, J., Havasi, C., Rumshisky, A. and Sauri, R. (2006). Classification of discourse coherence relations: an exploratory study using multiple knowledge sources. *In Proc. of the 7th SIGDIAL Workshop on Discourse and Dialogue*. pages 117-125.

Yarowsky, D. (1995). Unsupervised word sense disambiguation rivaling supervised methods, *In Proc. of the 33rd annual meeting on Association for Computational Linguistics*, pages 189-196, Cambridge, Massachusetts.

Zhou, Z., Lan, M., Niu, Z. and Su, J. (2010). The effects of discourse connectives prediction on implicit discourse relation recognition, *In Proc. of SIGDIAL 2010: the 11th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 139–146.

Chinese Evaluative Information Analysis

Yiou Wang Jun'ichi Kazama Takuya Kawada Kentaro Torisawa
National Institute of Information and Communications Technology (NICT)
3-5 Hikaridai, Seika-cho, Soraku-gun, Kyoto 619-0289, Japan
{wangyiou, kazama, tkawada, torisawa}@nict.go.jp

Abstract

Together with the ever-growing amount of Chinese web data, the number of opinions voiced by Chinese users is rapidly increasing, and analyzing them is an important task. This paper introduces a Chinese Evaluative Information Analyzer (CEIA) and proposes a method to improve its performance. We use *evaluative information* as a unifying term for the information about attitudes, opinions, sentiments and so on. This paper makes three contributions: (i) CEIA can identify and analyze a more diverse and richer set of evaluative information than previous studies for Chinese; (ii) to implement the system, we constructed an original annotated corpus for Chinese evaluative information and built a large sentiment dictionary; (iii) we introduce syntactic dependency, semantic class and distance features to improve the evaluative information extraction. The performance of the system and the effectiveness of the newly introduced features are evaluated in a series of experiments on our Chinese evaluative information corpus.

Title and Abstract in Chinese

中文评价信息分析

随着网络的不断普及，中文用户在网络平台上发表的评价性文本的数量也在迅速增长，因而分析这些日益膨胀的评价性信息则成为一项重要的课题。本论文介绍了一个中文评价信息分析系统（CEIA）并提出新的方法来提高系统的性能。在这里我们用评价信息来统一表达关于态度，观点和情感等的评价性相关信息。本文的贡献主要体现在以下三个方面：（i）与以往研究相比，CEIA可以识别和分析更加多种多样，更加丰富的评价信息；（ii）为了实现CEIA，我们标注了一个中文评价信息语料并且构筑了一个大规模的情感词典；（iii）我们引入了句法依存特征，语义聚类特征和距离特征来提高评价信息抽取的性能。通过在中文评价信息语料库上一系列实验，给出了系统的整体性能，并验证了新方法的有效性。

Keywords: Chinese Evaluative Information, Opinion Mining, Sentiment Analysis.

Keywords in Chinese: 中文评价信息，观点挖掘，情感分析.

Proceedings of COLING 2012: Technical Papers, pages 2773–2788,
COLING 2012, Mumbai, December 2012.

1 Introduction

To automatically find or track the attitudes, feelings and evaluations in texts, opinion mining and sentiment analysis have been extensively studied from different perspectives (Pang and Lee, 2008). With the ever-growing number of Chinese users (over half a billion users only in mainland China), the amount of web opinions in Chinese is rapidly increasing, and analyzing them is an important task. However, research and resources about the Chinese opinion analysis lag behind those for extensively studied languages, such as English. Therefore, opinion analyzers, which can deal with Chinese web data of a great variety of topics and styles, are especially in great need.

To meet this requirement, we introduce a Chinese Evaluative Information Analyzer (CEIA) that can mine a wide variety of evaluative information from Chinese web documents. We use *evaluative information* as a unifying term for the information concerning attitudes, opinions and sentiments, and so on, which is useful to provide a view of evaluation.

The system automatically analyzes Chinese evaluative information through the following processes: (1) *extracts evaluative expressions*; (2) *identifies evaluation holders*; (3) *extracts evaluation targets*; (4) *determinates evaluation types*; (5) *determinates the sentiment polarities of the evaluative expressions*.

CEIA has the following two characteristics:

Firstly, CEIA can analyze a more diverse and richer set of evaluative information than the previous studies for Chinese. The previous research on Chinese opinion analysis focuses on subjective expressions (*opinionated sentences*) (Liu, 2010), as in the Multilingual Opinion Analysis Task (MOAT) of NTCIR (Seki et al., 2010). However, some objective expressions that describe positive or negative facts are also informative in that they express some kinds of evaluations. Also, requests are some kinds of representations of opinions or attitudes. Consider the following sentences,

1. *Many people are using mobile phone A.*
2. *The users hope company A will offer them a security lock function.*

The sentence 1 suggests that "mobile phone A" is popular and has been chosen by many people. The sentence 2 claims that the company A does not offer a security lock function now and the user request the company to offer it. In some sense, this sentence also includes the evaluation or unsatisfied feelings of the users. We want to consider such cases as "implicit" evaluations for "mobile phone A" and "company A", in addition to subjective expressions such as "I love mobile phone A".

To the best of our knowledge, this is the first paper that treats the above implicit evaluations in Chinese evaluative information analysis. Implicit evaluations have been considered by Nakagawa et al. (2008) for Japanese. They presented the study about extracting subjective and objective Japanese evaluative expressions from the web and their work was used in WISDOM system (Akamine et al., 2010) [1] , and shown to be useful to support users' judgement of information credibility. Inspired by their work, we adopt the task definition and expand the research scope of Chinese evaluation information analysis.

Secondly, CEIA can deal with the data in diverse topics and writing styles. The existing studies about Chinese opinion analysis are domain-limited. For example, Chinese Opinion Analysis Evaluation (COAE) (Zhao et al., 2008) mainly deals with opinion analysis of reviews. MOAT (Seki et al.,

[1] http://wisdom-nict.jp/

2010) deals with the analysis of news articles, which are written in a formal writing style. To make our system more robust to the web data of a great variety of topics and styles, we constructed an original annotated Chinese evaluative information corpus whose sentences are extracted from web pages of wide range of topics and styles. CEIA consists of many machine learning modules such as CRFs and SVMs and the corpus was used to train these modules, resulting in a robust evaluative information analyzer.

To achieve high system performance is also a primal goal of evaluative information analysis. In this work, we introduce new features to improve the performance. Specifically, syntactic dependency features, semantic class features and distance features are added to the baseline models. To demonstrate the performance of our system and the effectiveness of our new features, we conducted a series of experiments on the Chinese evaluative information corpus.

2 CEIA

In this section, we describe the entire picture of CEIA and the resources for the system. We first introduce the specifications of the evaluative information on which this study is focused, and then we explain how an evaluative information corpus is constructed. Finally, we explain each process of CEIA in detail.

2.1 Evaluative Information

There is a wide variety of evaluative information on the web, such as reviews of products and criticisms of policies. The information reflects various perspectives of individuals or organizations. Research on evaluative information analysis are conducted from different points of views and at different levels of granularity (Kobayashi et al., 2004; Kaji and Kitsuregawa, 2006; Liu, 2010; Pang and Lee, 2008; Akamine et al., 2010). In this section, we describe the specifications of evaluative information on which this study is focused.

We analyze the evaluative information at a fine-grained level. We use a 5-tuple that consists of (1) an evaluative expression, (2) an evaluation holder, (3) an evaluation target, (4) an evaluation type, and (5) sentiment polarity as the basic unit of evaluative information and call it an *evaluative information set*. Each item is defined as follows.

Evaluative expression is a span of text that describes the evaluation. It can be a single word, a multi-word expression, or a sentence.

Evaluation holder is a person, a group or an organization that expresses the evaluation.

Evaluation target is a thing, a matter, or an entity about which the evaluation was expressed.

Evaluation type is the category to which the evaluative expression belongs. It will be explained in detail in the following subsection.

Sentiment polarity indicates whether the evaluation expression for the evaluation target is positive or negative from the viewpoint of the evaluation holder. For some cases, it may differ from the polarity of the whole sentence. For examples, *Mike strongly objected to the war.* Although the entire sentence is not negative, the sentiment polarity of evaluative expression "*strongly objected to*"is negative. That is to say, the evaluation holder "*Mike*" has a negative opinion on the evaluation target "*war*". From this point of view, we consider the sentiment polarity in the connection with specific evaluation holders and evaluation targets at fine-grained levels.

2.1.1 Evaluation Type and Sentiment Polarity

There are various kinds of evaluative expressions such as approving or opposing attitudes, description of merits or desirable events, and so on. To clarify the scope of evaluations that we address in this study,we classify evaluative expressions into several categories. Such categorization is also helpful for further use of evaluative information.

Following the work of Akamine et al. (2010), we use the following evaluation types. Each type, except for "Request", has sentiment polarities: positive (+) or negative (−). We use underline to show evaluation targets, boldface for evaluative expressions, and italics for the evaluation holders.

- Emotion+/−: an expression that expresses human feelings or emotions.
  e.g., *XiaoLi* **is not interested in** product A. (Emotion−)
- Comment+/−: an expression that expresses approval/disapproval or praise/criticism.
  e.g., *Mike* said that movie A **is one of the best he has ever seen**. (Comment+)
- Merit+/−: an expression that cites good points/shortcomings or merits/demerits.
  e.g., Drug A **starves and kills cancer cells**. (Merit+)
- Event+/−: an expression that describes good/bad events, desirable/undesirable experience.
  e.g., Camera X **broke just three days after I bought it**. (Event−)
- Adoption+/−: an expression that shows adoption, promotion or rejection.
  e.g., **Nobody bought** Mike's ebook. (Adoption−)
- Request: an expression that expresses proposals, obligations, advices, hopes or requests.
  e.g., *The users* **hope Company A can offer them a security lock function**. (Request)

2.2 Chinese Evaluative Information Corpus

To train our system and analyze a wide variety of evaluative information, we constructed an evaluative information corpus which consists of Chinese sentences extracted from web pages of wide range of topics and styles. We chose 66 topics which relate to things we use in daily life, controversial policies, movie reviews and so on. The followings are the steps for the corpus construction:

(1) Use the topic as the keyword and search documents using a Web search engine.

(2) Collect HTML files of 900 web pages from the retrieval results for each topic. Specifically, the first 300 pages in the retrieval results from forum sites, the first 300 pages from blogs and the first 300 pages from general sites.[2] In this way, the corpus can cover different writing styles and reflect more diverse perspectives.

(3) Randomly choose candidate sentences that include topic keywords from the above files. For each topic, we randomly collected 200 sentences, and for each sentence, we provided context information (the previous two sentences and subsequent two sentences) for annotation reference.

(4) Trained annotators judged whether a sentence contained any evaluative expressions or not. If the sentence contained evaluative expressions, the annotator annotated the evaluation holders, the text spans of the evaluative expressions, the text spans of the evaluation targets, the evaluation types and the sentiment polarities. That is to say, an evaluative information set was annotated for each evaluation expression. For evaluation holder annotation, if the writer is the evaluation holder, [*author*] is annotated as the holder. If the holder is neither explicitly written in the sentence

[2]We suppose the URL including "forum", as the web pages from forum sites, the URL including "blog", as the web pages from blog sites, and the rest are general sites, although it may include some noise.

Dictionary Origin	Positive	Negative	Postive + Negative
JSD	6,270	19,394	25,664
Giga-word	1,977	770	2,747
Total	8,247	20,164	28,411

Table 1: The statistics of sentiment dictionary

nor is the writer, [*undefined*] is annotated. When annotating the current sentences, the annotator could refer to its context information. In some cases, one sentence may contain multiple evaluative targets or multiple evaluation expressions, and then multiple evaluative information sets must be annotated. For example, *Mike said that Movie A is great but it is not better than Movie B which is the best movie he has seen.* Two evaluative information sets should be annotated: (1) (is great, Mike, Movie A, Comment+) and (2) (is the best movie he has seen, Mike, Movie B, Comment+). Note that *it (Movie A) is not better than Movie B* is a comparative expression. we do not deal with the comparative sentences that do not show clear sentiment polarities at present.

The total number of sentences in the corpus was 6,680. There were 5,111 evaluative information sets in the corpus. It took 380 man-hours to construct the entire corpus.

2.3 Sentiment Dictionary

A sentiment dictionary is a set of words and their polarities (for example, [break a record, +], [break the law, −]). Such a dictionary is a fundamental resource for evaluative information analysis. We built a Chinese sentiment dictionary in the following way.

(1) Since it is time-consuming to built a dictionary without any reference, we semi-automatically translated an existing Japanese sentiment dictionary (JSD)[3] to Chinese. We mapped the entries of JSD with a Japanese-Chinese bilingual dictionary, and obtained Chinese translations and their polarities transferred from Japanese entries. Unmapped entries were translated by human. The resulting Chinese entries and polarities were finally manually checked. There were 36,981 entries in JSD (9,030 positives and 27,951 negatives) , and we obtained 25,664 entries for Chinese.

(2) So that the dictionary covers the frequently used polarity-bearing words, we also auto-segmented and tagged the XIN_CMN portion of Chinese Gigaword Version 2.0 (LDC2009T14), which has approximately 311 million words, and collected adjectives (with POS tags "VA" and "JJ") and idiom candidates with high frequency. We removed the overlap between the words collected from JSD, and manually checked the rest of the candidates, and tagged them with polarity.

Finally, we build a Chinese sentiment dictionary with 28,411 entries. Its detailed statistics are shown in Table 1. It is used in evaluative expression extraction and polarity classification models.

2.4 CEIA System

CEIA flow is shown in Figure 1. First, the user inputs raw sentences; second, the system (1) extracts the evaluative expression from the input sentences, (2) identifies the evaluation holder, (3) extracts the evaluation target, (4) categorizes the evaluation type and (5) determinates the sentiment polarity. Finally the results from these processes are summarized and displayed as output to user. The rest of this section describes the above processes in detail.

[3]The dictionary is distrubuted only to the member of the ALAGIN forum (http://alaginrc.nict.go.jp). It is for the freely available package of opinion extraction tool, which can be obtained from http://alaginrc.nict.go.jp/opinion/index_e.html

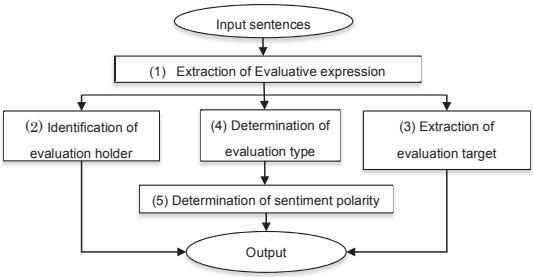


Figure 1: CEIA flow

Type	Feature	Description
Word feature	$w_{i-2}, w_{i-1}, w_i, w_{i+1}, w_{i+2}$, $w_{i-1}\&w_i, w_i\&w_{i+1}$	Word surfaces of the previous but one, previous, next, and next but one words; word surface bigram of the previous (next) word and the current word.
POS tag feature	$t_{i-2}, t_{i-1}, t_i, t_{i+1}, t_{i+2}$, $t_{i-1}\&t_i, t_i\&t_{i+1}$	POS tags of the previous but one, previous, next, and next but one words; POS tag bigram of the previous (next) word and the current word.
Polarity feature	$p_{i-2}, p_{i-1}, p_i, p_{i+1}, p_{i+2}$, $p_{i-1}\&p_i, p_i\&p_{i+1}$	The word polarities of the previous but one, previous, next, and next but one words; word polarity bigram of the previous (next) word and the current word.

Table 2: Feature templates for evaluative expression extraction

2.4.1 Extraction of Evaluative Expressions

The goal of this process is to identify the words, phrases or sentences that express the evaluations in the text. We use the sequence tagging method with the BIO tag-set, which was initially used for opinion extraction by Breck et al. (2007). In the method, each word is tagged with one of three types of labels based on its position in the evaluative expressions: (B) beginning of an evaluation expression, (I) inside of an evaluation expression or (O) outside of an evaluation expression. For example, for the sentence, "*The chief editor* **really loves** book A." , the BIO tags are encoded in the following way:

The/O chief/O editor/O **really/B loves/I** book/O A/O ./O

We employ the linear chain CRFs (Lafferty et al., 2001) as our learning model for BIO tagging. Specifically, we use CRF++ (version 0.54) implementation by Taku Kudo. [4]

The features shown in Table 2 are used in the CRF for the i-th word in a sentence. Here, w_i, t_i, and p_i denote the current word surface, the part-of-speech tag and the polarity of the i-th word in the input sentence, respectively. A word's polarity is obtained from the sentiment dictionary. To search the word in the dictionary, we use forward maximum matching. We generate the above features with the unigram template of CRF++ (i.e., as the combination with the output tag at the current position, o_i). We also use the tag bigram feature (i.e., $o_{i-1}\&o_i$) .

2.4.2 Extraction of Evaluation Targets

The evaluation target is extracted from a sentence that contains the evaluative expression with a BIO tagging method using a CRF, as in the extraction of evaluative expressions. We use the same word feature and POS tag feature as in evaluative expression extraction and introduce position

[4] http://crfpp.sourceforge.net/

Type	Feature	Description
unigram p	$w_1, t_1, w_2, t_2, ...w_s, t_s,$	For the words previous to the evaluation expressions, the word and POS tag unigrams are added as type-p unigram features
unigram x	$w_{s+1}, t_{s+1}, ...w_{s+n}, t_{s+n},$	For the words in the evaluation expressions, the word and POS tag unigrams are added as type-x unigram features
unigram n	$w_{s+n+1}, t_{s+n+1}, w_{s+n+2}, t_{s+n+2}, ...w_l, t_l,$	For the words next to the evaluation expressions, the word and POS tag unigrams are added as type-n features
bigram	$w_{s+1}\&w_{s+2}, t_{s+1}\&t_{s+2}, ... w_{s+n-1}\&w_{s+n}, t_{s+n-1}\&t_{s+n}$	For the words in the evaluation expressions, the word bigram and POS tag bigram features are added
category	$c_i\&w_{i-1}, c_i\&w_i, c_i\&w_{i+1}, c_j\&w_{j-1}, c_j\&w_j, c_j\&w_{j+1}..$	For the words in the evaluative expressions, the category and word bigram feature are added: the category and the previous word bigram, the category and current word bigram and the category and the next word bigram

Table 3: Feature templates for evaluation type determination

features. The position feature setting is $\{e_{i-2}, e_{i-1}, e_i, e_{i+1}, e_{i+2}, e_{i-1}\&e_i, e_i\&e_{i+1}\}$. Here, e_i is a flag that expresses the position of w_i with respect to the evaluative expression. If w_i is previous to an evaluative expression, then e_i is "p"; if w_i is in an evaluative expression, then e_i is "x"; and if w_i is next to an evaluative expression , then e_i is "n". For example, for the sentence, " *The chief editor* **really loves** book A." , the e_i is encoded in the following way. If no holder was found by the CRF model, [*undefined*] was set as the evaluation target of the current evaluation expression.

The/p chief/p editor/p really/x loves/x book/n A/n ./n

2.4.3 Determination of Evaluation Types

We predicted the evaluation types using one-versus-rest multi-class linear kernel support vector machines (SVMs). We used the features shown in Table 3 for SVMs. Here w_i, t_i and c_i denote the word surface, the part-of-speech tag and the type category of the i-th word, respectively. l, n and s denote the number of words in the input sentence, the number of words in the evaluative expression and the number of words previous to the evaluative expression in the input sentence, respectively.

A word's type category is obtained from a type category dictionary. In the investigation of evaluation types, which has been described in Section 2.1.1, we found that each evaluation type has some characteristic words. Therefore we manually listed such characteristic words for each evaluation type and generated a type category dictionary, which includes 141 entries, for example, [希望(hope), Request], [憎恨(hate), Emotion], [称赞(praise), Comment]. For words in the evaluative expressions, the category feature can be generated only when the word is in the type category dictionary. The category feature can provide some improvement in performance according to our preliminary experiments.

2.4.4 Identification of Evaluation Holders

While the evaluation holders are sometimes stated explicitly in sentences where the evaluative expressions are contained, in more than half of the cases in Chinese, they are not clearly stated in the sentence. When evaluation holders are not expressed in the sentence, the evaluation holder is usually the information sender, i.e., the "author" in other words. Therefore, we consider that the opinion holder identification consists of a classification task and an information extraction task. That is, the evaluation holder is identified in two steps in CEIA: (1) use linear kernel support vector machines (SVMs) to determine whether the evaluation holder is *author* or *not-author*; (2) if the evaluation holder is not the author, then use a CRF tagging

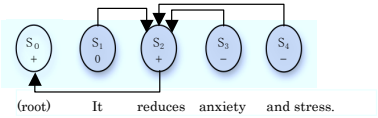


Figure 2: Example of CRFs with hidden variables

model to extract the evaluation holder for each evaluative expression. For the SVM model, in addition to the same features in types of unigram p, unigram x, unigram n and bigram as in Section 2.4.3, we also use the bigram p $\{w_1\&w_2, t_1\&t_2 ... w_{s-1}\&w_s, t_{s-1}\&t_s\}$ and bigram n $\{w_{s+1+1}\&w_{s+n+2}, t_{s+n+1}\&t_{s+n+2} ... w_{l-1}\&w_l, t_{l-1}\&t_l\}$ features to add the bigram information for the words previous to the evaluation expressions and the words next to the evaluative expressions. For CRF model, we use the same features as in Section 2.4.2. If no holder was found by the CRF model, [*undefined*] was set as the evaluation holder of the current evaluation expression.

2.4.5 Determination of Sentiment Polarity

A typical approach for sentiment classification is to use supervised machine learning algorithms with bag-of-words as features (Pang et al., 2002). However, this method cannot consider syntactic structures that seem essential to infer the polarity of a whole sentence. We follow the work of Nakagawa et al. (2010) and use a dependency tree-based method, which was demonstrated to perform better than other methods based on bag-of-words in both English and Japanese sentiment classification tasks. The sentiment polarity is classified using conditional random fields (CRFs) with hidden variables. In the method, the sentiment polarity of each dependency subtree, which is not observable in training data, is represented by a hidden variable. The polarity of the whole sentence is calculated by considering the interactions between the hidden variables. For example in Figure 2, each phrase (indicated by a circle) in the polarity-bearing sentence/expression has a random variables. The random variable represents the polarity of the dependency subtree whose root node is the corresponding phrase. Two random variables are dependent if their corresponding subtrees have head-dependent relations (indicated by an arc). Usually the polarity is labeled in expression/sentence level in the annotated corpus, and subtrees are not labeled, so all the random variables except for the root node are hidden variables (indicated by gray circles). In the model, if a head word tend to reverse the polarity of the dependent word, reversal polarity feature can be used. That is to say, it can deal with the reversal of sentiment polarities caused by polarity shifting words. For example, the "reduce" in the example is polarity shifting word. "Reduce anxiety" is positive, while "anxiety" is negative. In order to deal with the polarity shifting, 179 Chinese polarity shifting words were collected and used in the CEIA. As for the features, we used the same features as those in Nakagawa et al. (2010).

3 New Features

In this section, we describe our approach that effectively employs the dependency information, semantic class and distance information into the above evaluative information extraction (specifically evaluative expression extraction and evaluation target extraction).

3.1 Dependency Features

The use of syntactic or deep linguistic features has been tried in opinion analysis in the literature. Johansson and Moschitti (2010) demonstrated that the features derived from grammatical and

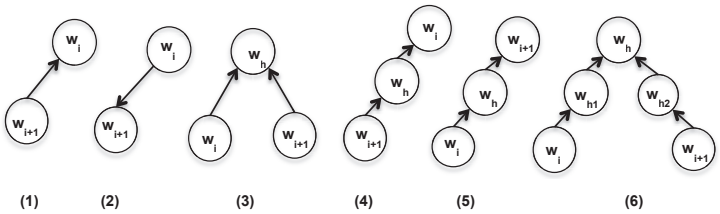


Figure 3: Different dependencies between w_i and w_{i+1} that can be linked by one or two arcs

semantic role structure can be used to improve the detection of opinionated expressions in subjectivity analysis. However, based on their evaluation, the precision decreases while the F-measure is increased. In addition, they claimed that a sequence tagging model cannot be used when using syntactic features, and they used reranking method, which will slowdown the processing. We introduce a simple dependency features for our tagging model that can be generated with the help of a Chinese dependency parser for evaluative information extraction.

Using a dependency parser, two kinds of dependency information can be obtained:

(i) *head* : the head of the current word, which is either a value of word ID, or zero ('0') if the word is the root node of the sentence.
(ii) *dependency relation*: the dependency relation of the current word to the head. The dependency relation is presented by the dependency labels: SBJ, OBJ, PRD, NMOD, VMOD, etc. The labels show function categories, such as the subject, object, predicate and so on.

We introduce the following two kinds of dependency features:

(i) *dependency head feature*: this feature is generated from the head information. The head-dependent relations between neighboring words w_i and w_{i+1} that can be linked by one or two arcs or can be linked to the same head by the same number (one or two) of arcs are summarized in Figure 3. We encoded the head-dependent relation into a new type of feature. We tried several feature representations and found that the features derived from the following method were most effective. We categorized the head-dependent relation between w_i and w_{i+1} into four groups:

- Near head-dependent relation (NH): the cases of (1), (2) and (3) in Figure 3.
- Medium head-dependent relation (MH): the cases of (4), (5) and (6) in Figure 3.
- Last word (LW): if w_i is the last word of the sentence/expression.
- Far head-dependent relation (FH): all the possible dependencies except for the above three groups.

The new features of $deph_i$ and $w_i\&deph_i$ are added for the i-th word in a sentence. Here $deph_i$ is head-dependent relation group of w_i and w_{i+1} , labeled with NH, MH, LW or FH. We suppose that such labels encode the syntactic distance information. For example, although w_i and w_{i+1} is the neighborhood in a sentence, they are distant syntactically, if the head-dependent relation group is labeled with FH.

(ii) *dependency relation feature*: this feature is generated with the information of the dependency relation. The dependency relation feature setting for evaluative expression extraction is

$\{depr_i, t_i\&depr_i\}$. Here, $depr_i$ is the dependency label of the relation between w_i and w_i's head in a sentence. Since the grammatical information is very important for evaluation target extraction, new features of { $depr_{i-2}$, $depr_{i-1}$, $depr_i$, $depr_{i+1}$, $depr_{i+2}$, $depr_{i-1}\&depr_i$, $depr_i\&depr_{i+1}$} are added for evaluation target extraction. With these features, the grammatical function information can be encoded in both the evaluative expression and evaluation target extraction tasks.

3.2 Semantic Class Features

The idea of combining semantic classes of words with discriminative learning has been previously reported in the context of named entity recognition (Miller et al., 2004; Kazama and Torisawa, 2008), dependency parsing (Koo et al., 2008) and Chinese word segmentation and POS tagging (Wang et al., 2011). We adopt and extend these techniques to evaluative information analysis and demonstrate their effectiveness in this task.

We produced the semantic classes of various levels of granularity, by using the Brown cluster hierarchy (Brown et al., 1992) at various lengths. Note that a semantic class is represented by a bit string that reflects the branching of the semantic class hierarchy.

We designed two kinds of semantic class features:
(i) full string feature: full string of the semantic class for w_i;
(ii) 6-bit prefix feature: 6-bit prefix of the semantic class for w_i.

3.3 Distance Feature

The target extraction task is to extract a target for a given evaluative expression. In most cases, the evaluation target and the evaluative expression are near to each other. Therefore, we add the distance label between w_i and the evaluative expression as a new feature for evaluation target extraction. The distance labels are defined in the following way: we first compute the distance d between w_i and the evaluative expression in word count; then when d is larger than 10, the distance label is "L"; otherwise if w_i is on the lefthand side of the evaluative expression, the distance label is d; and if w_i is on the righthand side of the evaluative expression, the distance label is $-d$. The feature setting of distance feature is $\{dis_{i-2}, dis_{i-1}, dis_i, dis_{i+1}, dis_{i+2}, dis_{i-1}\&dis_i, dis_i\&dis_{i+1}\}$. Here, dis_i is the distance label of w_i . With these feature, the position information with regard to the evaluative expression can be encoded.

4 Experiments

We evaluated the performance of the CEIA system and the effect of the new features.

4.1 Experimental Setting

We used the Chinese evaluative information corpus described in Section 2.2 as the training and test sets and performed 10-fold cross validation experiments on the corpus.

To conduct the experiments, we used the Chinese morphological analyzer described in Wang et al. (2011) and a Chinese dependency parser (CNP) [5] to obtain the Chinese word segmentation, part-of-speech tags and dependency information.

To generate the semantic classes of words, we used the XIN_CMN portion of Chinese Gigaword Version 2.0 (LDC2009T14), which has approximately 311 million words, as a large raw data and set the number of classes to 1000.

[5] http://alaginrc.nict.go.jp/cnp/index.html

We use the following measures to evaluate the performance of the system:

Recall (R) : ratio of correctly extracted evaluative expressions/targets/holders to the number of expressions/targets/holders in the gold standard corpus.
Precision (P): ratio of correctly extracted evaluative expressions/targets/holders to the number of expressions/targets/holders in system's output.
F-measure (F): harmonic mean of recall and precision.
Accuracy (Acc): ratio of the number of correct system output to the number in the gold standard. The accuracy of each tasks is defined as follows:

Accuracy of evaluation type determination: ratio of correctly identified evaluation types to the number of evaluative expressions in the gold standard corpus.
Accuracy of evaluation polarity determination: ratio of correctly classified sentiment polarities to the number of evaluative expressions of polarity-bearing evaluation types in the gold standard corpus.

To calculate the recalls, precisions and F-measures of the evaluative expressions and the evaluation targets, we use the following three criteria:

Exact match: extracted expression/target/holder is regarded as correct if it exactly matches the gold standard.

Partial match: extracted expression/target/holder is regarded as correct if it overlaps the gold standard's one. Our partial match is different from the overlap-based precision and recall measures in Breck et al. (2007). A potential issue with their overlap-based precision and recall is that the measures may drastically overestimate the system's performance as follows: a system predicting the whole sentence as an extracted expression would achieve 100% overlap-based recall and precision, if the gold standard contains any evaluative expression. In order to avoid this problem, we deal with the duplicate matches as follows: an extracted expression is only counted as overlapping with the first gold standard one, even if it can overlap with more than one gold standard's ones. From this point of view, our metric is stricter than in Breck et al. (2007).

Span partial match: this evaluation metric takes the span coverage of extracted expression/target/holder with respect to the span of the gold standard's one into consideration. We define this metric by refining the soft precision and recall described in Johansson and Moschitti (2010). First the span coverage c of a span s with respect to another span s', which measures how well s' is covered by s , was defined: $c(s,s') = \frac{|s \cap s'|}{|s'|}$. In this formula, the operator $|*|$ counts tokens (Chinese characters), and the intersection $\cap$ represents the overlap of the two spans. Then, if two spans overlapped, instead of adding "1" to the number of correctly extracted expression as in partial match, we add the span coverage to the number of correctly extracted expression. For example, if the gold standard evaluative expression had 8 tokens and 6 tokens of extracted expression overlapped with the gold standard, then we consider $3/4$ of the expression is correctly extracted. We deal with the duplicated matches in the same way as in partial match to avoid the overestimation. Although Johansson and Moschitti (2010) tried to alleviate the overestimation problem with their soft precision and recall, their measure still tend to reward long spans in recall [6] and overestimate the precision in some cases. Our metrics solved both the overestimation in recall and precision. Our metric is bounded below the exact match and above the partial match.

[6] a system predicting the whole sentence as an extracted expression would achieve 100% soft recall in Johansson and Moschitti (2010)

Task	Exact match	Partial match	Span partial match
Evaluative expression extraction	R=0.1730 P=0.2933 F=0.2176	R=0.4560 P=0.7728 F=0.5734	R=0.3934 P=0.6264 F=0.4832
Evaluation target extraction	R=0.4171 P=0.6530 F=0.5089	R=0.5442 P=0.8521 F=0.6640	R=0.5226 P=0.7934 F=0.6300
Evaluation holder identification	R=0.7455 P=0.9630 F=0.8401	R=0.7518 P=0.9714 F=0.8474	R=0.7509 P=0.8672 F=0.8047
Evaluation type determination	Acc = 0.5787	-	-
Evaluation polarity determination	Acc = 0.8146	-	-

Table 4: The performance of CEIA

4.2 Performance of CEIA

The performance of the entire CEIA system is shown in Table 4. The figures are for the best combination of the features, which will be described later. The performance of each task was evaluated independently. For example, for sentiment polarity determination task that determine the polarity of the evaluative expressions, the input evaluative expressions are the gold standard ones rather than the system output of the evaluative expression extraction task. For the evaluative expression, the performance of the exact match seems to be low. This is because it is difficult to detect the exact span of an evaluative expression. The evaluative expression detection in English also came to such situation and most work use partial match measures (Johansson and Moschitti, 2010). The performance of our system for partial match is reasonably good. Although the recall was not high, to extract information from a large amount of raw data, such as billions of web documents, we believe that the precision is a very important metric. The precision of the evaluative information extraction is 0.77. With such a relatively high precision, we suppose the evaluative expression extraction can play an active role in the actual application.

We also compared our system with the other works or systems reported in the literature, which are in the close task definition, although it is not fair to compare directly, because we deal with different languages and use different test sets. We just use their work as a reference to show that our system provided a reasonable result, when dealing with the same task in different language contexts.

As for sentiment polarity determination, we follow the work of Nakagawa et al. (2010). Their method was shown to perform better than other methods based on bag-of-words and provided accuracies ranging from 0.861 to 0.773 for a series of Japanese and English test sets. Because our test sets include various topics, this complicates the polarity classification task. Since our classification accuracy was 0.8146, we can say that Nakagawa et al. (2010)'s model also works well for Chinese.

As for the evaluative expression extraction, as we mentioned in the Section 1, Nakagawa et al. (2008) extracted subjective and objective Japanese evaluative expressions from the web. The result of their system with exact match is shown in Table5 . The performance scores are directly taken from their paper. The result indicates the difficulty of this task. The performance of our system is better than their work.

As for evaluation target extraction and evaluation holder identification, Multilingual Opinion Analysis Task (MOAT) of NTCIR-8 (Seki et al., 2010) included these tasks. Table 5 shows the best results with lenient match in opinion holder and opinion target identification tasks of simplified

work	Nakagawa et al. (2008)	MOAT of NTCIR-8	
Task	Evaluative Expression Extraction	Target Identification	Holder Identification
Recall	0.12	0.564	0.792
Precision	0.22	0.735	0.877
F	0.15	0.638	0.832

Table 5: Performance of previous works

Method	Exact match			Partial match			Span partial match		
Measure	R	P	F	R	P	F	R	P	F
Baseline	0.1628	0.3005	0.2110	0.4155	0.7678	0.5388	0.3599	0.6229	0.4557
Baseline+class (6-bit prefix)	0.1715	0.3018	0.2186	0.4386	0.7718	0.5593	0.3786	**0.6289**	0.4724
Baseline+class (full string)	**0.1746**	0.3005	**0.2208**	0.4474	0.7696	0.5655	0.3874	0.6257	0.4784
Baseline+dependency head	0.1688	0.3010	0.2162	0.4319	0.7695	0.5531	0.3735	0.6278	0.4681
Baseline+dependency relation	0.1686	**0.3036**	0.2167	0.4276	0.7699	0.5496	0.3709	0.6281	0.4660
Baseline+all features	0.1730	0.2933	0.2176	**0.4560**	**0.7728**	**0.5734**	**0.3934**	0.6264	**0.4832**

Table 6: Performance of new features in evaluative expression extraction

Chinese in MOAT. The results are directly taken from Seki et al. (2010). Although we use different test set and cannot compare the results directly, we can conclude that our system's F-measure is competitive with the systems that deal with a similar task.

4.3 Effect of New Features

We added the new features described in Section 3 to the evaluative expression extraction and target extraction models and performed 10-fold cross validation experiments to evaluate their effectiveness. We also tested the new features for evaluation holder extraction. However we omit the results here because the improvement by the new features was slight.

Table 6 shows the performance of the new feature in the evaluative expression extraction. Here, "all features" is the result of the combination of all the features. As mentioned in Section 4.2, to exactly identify the span of the evaluation is very difficult. Thus, we mainly refer to the results measured by partial match and span partial match here. Dependency features achieved an improvement in both recall and precision. The dependency features that introduced by Johansson and Moschitti (2010) only showed positive effect on recall with their soft partial match measure and partial match. Our span partial match and partial match are stricter measures than theirs. Note that we also evaluated our dependency head and dependency relation features use their soft precision and recall. There was no decrease in both soft precision and recall. In this point, our dependency features was comparably effective. Furthermore, our method uses the dependency features in sequence tagging model and is simpler than their method. The results also show that the full string semantic class features were the most effective ones and that a combination of four types of features achieves the best performance in F-measure. This suggests that these features are relatively independent in feature characteristics.

Table 7 shows the performance of the new features in the evaluation target extraction. The results show that the semantic class feature shows less effect in target extraction task than in evaluative expression extraction task and distance feature were the most effective one. 6-bit prefix features achieved an improvement in partial match. While the dependency head feature did not show a positive effect on the recall , it achieved best results on precision. Dependency relation features and distance features had positive effect for both recall and precision. The combination of all

Method	Exact match			Partial match			Span partial match		
Measure	R	P	F	R	P	F	R	P	F
Baseline	0.4040	0.6643	0.5021	0.5143	0.8454	0.6391	0.4942	0.7960	0.6094
Baseline+class (6-bit prefix)	0.4026	0.6595	0.5000	0.5186	0.8501	0.6440	0.4977	0.7983	0.6129
Baseline+class (full string)	0.4058	0.6546	0.5008	0.5267	0.8494	0.6495	0.5051	0.7933	0.6169
Baseline+dependency head	0.4039	**0.6753**	0.5052	0.5135	0.8590	0.6425	0.4930	**0.8083**	0.6122
Baseline+dependency relation	0.4074	0.6710	0.5069	0.5219	0.8594	0.6491	0.5006	0.8072	0.6177
Baseline+distance	0.4135	0.6717	**0.5117**	0.5290	**0.8597**	0.6548	0.5073	0.8081	0.6232
Baseline+all features	**0.4171**	0.6530	0.5089	**0.5442**	0.8521	**0.6640**	**0.5226**	0.7934	**0.6300**

Table 7: Performance of new features in evaluation target extraction

features can provide best result in recall and F-measure in partial match.

5 Related Work

Some previous research extracted evaluative or polarity-bearing expressions from web documents with pre-defined linguistic patterns (Kobayashi et al., 2004; Kaji and Kitsuregawa, 2006). However, it is difficult to prepare a small number of fixed syntactic patterns to extract a wide range of evaluative expressions. Nakagawa et al. (2008) presented the study about extracting Japanese evaluative expressions from the web. Our task definition is based on their work. We applied these tasks to Chinese, made a Chinese corpus and presented our new features to improve the performance of evaluative information extraction.

In recent years, there have been several opinion-related evaluation workshops concerning Chinese opinion mining, such as Chinese Opinion Analysis Evaluation (COAE) (Zhao et al., 2008) and the Multilingual Opinion Analysis Task (MOAT) of NTCIR (Seki et al., 2010). Several subtasks are conducted in both COAE and MOAT, including the opinion-bearing sentence detection, opinion target extraction and polarity determination. The opinion target extraction task in COAE identifies the product features, which are defined as product components or attributes. Compared with COAE, the evaluation targets extracted by our system can cover a wider scope; they can be nouns, multi-word expressions or nouns modified by clauses. At the same time, we considered evaluation holders in this research. Since opinion expressers influence the credibility, identifying the evaluative holders is very important for analyzing the evaluations. MOAT also includes the opinion target and opinion holder extraction tasks. Compared with MOAT, we introduce evaluation types and extend the coverage of the opinion mining targets. Explicit and implicit opinions, and subjective and objective evaluations are considered in our research, while MOAT only considers the opinionated sentences, not including the general facts, such as positive or negative facts. Furthermore, COAE mainly deals with opinion analysis in reviews, and MOAT deals with the opinion analysis in news, which are written in a more formal writing styles. Since our system was trained with a corpus, which is written in more diverse writing styles and covers wide domains, we believe it is more robust to the web data of a great variety of topics and styles.

Conclusion

In this paper, we presented a Chinese evaluative information analysis system and proposed new simple yet effective features to improve its performance. Through a series of experiments, we demonstrated that our system can achieve reasonably good performance and that our new features provides substantial improvement in evaluative expression extraction and evaluation target extraction tasks.

References

Akamine, S., Kawahara, D., Kato, Y., Nakagawa, T., Leon-Suematsu, Y. I., Kawada, T., Inui, K., Kurohashi, S., and Kidawara, Y. (2010). Organizing information on the web to support user judgments on information credibility. In *Proceedings of the 4th International Universal Communication Symposium (IUCS2010)*, pages 122–129.

Breck, E., Choi, Y., and Cardie, C. (2007). Identifying expressions of opinion in context. In *Proceedings-IJCAI-2007*, pages 2683–2688.

Brown, P., Pietra, V. D., de Souza, P., Lai, J., and R.L.Mercer (1992). Class-based n-gram models of natural language. *Computational Linguistics*, 18:467–479.

Johansson, R. and Moschitti, A. (2010). Syntactic and semantic structure for opinion expression detection. In *In Proceedings of ACL-2010*, pages 67–76.

Kaji, N. and Kitsuregawa, M. (2006). Automatic construction of polarity-tagged corpus from html document. In *Proceedings-COLING/ACL-2006*, pages 452–459.

Kazama, J. and Torisawa, K. (2008). Inducing gazetteers for named entity recognition by large-scale clustering of dependency relations. In *Proceedings of ACL-2008*, pages 665–673.

Kobayashi, N., Inui, K., Matsumoto, Y., Tateishi, K., and Fukushima, T. (2004). Collecting evaluative expressions for opinion extraction. In *Proceedings-IJCNLP-2004*, pages 584–589.

Koo, T., Carreras, X., and Collins, M. (2008). Simple semi-supervised dependency parsing. In *Proceedings of ACL-2008*, pages 595–603.

Lafferty, J., McCallum, A., and Pereira, J. (2001). Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of ICML01*, pages 282–289.

Liu, B. (2010). *Sentiment Analysis and Subjectivity*. CRC Press, Taylor and Francis Group.

Miller, S., Guinness, J., and Zamanian, A. (2004). Name tagging with word clusters and discriminative training. In *Proceedings of HLT-2004*, pages 337–342.

Nakagawa, T., Inui, K., and Kurohashi, S. (2010). Dependency tree-based sentiment classification using CRFs with hidden variables. In *Proceedings-HLT-NAACL-2010*, pages 786–794.

Nakagawa, T., Kawada, T., Inui, K., and Kurohashi, S. (2008). Extracting subjective and objective evaluative expressions from the web. In *Proceedings of the 2nd International Symposium on Universal Communication*, pages 251–258.

Pang, B. and Lee, L. (2008). Opinion mining and sentiment analysis. *Foundations and Trends in Information Retrieval*, 2(1-2):1–135.

Pang, B., Lee, L., and Vaithyanathan, S. (2002). Thumbs up? sentiment classification using machine learning techniques. In *Proceedings-EMNLP-2002*, pages 79–86.

Seki, Y., Ku, L.-W., Sun, L., Chen, H.-H., and Kando, N. (2010). Overview of multilingual opinion analysis task at ntcir-8: A step toward cross lingual opinion analysis. In *Proceedings of NTCIR-8 Workshop Meeting-2010*, pages 209–220.

Wang, Y., Kazama, J., Tsuruoka, Y., Chen, W., Zhang, Y., and Torisawa, K. (2011). Improving chinese word segmentation and pos tagging with semi-supervised methods using large auto-analyzed data. In *Proceedings-IJCNLP-2011*, pages 309–317.

Zhao, J., Xu, H., Huang, X., Tan, S., Liu, K., and Zhang, Q. (2008). Overview of chinese opinion analysis evaluation. In *Proceedings of the First Chinese Opinion Analysis Evaluation*, pages 1–20.

Harnessing the CRF complexity with domain-specific constraints. The case of morphosyntactic tagging of a highly inflected language.

Jakub Waszczuk
Institute of Computer Science, Polish Academy of Sciences
Jakub.Waszczuk@ipipan.waw.pl

ABSTRACT
We describe a domain-specific method of adapting conditional random fields (CRFs) to morphosyntactic tagging of highly-inflectional languages. The solution involves extending CRFs with additional, position-wise restrictions on the output domain, which are used to impose consistency between the modeled label sequences and morphosyntactic analysis results both at the level of decoding and, more importantly, in parameters estimation process. We decompose the problem of morphosyntactic disambiguation into two consecutive stages of the context-sensitive morphosyntactic guessing and the disambiguation proper. The division helps in designing well-adjusted, CRF-based methods for both tasks, which in combination constitute Concraft, a highly accurate tagging system for the Polish language available under the 2-clause BSD license. Evaluation on the National Corpus of Polish shows that our solution significantly outperforms other state-of-the-art taggers for Polish – Pantera, WMBT and WCRFT – especially in terms of the accuracy measured with respect to unknown words.

KEYWORDS: morphosyntactic tagging, conditional random fields, Polish.

Proceedings of COLING 2012: Technical Papers, pages 2789–2804,
COLING 2012, Mumbai, December 2012.

1 Introduction

Morphosyntactic tagging is one of the basic problems in Natural Language Processing (NLP), which can be described as choosing most appropriate morphosyntactic descriptions for each word in the particular sentence. We ignore the problem of lemma-ambiguity in our work, since it doesn't occur often within the context of highly-inflectional languages like Polish or Czech (Smith et al., 2005), and concentrate on the problem of choosing correct values of grammatical attributes. Numerous methods have been proposed as a solution to the tagging problem, ranging from rule-based with hand-crafted rules, through transformation-based with rules automatically extracted from a dataset, to stochastic methods like hidden Markov models (Jurafsky and Martin, 2008) or linear-chain conditional random fields (Lafferty et al., 2001; Sutton and Mccallum, 2006). We focus on stochastic methods in this work.

Best solutions for the English language yield tagging accuracy exceeding 97%, but methods designed for one language do not necessarily solve problems inherent to another one. In case of highly-inflectional languages the first (and rather technical) obstacle is the complexity of the morphosyntactic tagset. Polish tags carry information about a number of grammatical attributes – part of speech (POS), case, number, gender etc. – with 13 grammatical categories in total. The Polish tagset used within the 1-million-word National Corpus of Polish (NCP) (Przepiórkowski et al., 2010) is a positional one: first position of the tag represents the grammatical class and subsequent positions contain category values. In the NCP tagset the first position determines the set of grammatical categories (some obligatory, some optional) used in conjunction with the class. For example, the **subst:sg:nom:m1** tag describes a singular (**sg**) human masculine (**m1**) noun (**subst** – substantive) in the nominative case (**nom**).

There are more than 1000 morphosyntactic tags which can be directly extracted from the NCP corpus. It makes the application of methods, which do not scale with the number of tags very well, hard in practice. One of such methods is the conditional random fields (CRFs) formalism, with tagging complexity which is quadratic in the number of tags[1] and with expensive global training process[2]. After the introduction of CRFs within the field of NLP (Lafferty et al., 2001) many methods have been developed in order to exploit their benefits (conditional, feature-rich modeling) while not being restrained by the expensive, global training process of CRFs at the same time. Collins (2002) has shown that by using discriminative methods hidden Markov models (HMMs) can be trained to achieve comparable performance. Toutanova et al. (2003) have proposed tagging with cyclic dependency network, an alternative to CRFs with similar advantages and, additionally, a much less expensive training procedure. In case of CRFs, Sokolovska et al. (2009) have shown that when the set of bigram features is small, the sparse forward-backward recursions method can be used to significantly speed-up the training process. Other optimization techniques, which make it possible to use CRFs to solve large-scale problems (with large output space, in particular), have been described in (Lavergne et al., 2010). However, all these alternatives either deprive us from the pure context of CRFs, or require some additional assumptions and complicate the implementation. We describe a simple extension of CRFs which makes them better suited to the morphosyntactic tagging and partially solves the problem of high computational cost of the training process. Moreover, it can be used together with more implementation-specific optimizations mentioned above.

[1] In case of the simple first-order model.

[2] Assuming that training is performed with respect to the standard log-likelihood function.

We show that by integrating morphosyntactic analysis results into the internals of CRFs it is easy to obtain a high-quality, efficient solution of the tagging problem, even for a language with rich morphosyntactic structure like Polish. Our method can be used within the context of any language for which a high-quality morphosyntactic analysis tool exists. The analyzer should conform to the principle that, for any given input word, the resulting set of potential morphosyntactic tags is either complete (all theoretically possible interpretations are included) or empty (the tool doesn't recognize the word). Partial results can harm the tagging performance because our method always chooses tags consistent with the analysis output. The current implementation requires that tags are represented in a positional form, but this requirement is of a technical nature and it could be easily relaxed. The only assumption which is actually made by the disambiguation method is that tags can be divided into relatively independent parts which are subsequently modeled in separate CRF layers. In practice, when the size of the tagset is small and only one disambiguation layer is sufficient, there are virtually no requirements about the form or structure of the tagset.

The rest of this document is organized as follows. First we describe the constrained version of CRFs (section 2), which provides a way to restrict sets of possible output labels on individual sentence positions. Restrictions are used to impose consistency between morphological analysis results and CRF output labels. Next we show how the introduced mathematical formalism can be used to perform a context-sensitive morphosyntactic guessing (section 3) and disambiguation (section 4) which together constitute Concraft[3], a state-of-the-art tagging system for the Polish language. Final evaluation on the National Corpus of Polish, carried out in accordance with the guidelines described in (Radziszewski and Acedański, 2012), is described in section 6.

2 Constrained Conditional Random Fields

We now recall the definition of conditional random fields and define their extension which we call constrained conditional random fields (CCRFs) throughout this paper. Similar modification of the CRF formalism can be found in (Smith et al., 2005), while comprehensive introduction to CRFs can be found in (Lafferty et al., 2001; Sutton and Mccallum, 2006). CCRFs is a simple but useful extension which employs constraints over individual labels in the output sequence. We represent each constraint as a set of labels, to which a label on the particular position in the sentence must belong. Those constraints are satisfied both during the inference and parameters estimation algorithms, thus improving their efficiency and allowing the model to be better adapted to a particular problem domain. Within the context of morphosyntactic tagging, constraints can be used to enforce consistency between results of morphosyntactic analysis and the tagging process itself.

For simplicity we consider only a first-order (with features defined over two subsequent labels at maximum) HMM-like (no bigram features) conditional random fields. Without the loss of generality we assume that (i) each word in the input sequence is represented by a descriptive set of observations, with each observation capturing some aspect of the word itself or its context, (ii) features can refer only to observations related to the current word. This design decision makes it easier to separate the notion of conditional knowledge, supplied in form of the input sequence, from the internal workings of the CRF implementation. Definitions and equations given below can be extended to describe models of higher order and models with more complex feature types.

[3]Available at `http://hackage.haskell.org/package/concraft` under the 2-clause BSD license. The acronym stands for *Constrained Conditional Random Fields Tagger*.

The formalism described in this section, as well as its generic extension to the second-order case, have been implemented as separate Haskell programming language modules[4] which can be used with other applications in mind (chunking or named entity recognition, for instance). The Haskell language combines advantages of high-level programming and type safety with excellent performance of generated programs[5], which makes it an ideal candidate for conceptually complex mathematical tasks. Both libraries have been used to perform tests described in this paper, as a statistical core of the Concraft tool.

2.1 Definition

Let O be a set of *observations*, Y a set of *labels* and $X = 2^O$. Let $\mathbf{x} = (x_1 \in X, \ldots, x_n \in X)$ be an input sequence of words, where each word is represented by a descriptive set of observation, and $\mathbf{y} = (y_1 \in Y, \ldots, y_n \in Y)$ an output sequence of labels. We also assume that $x_i = \emptyset$ and $y_i = \delta$ for $i < 1 \vee i > n$, where δ is a dummy label not belonging to the Y set. First-order linear-chain CRF defines a conditional probability of the sequence of labels $\mathbf{y}$ given the sentence $\mathbf{x}$ as a normalized product of position-wise model potentials:

$$p_\theta(\mathbf{y}|\mathbf{x}) = Z_\theta(\mathbf{x})^{-1} \prod_{i=1}^{n} \phi_\theta(x_i, y_i, y_{i-1}) \tag{1}$$

where $\theta = \{\theta_k\}_{k=1}^K$ is a set of *parameters*, with the k-th parameter representing the contribution of the k-th *feature* to the overall probability, $\phi_\theta(x_i, y_i, y_{i-1})$ denotes a *potential* on the i-th position of the sentence and $Z_\theta(\mathbf{x})$ is a normalization factor. Potential ϕ is defined as an exponential of the sum of parameters related to features present within the context of (x_i, y_i, y_{i-1}):

$$\phi_\theta(x_i, y_i, y_{i-1}) = \exp\left(\sum_{k=1}^{K} \theta_k f_k(x_i, y_i, y_{i-1})\right) \tag{2}$$

The implemented CCRF handles two kinds of features. A *unigram feature* $(u \in Y, o \in O)$ takes responsibility for modeling dependency between a word described by the observation o and the label u assigned to that word. A *transition feature* $(v \in Y \cup \{\delta\}, u \in Y)$, on the other hand, serves to model dependency between adjacent labels v (which is equal to δ when considering the first position of the sentence) and u. Therefore, the form of the feature function f_k depends on what kind of feature, a unigram one or a transition one, it represents:

$$f_k(x_i, y_i, y_{i-1}) = \begin{cases} \mathbf{1}(y_i = u, o \in x_i) & \text{if } k \text{ identifies unigram feature } (u, o) \\ \mathbf{1}(y_i = u, y_{i-1} = v) & \text{if } k \text{ identifies transition feature } (v, u) \end{cases} \tag{3}$$

where $\mathbf{1}(cond.)$ is equal to 1 when the given condition holds and to 0 otherwise.

Finally, the normalization factor Z is defined as

$$Z_\theta(\mathbf{x}) = \sum_{\mathbf{y} \in Y^n} \prod_{i=1}^{n} \phi_\theta(x_i, y_i, y_{i-1}). \tag{4}$$

[4]Available at `http://hackage.haskell.org/package/crf-chain1-constrained` and `http://hackage.haskell.org/package/crf-chain2-generic` under the 2-clause BSD license.

[5]We relate to the efficiency of optimized programs compiled with a Glasgow Haskell Compiler.

2.2 Constraints

Let $\mathbf{r} = (r_1 \subseteq Y, \ldots, r_n \subseteq Y)$ be a sequence of non-empty constraints over individual labels within the given sentence $\mathbf{x}$. For convenience we assume that $r_i = \{\delta\}$ for $i < 1 \vee i > n$. Definition 1 can be easily modified to take those constraints into account. In particular, the model assigns probability 0 to all label sequences which are inconsistent with the constraints imposed by $\mathbf{r}$:

$$p_\theta(\mathbf{y}|\mathbf{x},\mathbf{r}) = \begin{cases} Z_\theta(\mathbf{x},\mathbf{r})^{-1} \prod_{i=1}^{n} \phi_\theta(x_i, y_i, y_{i-1}) & \text{if } \mathbf{y} \in \prod_{i=1}^{n} r_i \\ 0 & \text{otherwise} \end{cases} \tag{5}$$

The normalization factor within the constrained context is defined as

$$Z_\theta(\mathbf{x},\mathbf{r}) = \sum_{\mathbf{y} \in \prod_{i=1}^{n} r_i} \prod_{i=1}^{n} \phi_\theta(x_i, y_i, y_{i-1}). \tag{6}$$

2.3 Inference

We describe here two methods of inference using CCRFs: finding the most probable label sequence given the sentence and assigning marginal probabilities to individual labels from restricted sets. The first one is used within the context of morphosyntactic disambiguation (section 4), second one within the context of guessing (section 3). Throughout this section we assume that input sentence $\mathbf{x}$, constraints $\mathbf{r}$ and parameters θ are fixed.

2.3.1 Decoding

Given a sentence $\mathbf{x}$ and constraints $\mathbf{r}$, the CCRF model with parameters θ can be used to find the most probable label sequence $\mathbf{y}^*$. Since the normalization factor $Z_\theta(\mathbf{x},\mathbf{r})$ doesn't depend on the label sequence $\mathbf{y}$, the task is equivalent to finding label sequence which maximizes the potential.

$$\mathbf{y}^* = \operatorname*{arg\,max}_{\mathbf{y} \in \prod_{i=1}^{n} r_i} \left(\prod_{i=1}^{n} \phi_\theta(x_i, y_i, y_{i-1}) \right) \tag{7}$$

The task of finding $\mathbf{y}^*$ is often called *decoding*. It can be solved efficiently using the *max-product* algorithm (Wainwright and Jordan, 2008), which can be also employed within the context of the constrained CRF model. Let $r[i \to u]$ denote a sequence of constraints resulting from substituting the i-th element in the sequence r with the singleton $\{u\}$. For $i \in [0..n+1]$ and $u \in r_i$ we denote by $\omega_i(u)$ the partial, maximum cumulative potential acquired between the beginning of the sentence and the i-th position assuming, that label at position i takes the value of $u \in r_i$.

$$\omega_i(u) = \max_{\mathbf{y} \in \prod_{j=1}^{i} r[i \to u]_j} \left(\prod_{j=1}^{i} \phi_\theta(x_j, y_j, y_{j-1}) \right) \tag{8}$$

It can be also defined in a recursive way:

$$\omega_i(u) = \begin{cases} \max_{v \in r_{i-1}} \omega_{i-1}(v) \cdot \phi_\theta(x_i, u, v) & \text{if } i > 0 \\ 1 & \text{otherwise} \end{cases} \tag{9}$$

The recursive definition, together with dynamic programming or memoization techniques, can be used to find the most probable label sequence $\mathbf{y}^*$ (the sequence corresponding to the value $\omega_{n+1}(\delta)$) in an efficient way.

2.3.2 Marginal Probabilities

The CRF models conditional probability of the entire label sequence, but conditional *marginal* probabilities also can be computed in an efficient way. Marginal probabilities can be used in the output of the CRF instead of (or together with) the decoded sequence $\mathbf{y}^*$. In this work we use marginal probabilities to perform context-sensitive morphosyntactic guessing, which is described in section 3.

Let $i \in [0..n+1]$ and $u \in r_i$. We define a forward potential sum as

$$\alpha_i(u) = \sum_{\mathbf{y} \in \prod_{j=1}^{i} r[i \to u]_j} \left(\prod_{j=1}^{i} \phi_\theta(x_j, y_j, y_{j-1}). \right) \tag{10}$$

The notion of a forward potential sum is similar to that defined by the formula (8), but instead of maximizing the cumulative potential between the first and the i-th position we are summing up all cumulative potentials between positions 1 and i of the input sentence. Backward potential sum $\beta_i(u)$ is a reverse concept corresponding to the sum of cumulative potentials between positions $i+1$ and n given that the value of the i-th label is equal to u.

It should be noted that both forward and backward potential sums can be used to compute the value of the normalization factor, since $Z_\theta(\mathbf{x}, \mathbf{r}) = \alpha_{n+1}(\delta) = \beta_0(\delta)$ for the input sentence $\mathbf{x}$ and constraints $\mathbf{r}$. Furthermore, it can be shown that all three concepts provide the following formula to determine the marginal probability of label u at position i:

$$p_\theta(y_i = u | \mathbf{x}, \mathbf{r}) = \alpha_i(u) \cdot \beta_i(u) \;/\; Z_\theta(\mathbf{x}, \mathbf{r}) \tag{11}$$

Finally, the efficient algorithm for forward potential sum computation is known as a *sum-product* algorithm (Wainwright and Jordan, 2008) and it is based on the following recursive definition of the forward potential sum:

$$\alpha_i(u) = \begin{cases} \sum_{v \in r_{i-1}} \alpha_{i-1}(v) \cdot \phi_\theta(x_i, u, v) & \text{if } i > 0 \\ 1 & \text{otherwise} \end{cases} \tag{12}$$

which has an analogous structure to the definition presented in equation (9). Backward potential sum β can be recursively defined in a similar way.

2.4 Parameters estimation

We note only that we apply a Stochastic Gradient Descent method in order to find parameter values which maximize log-likelihood of a training dataset with a Gaussian prior over the parameter values.

3 Morphosyntactic Guessing

In this section we describe an adaptation and an application of the formalism described in section 2 to the morphosyntactic guessing. A dictionary of known forms can be used in order to find all possible interpretations – tags and base forms – of the given word. This language processing phase is called *morphosyntactic analysis*. Unfortunately, due to incompleteness of data resources or spelling errors some forms may not be recognized. Morphosyntactic tagging of such words, called *unknown* throughout this paper, is severely hampered. The average

number of morphosyntactic tags assigned by the morphosyntactic analysis tool to known words, computed on the basis of the NCP million subcorpus, is approximately equal to 4. In case of unknown words we have to consider every possible morphosyntactic tag present in the tagset. This difficulty is reflected in the evaluation results of taggers designed for Polish (see section 6). Results of the morphosyntactic analysis are especially helpful in selecting correct values of lexeme-specific attributes like gender, since it is often hard to determine their values even on the basis of the occurrence context.

By morphosyntactic guessing we mean a method of determining sets of possible morphosyntactic tags for unknown words in the sentence. Thus, it can be thought of as a complement problem to the morphosyntactic analysis. Conceptually, the division of the tagging process into three separate steps of analysis, guessing and disambiguation, is a superficial one. It can be also pointed out that context-sensitive guessing is – from a theoretical, not practical, point of view – a generalization of the disambiguation process. Disambiguator is just a guesser which always proposes one possible tag in the resulting sequence. Yet, it is easier to reason about those three processing stages and to develop efficient tools designed to solve them separately. For example, if we ignore the problem of unknown words completely, we will be constrained to use methods which manage to handle adjacent words with more than 1000 possible tags at the stage of disambiguation. In particular, the method described in section 4 would not be able to handle such cases in an efficient way.

3.1 Model Adaptation

To use the constrained model described in section 2, every input word has to be linked with a list of possible morphosyntactic tags. Each tag is treated as an atomic entity, that is two tags are considered to be different if they differ on at least one grammatical attribute value. Since the method doesn't adopt any assumptions about the structure of morphosyntactic tags, it is very flexible and can be applied to a broad class of natural languages. In the case of known words the choice of possible labels is simple – morphosyntactic analysis is a source of label restrictions. For each unknown word we should consider all morphosyntactic tags from the tagset as potential interpretations. We use a simple heuristic instead, described in (Radziszewski, 2013), where a special set U – a set of all labels assigned to unknown words in the training corpus – is prepared. Since the U set can be in practice much bigger ($\approx$ 300 tags in NCP experiments) than the average set proposed by the analyzer and it is often the case that two adjacent words are unknown, we use the first-order model for the task of morphosyntactic guessing.

While there are many possible ways of using the CCRF to perform actual guessing, we have tested only the simplest one so far. For each unknown word in the input sequence, with the set of restrictions $\mathbf{r}$ fixed to U, we take k most probable labels according to marginal probabilities determined by the CCRF (see 2.3.2) for some arbitrarily chosen value of k.

To speed-up computations, especially within the context of known words, we implement sparse forward-backward recursions (Sokolovska et al., 2009; Lavergne et al., 2010). To make this optimization effective, a set of model features is constructed with respect to words and chosen labels from the training data T, but not with respect to restrictions in T. It is important to note that we can safely apply this optimization within the context of the sum-product algorithm (used during the training phase), but not with the max-product algorithm (used in decoding). Fortunately, to perform guessing we do not have to use max-product algorithm, since only the sum-product is needed to find marginal probabilities of individual labels.

3.2 Observations

A common solution to the morphosyntactic guessing problem is to propose the set of possible tags on the basis of orthographic features acquired for the particular word (e.g. prefixes and suffixes). By using the CRF sequential model we extend this method to take tags assigned on adjacent positions into account. The advantage of such a solution is that the guesser is able to reject morphosyntactic interpretations on the basis of two factors simultaneously: the context and orthographic features of the word.

Our observation schema contains the following set of observations, which are subsequently used to construct the set of unigram features:

- Lowercase prefixes and suffixes of lengths 1 and 2,
- A Boolean value indicating if the word is known,
- Packed shape of the word, and information whether the word is positioned at the beginning of the sentence, combined into one observation.

where each observation is related to the current word, the shape of a word is a string with all lowercased characters replaced by 'l', uppercased characters replaced by 'u', digits replaced by 'd' and any other character replaced by 'x' („Coling-2012" → „ullllllxdddd"), while a packed shape is a shape with all duplicate code characters removed („ullllxdddd" → „ulxd").

word	observations	choice	interpretations
Szef	{ 1.S, 2.Sz, 3.f, 4.ef , 5.True, 6.ul-True }	subst:sg:nom:m1	{ subst:sg:nom:m1 }
administracji	{ 1.a, 2.ad, 3.i, 4.ji , 5.True, 6.l-False }	subst:sg:gen:f	{ subst:sg:gen:f , subst:sg:dat:f , subst:sg:loc:f , subst:pl:gen:f }
Wołodymyr	{ 1.W, 2.Wo, 3.r, 4.yr , 5.False, 6.ul-False }	subst:sg:nom:m1	*U*
Łatwyn	{ 1.Ł, 2.Ła, 3.n, 4.yn , 5.False, 6.ul-False }	subst:sg:nom:m1	*U*

Table 1: Morphosyntactically analyzed sentence *Szef administracji Wołodymyr Łatwyn – Chief administrator Wołodymyr Łatwyn* prepared for the CCRF guessing. The observations column includes observations extracted for individual words according to the guessing observation schema. The choice indicates the morphosyntactic tag correct within the context. In the last column results of morphosyntactic analysis are presented. In case of unknown words the *U* set designates the set of potential interpretations.

Prefix and suffix observations are the basic type of information needed to determine the set of possible morphosyntactic interpretations of the given word. By using only short affixes we can be sure to get highly accurate estimations of corresponding parameters. The model learns dependencies between prefixes/suffixes and morphosyntactic tags mostly on words from the corpus which are known, since they are far more frequent in the training material than the unknown words. After that it is able to use this knowledge to guess interpretations of unknown words. Packed shape might be helpful to distinguish some frequent classes of unknown words –

named entities and numbers, in particular. Information about the fact that the word is known or not is also provided. It allows the guesser to exhibit a slightly different behavior on known and unknown words. We do not employ the standard observation – full orthographic form – on purpose. It would probably make the model more accurate with respect to known words, but our main goal is to obtain accurate estimations of prefix- and suffix-specific parameters.

Table 1 presents a sample sentence prepared for the subsequent processing with the CCRF guessing module. For each word the set of observations is determined on the basis of the observation schema described above. Observations are extended with additional prefixes identifying the kind of the particular observation. Every observation is a string, but from a technical point of view observations can be represented by more complex structures as long as their data type is totally ordered, which is the only assumption about the form of observations made on the level of the CCRFs implementation. The presented structure can be used to perform subsequent CCRF processing in the following way:

- The 'observations' column is translated to the input sequence **x**,
- The 'interpretations' column is converted to the sequence of CCRF constraints **r**,
- The contents of the 'choice' column (if any) can be used as the output sequence **y**.

All three columns can be used as the input to the CCRF training process. To perform the actual guessing only the observations and interpretations columns need to be known, of course.

word	guessing results
Szef	{ subst:sg:nom:m1 }
administracji	{ subst:sg:gen:f , subst:sg:dat:f , subst:sg:loc:f , subst:pl:gen:f }
Wołodymyr	{ subst:sg:nom:m2, subst:sg:nom:n, subst:sg:gen:m3 , subst:pl:nom:m1, subst:sg:nom:m3, subst:sg:nom:m1 , adj:sg:gen:m3:pos, subst:sg:loc:f, qub, brev:npun }
Łatwyn	{ subst:sg:gen:m3, subst:sg:nom:m2, subst:sg:nom:n , subst:sg:gen:m1, subst:sg:gen:f, subst:sg:nom:m3, subst:sg:acc:n , subst:sg:nom:m1, subst:pl:gen:m1, subst:sg:nom:f }

Table 2: Results of guessing interpretations of unknown words.

Table 2 shows the results of the guessing process performed on the phrase presented in table 1 with $k = 10$. We do not show the 'choice' column, since it is not used within the process, and the 'observations' column, which has been relevant only for input. Potential interpretations assigned to known words are not influenced by the guesser. The guessing has been performed using the model trained on the part which has not contained the phrase used as an example.

Since the guesser always proposes multiple morphosyntactic tags[6] for individual unknown words in the input, there must be also a lot of incorrect (w.r.t. the context) or even theoretically impossible (w.r.t. the word in question) interpretations in the output. The 'subst:sg:loc:f' tag is clearly not a possible interpretation of the masculine 'Wołodymyr' lexeme, but this fact is hard to determine without any reference to the word in the external dictionary. Therefore, to estimate the accuracy of the guesser we should only take the presence (or absence) of the correct tag

[6] Assuming that $k > 1$ and $|U| > 1$.

in the output set into account. As long as the correct tag is included in the output, it is of no particular concern to us what other output interpretations are. Looking at the 'guessing results' column in table 2 we can see that the guesser successfully determined that noun (substantive) is the most probable class for both unknown words, 'Wołodymyr' and 'Łatwyn', in the example. It has been also able to determine nominative case with a high confidence. The gender attribute is the biggest source of ambiguity in the output results – since the first-order model is used to perform the guessing, the method is not able to agree gender values of unknown words with the gender of the first word in the sentence, 'Szef', which is known to represent a human masculine (m1). Finally and most importantly, the correct full morphosyntactic tag (subst:sg:nom:m1) is included in the output for both unknown words.

4 Morphosyntactic Disambiguation

The last and the most important step of morphosyntactic tagging is a disambiguation phase. It involves choosing for each word, from the set of possible tags, the most appropriate one in the context of the particular sentence. It can be resolved without the prior morphosyntactic analysis or guessing, but these steps – as long as they demonstrate low false negative error rate – should only accelerate disambiguation (thanks to the reduced search space) and improve its accuracy.

Second-order stochastic methods are often chosen as a solution to the tagging problem, see (Collins, 2002; Smith et al., 2005), and we use the second-order flavor of constrained CRFs for disambiguation as well. The complexity of the task is no longer problematic, because sets of possible interpretations are strongly limited for individual words. If a word is known, the set of potential labels is provided by the morphosyntactic analyzer. Otherwise, potential labels are restricted to the set of the k most probable labels determined at the stage of morphosyntactic guessing described in the previous section, where k is the parameter of the guesser.

4.1 Model Adaptation

In case of the disambiguation problem we are more concerned about the performance of the solution in terms of quality than in terms of speed. We rely on the analysis and guessing tools in the second aspect and, in particular, on their ability to reduce the search space for the disambiguator. Therefore, we have chosen to implement a second-order variant of CCRF which incorporates a dense feature set (constraints from the training set are taken into account during the construction of the feature set) with no sparse forward-backward recursions optimization.

The second-order characteristic means that relations between three subsequent labels are modeled directly within the CRF. The sparse forward-backward recursions optimization has not been implemented in this case because it doesn't bring any improvements when the set of model features is dense. An important difficulty stems from the fact that we use the second-order model within the context of a language with more than 1000 potential morphosyntactic tags. It is easy to deduce that there are more than 10^9 possible transition features and, even with additional constraints taken into consideration, there are more than $1.5 * 10^7$ transition features which can be extracted from the NCP corpus directly. It is not only hard to store the model with so many features, but it can also cause problems related to data sparseness and make the model prone to overfitting.

To alleviate this problem we change our treatment of morphosyntactic tags as atomic labels and regard them as complex structures. We also extend our second-order CCRF to model multiple morphosyntactic layers. Tags are divided into separate label layers according to a configuration

file. The file specifies the number of layers and the destination layers (zero, one or more) for each grammatical attribute (POS or grammatical category). Layers are modeled separately (i.e., features cover label values from exactly one layer), but not independently. Finally, labels in individual layers are treated as atomic entities.

This method is also rather flexible and could be used with respect to a language with other tagset structure than the positional one. The only important assumption which is really adopted here is that it is possible to divide a tag into parts which are relatively independent. For instance, in case of the partitioning which assigns case values to the first layer and number values to the second layer, implicit assumption is being made that – given observations – there is no direct dependency between those values apart from the dependency enforced by constraints. If we ignore constraint-related dependencies we can write it in a more formalized way: the case value of the i-th word is conditionally independent of its number value (and, in fact, all other number values in the sentence) given case values of words in positions $i-2$, $i-1$, $i+1$ and $i+2$ and observation values related to the i-th word.

4.2 Observations

In the case of disambiguation we use a richer set of observation types than within the context of guessing, yet it is still rather a minimalistic one. Let w_i denote the word at the i-th position in the input sentence. The following set of observations is used with respect to the i-th position:

- Lowercase orthographic forms of words w_{i-1}, w_i and w_{i+1},
- If the word w_i is unknown:
  - Lowercase prefixes and suffixes of length 1, 2 and 3 of w_i,
  - Packed shape of w_i and information, whether w_i is positioned at the beginning of the sentence, combined into one observation.

The shape observation type has been mentioned earlier within the context of the guesser's observation schema. We haven't recorded any improvements in the tagging quality when using prefix, suffix or shape features as observations for known words. Therefore, we do not use them in our final model.

4.3 Features

Let L be a number of layers and $y(l)$ denote a part of the morphosyntactic tag y assigned to the l-th layer for $l \in \{1,\ldots,L\}$. The layered model handles unigram (l,u,o) and transition (l,w,v,u) features in a similar way to that specified in equation 3. However, in the layered case features are enriched with value l identifying the layer to which the particular feature should be related. Moreover, the form of the feature function f_k changes due to the second-order character of the model.

$$f_k(x_i,y_i,y_{i-1},y_{i-2}) = \begin{cases} \mathbf{1}(y_i(l)=u, o \in x_i) & k \text{ identifies } (l,u,o) \\ \mathbf{1}(y_i(l)=u, y_{i-1}(l)=v, y_{i-2}(l)=w) & k \text{ identifies } (l,w,v,u) \end{cases} \quad (13)$$

While the implementation is general and follows the specification described above, all tests within this work (most importantly, the evaluation) were performed using the configuration inspired by (Acedański, 2010). Tags are divided into two layers, with POS, case and person in the first layer and all the other grammatical categories in the second layer.

5 Related Work

We compare our method with solutions for morphosyntactic tagging of highly inflectional languages, the Polish language in particular. However, the method described in (Smith et al., 2005) – which bears many similarities with our solution – has been successfully used in tagging not only inflectional but also concatenative and templatic languages and we believe that our system could be also adapted to handle them.

Radziszewski (2013) describes WCRFT, an implementation of a tiered tagging method in which a cascade of independently trained linear-chain first-order CRF models is used to disambiguate input in a multi-pass manner: POS values are resolved at first, then case values, gender values and so on. Our system relies on a more general method of tag partitioning and performs disambiguation on all layers simultaneously, so that the choice made in higher layers can propagate down into lower layers, which is not possible in case of a cascade. Besides, our system uses second-order model for disambiguation which allows to capture dependencies between three subsequent tags in an idiomatic way. WCRFT uses the simple heuristic mentioned in section 3.1 to ascertain potential interpretations of unknown words and, optionally, a morphosyntactic guesser developed within a rule-based TaKIPI tagger (Piasecki, 2007; Piasecki and Radziszewski, 2007). It is an *a tergo* guesser, which uses pseudo-suffixes to infer potential morphosyntactic interpretations of unknown words. While it also proposes base forms in its output, it doesn't make use of context-specific information and often commits false negative errors (does not include correct morphosyntactic tags in its output). Since the disambiguation module in not designed so as to correct errors performed on the level of guessing, those false negative errors propagate to subsequent processing phases.

Pantera (Acedański, 2010) is an implementation of the Brill method (Brill, 1992) adapted for the specifics of highly inflectional languages. Pantera also adopts multi-pass method for morphosyntactic tagging without the possibility of downward propagation of disambiguation choices. Tag partitioning in Pantera is a configurable process and the evaluation of Pantera involved the same tag partitioning as the one used for the evaluation of our system: POS, case and person attributes are placed in the first layer, while the rest of grammatical categories is stored in the second layer (Acedański, 2010).

A variant of CRF, where each label must belong to a restricted set of interpretations recognized by a morphosyntactic analysis tool, has been previously used in tagging Korean, Arabic and Czech language (Smith et al., 2005). While the Czech model exhibits similarities to the model described in this paper, there are also important differences between them. In (Smith et al., 2005) there is no special treatment of unknown words. We, in comparison, introduce the context-sensitive morphosyntactic guessing method designed specifically for them. All grammatical attributes are taken into account in our model for Polish, while the model for Czech includes only the four main grammatical attributes – POS, case, number and gender – thus rendering it incapable of disambiguating between values of the remaining attributes. In order to make the training process feasible, Smith et al. (2005) employ factored training, dividing features into five separate classes and estimating parameters within each class independently. We do not adopt such parameter-level independence assumptions and perform global training using the Stochastic Gradient Descent method, yet we achieve an empirical performance which is several times better in terms of the training time. According to Smith et al. (2005), training the POS-specific part of the model for Czech on 768K words from the Czech PDT corpus takes up to two weeks on the 2GHz Pentium machine. Our system needs less than 8 hours to learn

parameters for guessing (3 hours) and disambiguation (5 hours) on 1M words from the NCP corpus using one core of the 2.40GHz Xeon E5620 machine.

6 Evaluation

Evaluation of Concraft has been performed on the one-million-word, balanced NCP subcorpus (Przepiórkowski et al., 2010). In order to be able to compare the tool with other taggers available for Polish, we have followed guidelines provided by Radziszewski and Acedański (2012). It describes a fair method of taggers evaluation, which involves obligatory resegmentation (sentence splitting and tokenization) and reanalysis of the evaluation corpus part. The advantage of this evaluation method is that it measures tagging quality in an environment as close as possible to the real usage case, given the available resources. No artificial assumptions of perfect segmentation or perfect morphosyntactic analysis are made, which was often the case with earlier evaluations of tagging systems (Acedański, 2010; Radziszewski and Śniatowski, 2011) which reported much higher results than could be expected in the real-world usage.

Our tool does not include any sentence splitting, tokenization or morphosyntactic analysis modules yet. Results presented below have been acquired using the preprocessing functionality provided by the MACA tool (Radziszewski and Śniatowski, 2011). For each cross-validation fold, the guesser is trained on the reanalyzed training part. The resulting model is used to guess potential interpretations of unknown words both in the training and the evaluation part, with k (number of retained labels for each unknown word) set to 10. Next, the disambiguation tool is trained on the training part with tags already guessed. Finally, the model obtained is used to perform disambiguation on the evaluation part (processed beforehand by the guesser).

Tagger	Acc_{lower}	Acc_{upper}	Acc^{K}_{lower}	Acc^{U}_{lower}
Pantera	88.99%	89.28%	91.27%	14.74%
WMBT	89.71%	90.04%	91.20%	41.45%
WCRFT	90.34%	90.67%	91.89%	40.13%
Concraft	91.12%	91.44%	92.10%	59.19%

Table 1: Average accuracy measures obtained by individual taggers on the NCP corpus.

Table 1 presents cross-validation results of our system in comparison to the state-of-the-art taggers for Polish: Pantera (Acedański, 2010), WMBT (Radziszewski and Śniatowski, 2011) and WCRFT (Radziszewski, 2013). Pantera and WCRFT have been described in the 5 section. WMBT (Radziszewski and Śniatowski, 2011) and WCRFT (Radziszewski, 2013) both represent a class of tiered tagging methods (Tufiş, 1999), using a cascade of models to perform disambiguation. WMBT uses a memory-based learning method to model individual tiers, while WCRFT uses CRFs for this task. All tools have been evaluated on the same extract of the NCP corpus, and with respect to exactly the same corpus partitioning. Numbers included in the first three rows have been reported in (Radziszewski and Acedański, 2012) and (Radziszewski, 2013).

We report separate evaluation statistics corresponding to different assumptions about the system's behavior in the situation of tokenization-level errors. Accuracy lower bound Acc_{lower} corresponds to the situation when each tokenization error is penalized and treated as a tagging error. Conversely, accuracy upper bound Acc_{upper} doesn't take tokenization errors into account – the measure is computed as if for each token from the reference corpus associated with the incorrectly tokenized part the tagger made the correct decision. Therefore, assuming perfect tokenization, accuracy of the tagger would be somewhere between Acc_{lower} and Acc_{upper}. Next

two statistics, Acc^{K}_{lower} and Acc^{U}_{lower}, show lower bound accuracy with respect to known and unknown words, respectively. Sentence splitting errors are not taken into account. Detailed description of individual statistics can be found in (Radziszewski and Acedański, 2012).

The results show that our system significantly outperforms other taggers, especially within the context of unknown words. They suggest that feature-rich stochastic methods perform better than transformation-based methods (represented by Pantera in the comparison). We suspect that the main reason of the gap between the WCRFT and Concraft performance is the difference in the choice of guessing methods. Our context-sensitive guessing method has been specifically designed as a preliminary filter, which reduces the number of possible interpretations of unknown words. The TaKIPI guesser used within the WCRFT tool during the evaluation yields too many false negatives, which cannot be corrected on the level of disambiguation.

Conclusion and perspectives

We have shown that a domain-specific modification of the CRFs formalism solves most of the complexity-related issues inherent to CRFs with respect to the morphosyntactic tagging of highly-inflectional languages. Not only is the modification easy to comprehend and implement, but it can be also used in combination with other optimizations designed for CRFs in general. We regard this as a confirmation that adapting the model to a particular problem domain should be the foremost optimization considered when designing a solution of a high quality.

The comparison of Concraft and WCRFT doesn't clearly show which tagging system would perform better if we used both of them in combination with the same guesser and this question should be further investigated. In particular, we plan to evaluate the guessing and the disambiguation components of the Concraft system separately. The current state of the disambiguation module still leaves some room for improvement on the level of tag partitioning and observation schema configuration. The CRF implementation can be further generalized by expanding the set of features with bigrams, trigrams or features capturing dependencies between individual tag layers (between parts of the same morphosyntactic tag for instance).

The guessing model can be used to perform guessing in a few reasonable ways. The first one, used within this work, is to output the k most probable labels from the U set for each unknown word in the sentence. Another option is to choose all labels with marginal probability higher than some arbitrarily chosen threshold. This solution has the advantage of making the size of the label set depend on probabilities of individual labels, so that very improbable labels do not get into the result even if they are in the set of the k most probable ones. Advantages of both solutions could be aggregated in a solution which uses both the marginal probability threshold and limits the size of the output set. The same techniques can be used to trim down not only label sets for unknown words, but also those which are assigned to known words. Advantage given by this solution would be a further speed up of computations on the level of the disambiguation phase, but it could potentially harm the tagging accuracy.

Finally, it would be worthwhile to confirm the usability of the tagger by testing its performance on another Slavic language and, afterwards, on a typologically different language (e.g. English, German or French) for which high-quality morphosyntactic analysis tools exist.

Acknowledgments

The release of the tagger described here is supported by the CESAR project (European project CIP ICT-PSP 271022, part of META-NET).

References

Acedański, S. (2010). A Morphosyntactic Brill Tagger for Inflectional Languages. In Loftsson, H., Rögnvaldsson, E., and Helgadóttir, S., editors, *Advances in Natural Language Processing*, volume 6233 of *Lecture Notes in Computer Science*, pages 3–14. Springer.

Brill, E. (1992). A simple rule-based part of speech tagger. In *Proceedings of the third conference on Applied natural language processing*, ANLC '92, pages 152–155, Stroudsburg, PA, USA. Association for Computational Linguistics.

Collins, M. (2002). Discriminative training methods for hidden Markov models: theory and experiments with perceptron algorithms. In *Proceedings of the ACL-02 conference on Empirical methods in natural language processing - Volume 10*, EMNLP '02, pages 1–8, Stroudsburg, PA, USA. Association for Computational Linguistics.

Jurafsky, D. and Martin, J. H. (2008). *Speech and Language Processing (2nd Edition) (Prentice Hall Series in Artificial Intelligence)*. Prentice Hall, 2 edition.

Lafferty, J. D., McCallum, A., and Pereira, F. C. N. (2001). Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data. In *Proceedings of the Eighteenth International Conference on Machine Learning*, ICML '01, pages 282–289, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Lavergne, T., Cappé, O., and Yvon, F. (2010). Practical very large scale CRFs. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, ACL '10, pages 504–513, Stroudsburg, PA, USA. Association for Computational Linguistics.

Piasecki, M. (2007). Polish Tagger TaKIPI: Rule Based Construction and Optimisation. *Task Quarterly*, 11(1–2):151–167.

Piasecki, M. and Radziszewski, A. (2007). Polish Morphological Guesser Based on a Statistical A Tergo Index. In *Proceedings of the International Multiconference on Computer Science and Information Technology — 2nd International Symposium Advances in Artificial Intelligence and Applications (AAIA'07)*, pages 247–256.

Przepiórkowski, A., Górski, R. L., Łaziński, M., and Pęzik, P. (2010). Recent Developments in the National Corpus of Polish. In Calzolari, N., Choukri, K., Maegaard, B., Mariani, J., Odijk, J., Piperidis, S., Rosner, M., and Tapias, D., editors, *Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC'10)*, Valletta, Malta. European Language Resources Association (ELRA).

Radziszewski, A. (2013). A tiered CRF tagger for Polish. In R. Bembenik, Ł. Skonieczny, H. R. M. K. M. N., editor, *Intelligent Tools for Building a Scientific Information Platform: Advanced Architectures and Solutions*. Springer Verlag.

Radziszewski, A. and Acedański, S. (2012). Taggers gonna tag: an argument against evaluating disambiguation capacities of morphosyntactic taggers. In *Proceedings of TSD 2012*, LNCS. Springer-Verlag.

Radziszewski, A. and Śniatowski, T. (2011). Maca — a configurable tool to integrate Polish morphological data. In *Proceedings of the Second International Workshop on Free/Open-Source Rule-Based Machine Translation*.

Radziszewski, A. and Śniatowski, T. (2011). A Memory-Based Tagger for Polish. In *Proceedings of the LTC 2011*. Tagger available at http://nlp.pwr.wroc.pl/redmine/projects/wmbt/wiki/.

Smith, N. A., Smith, D. A., and Tromble, R. W. (2005). Context-based morphological disambiguation with random fields. In *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, HLT '05, pages 475–482, Stroudsburg, PA, USA. Association for Computational Linguistics.

Sokolovska, N., Lavergne, T., Cappé, O., and Yvon, F. (2009). Efficient Learning of Sparse Conditional Random Fields for Supervised Sequence Labelling. *CoRR*, abs/0909.1308.

Sutton, C. and Mccallum, A. (2006). *Introduction to Conditional Random Fields for Relational Learning*. MIT Press.

Toutanova, K., Klein, D., Manning, C. D., and Singer, Y. (2003). Feature-rich part-of-speech tagging with a cyclic dependency network. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology - Volume 1*, NAACL '03, pages 173–180, Stroudsburg, PA, USA. Association for Computational Linguistics.

Tufiş, D. (1999). Tiered Tagging and Combined Language Models Classifiers. pages 28–33.

Wainwright, M. J. and Jordan, M. I. (2008). *Graphical Models, Exponential Families, and Variational Inference*. Now Publishers Inc., Hanover, MA, USA.

A Latent Discriminative Model for Compositional Entailment Relation Recognition Using Natural Logic

Yotaro Watanabe[1] *Junta Mizuno*[1] *Eric Nichols*[1]
Naoaki Okazaki[1,2] *Kentaro Inui*[1]
(1) Graduate School of Information Sciences, Tohoku University
(2) Precursory Research for Embryonic Science and Technology, Japan Science and Technology Agency
{yotaro-w,junta-m,eric,okazaki,inui}@ecei.tohoku.ac.jp

ABSTRACT
Recognizing semantic relations between sentences, such as entailment and contradiction, is a challenging task that requires detailed analysis of the interaction between diverse linguistic phenomena. In this paper, we propose a latent discriminative model that unifies a statistical framework and a theory of Natural Logic to capture complex interactions between linguistic phenomena. The proposed approach jointly models alignments, their local semantic relations, and a sentence-level semantic relation, and has hidden variables including alignment edits between sentences and their semantic relations, only requires sentences pairs annotated with sentence-level semantic relations as training data to learn appropriate alignments. In evaluation on a dataset including diverse linguistic phenomena, our proposed method achieved a competitive results on alignment prediction, and significant improvements on a sentence-level semantic relation recognition task compared to an alignment supervised model. Our analysis did not provide evidence that directly learning alignments and their labels using gold standard alignments contributed to semantic relation recognition performance and instead suggests that they can be detrimental to performance if used in a manner that prevents the learning of globally optimal alignments.

KEYWORDS: Recognizing Textual Entailment, Natural Logic, Latent Variable Model.

KEYWORDS IN L_2: .

Proceedings of COLING 2012: Technical Papers, pages 2805–2820,
COLING 2012, Mumbai, December 2012.

1 Introduction

Recognizing Textual Entailment (RTE) (Dagan et al., 2005) is the task of recognizing entailment relations between a given text pair, *Text T* and *Hypothesis H*. RTE is useful for many information access tasks that depend on natural language processing technologies, and a breakthrough would lead to significant progress in information retrieval, document summarization, and question answering, among other tasks.

The majority of approaches proposed in previous work recognize entailment relations between a pair of texts by capturing lexical or structural correspondences. Methods include simple word overlap-based measures (Jijkoun and de Rijke, 2005) as well as alignment of syntactic and semantic dependencies (Sammons et al., 2009; Wang and Zhang, 2009). However, sentence-level semantic relations are affected by various linguistic phenomena: not only lexical semantic relations (synonyms, antonyms) but also monotonicity (e.g. downward-monotone caused by scope of negation), implicative/factive expressions, quantifiers, etc. Thus similarity measures are insufficient to capture these phenomena and their interactions.

Transformation-based approaches are one way to capture the affects of diverse linguistic phenomena and their interactions, where a set of linguistic phenomena are decomposed into units. By doing so it becomes possible to consider their effects on entailment independently. A number of previous works explores transformation-based entailment relation recognition. The approach of Stern et al. (2011) recognizes a sentence-level semantic relation through a proof which represents a sequence of edits from T to H produced by applying various entailment rules and the operations such as insertion, deletion, moving subtrees, etc. In addition, Heilman and Smith (2010) proposed a tree edit model which selects a sequence of edits using Tree Kernels, and Wang and Manning (2010) proposed a latent variable model which consider possible alignments as hidden structures. However, these model do not sufficiently represent interactions between linguistic phenomena such as factuality reversals caused by negation and flipping of entailment direction under downward-monotone contexts. In order to realize precise entailment relation recognition, we need to appropriately deal with semantic relations resulting from the interaction between linguistic phenomena.

One of the most promising approaches to RTE is Natural Logic-based recognition (MacCartney and Manning, 2008; MacCartney, 2009). This approach represents transformations from T to H with a set of three types of alignment edits (*substitution*, *insertion* and *deletion*), and assigns one of a set-theoretically defined semantic relations to each alignment edit. This approach is based on the principle of compositionality, i.e. the sentence-level semantic relation is derived by combining semantic relations of edits using pre-defined composition rules. By doing so, this approach makes progress toward precise sentence-level entailment relation recognition that considers linguistic phenomena and their interactions when assigning semantic relations.

However, several issues remain unexplored. While it is common for alignment inference methods to require data annotated with alignments, it is a challenge to manually annotate alignments in a consistent manner. Annotation of alignments with semantic relations from Natural Logic is a greater challenge due to the complex nature of the semantic relations. In addition, even alignments can be annotated consistently, there is no guarantee of their *global optimality*; that is to say the alignments identified as correct by annotators may not necessarily contribute to identifying the correct semantic relation between a pair of sentences. Identifying alignments considering the full context of a sentence pair is a much more difficult annotation task. However, even without manual alignment annotations, it may be possible to infer consistent and plausible alignments by learning models that promote alignments which agree with annotations of correct semantic relations between sentences. A unified model of alignment and semantic relation

recognition between sentences is needed that learns the alignments which will generate the correct semantic relation by considering the interaction between diverse linguistic phenomena.

In this paper, we propose a novel latent discriminative model that jointly handles predicting alignment edits, classification of their semantic relations and entailment relation recognition by providing a joint distribution of variables including alignment edits, their local semantic relations and sentence-level semantic relations. Inspired by the Natural Logic-based approach of (MacCartney et al., 2008), we incorporate the set of semantic relations and their composition rules from Natural Logic into our proposed model. In addition, our model can be trained from only sentence-level semantic relations to predict alignments and semantic relations that are consistent with Natural Logic composition. To the best of our knowledge, our study is the first work to propose a latent model for training a Natural Logic-based semantic relation recognition system that does not require alignment annotations and that jointly predicts plausible alignments and semantic relations between sentences, modeling a variety of linguistic phenomena and their interactions in a compositional manner.

2 Natural Logic

The concept of *Natural Logic*, a logic over natural language, is originally proposed by Lakoff (1970), and then van Benthem (1988, 1991) and Valencia (1991) explored monotonicity calculus[1] to explain entailment relations using Natural Logic. While they considered only containment relations, MacCartney and Manning (2008) introduced an *exclusion* relation to deal with entailment relations which involve different objects or concepts (e.g. Stimpy is a cat $\models$ Stimpy is not a poodle). In this section, we describe the theory of Natural Logic proposed by (MacCartney and Manning, 2008; MacCartney, 2009).

The basic idea of MacCartney et al's theory is that the semantic relation between sentences can be derived from the semantic relations of *edits* (substitution, deletion and insertion) from T to H. The fundamental assumption of the theory is *compositionality*: (some of) the entailments of a compound expression are a function of the entailments of its parts. They defined the seven types of semantic relations for edits: equivalence ($a \equiv b$ if $a = b$), forward-entailment ($a \sqsubset b$ if $a \subset b$), backward-entailment ($a \sqsupset b$ if $b \supset a$), negation ($a \,^\wedge\, b$ if $a \cap b = \phi \wedge a \cup b = U$)[2], alternation ($a \mid b$ if $a \cap b = \phi \wedge a \cup b \neq U$), cover ($a \cup b \neq \phi \wedge a \cup b = U$), and independence ($a \# b$ otherwise).

Semantic relations provided by edits are *projected* onto other relations depending on their contexts using *projection rules*. For example, in a scope of negation, forward-entailment is projected onto backward-entailment (e.g *soccer* $\sqsubset$ *sports*, *I didn't play soccer.* $\sqsupset$ *I didn't play sports.*). Other linguistic expressions such as logical connectives and quantifiers also projects semantic relations. A semantic relation between sentences is derived by combining the projected semantic relations of edits using *composition rules*. The rules are defined as tuples of semantic relations. Let the seven types of relations be $\mathscr{R}$, $r_i \in \mathscr{R}$, $r_j \in \mathscr{R}$, then a compositional rule is represented by $r_i \bowtie r_j \Rightarrow \boldsymbol{r} \subseteq \mathscr{R}$. Some compositional rule derive a single relations (e.g. $\equiv \bowtie \sqsubset \Rightarrow \sqsubset$), and others derive more than one semantic relations (e.g. $\mid \bowtie \mid \Rightarrow \bigcup \{\equiv, \sqsubset, \sqsupset, \mid, \#\}$). As semantic relation composition proceeded, semantic relations tend to move toward $\#$ [3].

[1] In an *upward-monotone* context, replacing a linguistic expression with a more general expression preserves truth. On the other hand, in a *downward-monotone* context, replacing a linguistic expression with a more specific expression preserves truth.

[2] U denotes a universe.

[3] Due to spacial limitations, we can not give all of the composition rules. For more details, see (MacCartney, 2009).

3 A Latent Discriminative Model for Compositional Entailment Relation Recognition

Given a text T and a hypothesis H, the task of RTE is to infer the correct semantic relation between T and H. However, we attempt to learn not only the correct semantic relation between T and H but also the characteristics of the alignments most likely to support that relation.

We assume that sentence-level semantic relations can be derived compositionally. Following the framework of Natural Logic proposed by (MacCartney and Manning, 2008; MacCartney, 2009), our proposed model assigns local semantic relations to edits which represent a transformation from T to H. A valid set of edits represents an alignment between T and H. Each edit is categorized as one of three types: *substitution*, *deletion* or *insertion*, and is given one of the seven semantic relations defined in Natural Logic described in §2. A semantic relation between T and H is derived from a set of semantic relations of alignment edits by using the projection rules and the composition rules.

The proposed model learns appropriate alignments which are consistent with compositional rules of Natural Logic from only sentence-level semantic relations, where appropriate alignments, their semantic relations and their projections are represented using hidden variables. We use a log-linear discriminative model with hidden variables to provide conditional joint probabilities of alignments, their associated semantic relations, and their projections and a sentence-level semantic relation.

3.1 Model

Our proposed model provides a conditional joint distribution of alignment edits, their semantic relations, their projected relations and the final semantic relation between T and H as follows.

$$p(\boldsymbol{e}, \boldsymbol{r_e}, \boldsymbol{r_e}^P, \boldsymbol{r}^C | \boldsymbol{x}; \boldsymbol{\lambda}) = \frac{1}{Z(\boldsymbol{x})} \exp\left(\sum_k \Psi_k(\boldsymbol{e}, \boldsymbol{r_e}, \boldsymbol{r_e}^P, \boldsymbol{r}^C, \boldsymbol{x}; \boldsymbol{\lambda}) \right) \quad (1)$$

$\boldsymbol{e} = \{e_i\}$ denotes the variables representing edits, and each edit $e_i = \langle \boldsymbol{t}_i, \boldsymbol{h}_i \rangle$ consists of $\boldsymbol{t}_i$, a subset of indices of units (e.g. words) in T, and $\boldsymbol{h}_i$, a subset of indices of units in H. An edit corresponds to substitution if $\boldsymbol{t}_i \neq \phi$ and $\boldsymbol{h}_i \neq \phi$, deletion if $\boldsymbol{t}_i \neq \phi$ and $\boldsymbol{h}_i = \phi$, and insertion if $\boldsymbol{t}_i = \phi$ and $\boldsymbol{h}_i \neq \phi$. $\boldsymbol{r_e}$ represents the set of semantic relations for $\boldsymbol{e}$, where $r_{e_i} \in \boldsymbol{r_e}$ corresponds to the semantic relation of e_i. Since r_{e_i} is derived without considering its context, r_{e_i} can be seen as the semantic relation between $\boldsymbol{t}_i$ and $\boldsymbol{h}_i$. The variables $\boldsymbol{r_e}^P$ represents a set of projected semantic relations derived from $r_{\boldsymbol{e}}$, taking into account their contexts. If an edit is under the scope of negation, a quantifier or a conditional, then r_{e_i} is mapped to an appropriate semantic relation $r_{e_i}^P$ based on that context. Therefore $r_{e_i}^P$ can be seen as the sentence-level semantic relation between T and the sentence which can be obtained by applying the edit e_i to T. The variables $\boldsymbol{r}^C$ denotes a set of semantic relations derived by combining $\boldsymbol{r_e}^P$, where each $r^C \in \boldsymbol{r}^C$ corresponds to the result of composition of two semantic relations. Hereafter, we use r_T^C as the sentence-level semantic relation. Note that $r_T^C \in \boldsymbol{r}^C$. Each variable r in $\boldsymbol{r_e}$, $\boldsymbol{r_e}^P$ and $\boldsymbol{r}^C$ can have seven types of semantic relations described previously. Ψ_k in equation (1) is a *factor* which scores the plausibility of alignment edits, their semantic relations, etc.

Our proposed model uses the following four types of factors to score the plausible alignment edits, their semantic relations and a sentence-level semantic relation.

Alignment Factor $\Psi_A(e, \boldsymbol{x})$ is used to deal with (unlabeled) phrase alignment for entailment relation recognition and is defined as $\Psi_A(e, \boldsymbol{x}) = \boldsymbol{\lambda} \cdot \boldsymbol{f}_A(e, \boldsymbol{x})$. In order to provide good alignments, it is necessary to capture the lexical similarity between words. The features used in this factor are mainly (i) surface-based similarity between alignment units, (ii) semantic relatedness of alignment units, which can be extracted from diverse lexical knowledge databases, and (iii) the contextual information for an edit.

Alignment Semantic Relation Factor $\Psi_S(e, r_e, \boldsymbol{x})$ is introduced to provide plausibility of a semantic relation $r_e \in \boldsymbol{r_e}$ for an alignment edit $e \in \boldsymbol{e}$ and is defined as $\Psi_S = \boldsymbol{\lambda} \cdot \boldsymbol{f}_S(e, r_e, \boldsymbol{x})$. Each variable r_e has a distribution over the seven types of semantic relations defined in Natural Logic. In order to classify semantic relations, not only surface-based similarities, but also lexical semantic relations play an important role. In the NatLog system developed by (MacCartney, 2009), an implementation of an RTE system of Natural Logic, lexical resource-derived features (e.g. WordNet, NomBank, etc.), string similarity features, and lexical category features are used. For this factor, we exploit diverse lexical resources to provide informative features for classifying semantic relations of edits.

Projection Factor $\Psi_P(r_e, r_e^P, \boldsymbol{x})$ provides an appropriate projection from r_e to r_e^P by considering the context of e, and is defined by $\Psi_P(r_e, r_e^P, \boldsymbol{x}) = \boldsymbol{\lambda} \cdot \boldsymbol{f}_P(r_e, r_e^P, \boldsymbol{x})$. This factor captures the effects of monotonicity (e.g. upward, downward). Given r_e and its contexts, the semantic relation of the projected variable r_e^P is uniquely determined using the monotonicity rules of (MacCartney, 2009).

Composition Factor $\Psi_C(r_{i-1}^C, r_e^P, r_i^C, \boldsymbol{x})$ scores tuples of semantic relations, and is defined by $\Psi_{C_i}(r_{i-1}^C, r_e^P, r_i^C, \boldsymbol{x}) = \boldsymbol{\lambda} \cdot \boldsymbol{f}(r_{i-1}^C, r_e^P, r_i^C, \boldsymbol{x})$. In this factor, we use the composition rules used in (MacCartney, 2009) with some modification. We set the derived semantic relations to independence (#) for the rules which derive more than one semantic relations. Therefore, as with Ψ_P, given two semantic relations r_{i-1}^C and r_e^P, the joined relation of the variable r_i^C is uniquely determined.

An overview of the proposed model is shown in Figure 1. In this figure, we show the factor graph constructed by our proposed model for a pair of sentences in Japanese. Our model is divided by three layers: the *alignment layer*, the *projection layer* and the *composition layer*. First, in the alignment layer, our proposed model scores possible alignments using Ψ_A and Ψ_S. For alignment units, we use *bunsetsu* which is a reasonable unit for Japanese linguistic analysis. A *bunsetsu* is a chunk-like unit that consists of one or more content words and zero or more functional words. A set of possible alignments are obtained using an extended MANLI algorithm (MacCartney et al., 2008).

Next, for each alignment obtained by the alignment algorithm, we construct a factor graph as shown in Figure 1. The factor graph has variables for alignments, projected relations, joined relations, and the factors defined previously. In the projection layer, semantic relations of alignments are projected by Ψ_P, and finally a sentence-level semantic relation is obtained in the composition layer using the projected relations and composition rules encoded in Ψ_C.

In inference, since variables related to Ψ_P and Ψ_C are uniquely determined if $\boldsymbol{r_e}$ is given, the model derives the best alignments, their semantic relations, and a sentence-level semantic relation simultaneously. In training, the parameters of the model are updated so as to derive alignments and their semantic relations which derive the correct sentence-level semantic relation based on the composition rules of Natural Logic.

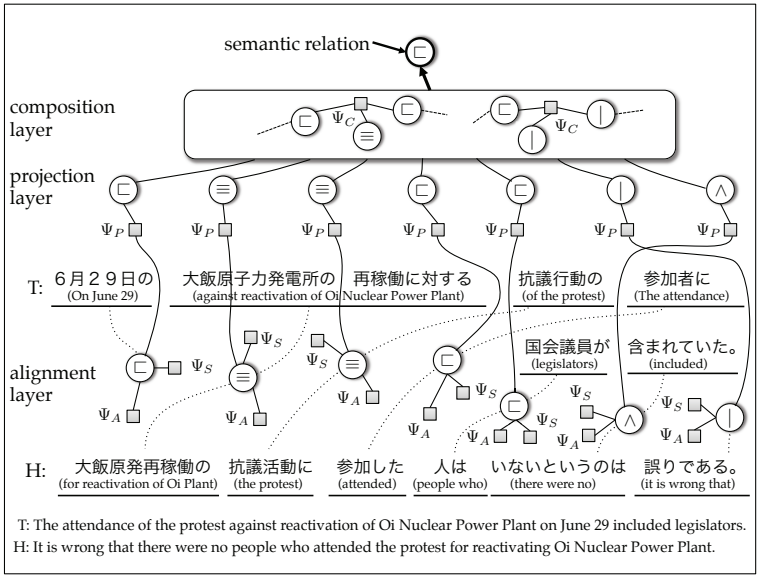


Figure 1: An overview of the proposed model.

3.2 Features

The features used in the proposed model are listed in Table 3.2.

Because RTE datasets are small, it is difficult to incorporate lexical features directly into our model as they may cause overfitting. Instead, we incorporate similarity metrics to model lexicality. On the other hand, because function words are closed class and present in all texts, we can directly use them as features.

While both Ψ_A and Ψ_S score the plausibility of alignments, the features used in their factors are also different. Ψ_A considers not only lexical similarities but also contexts of edits. Let us consider a simple sentence pair *T: USA won the war but Japan lost the war. H: Japan won the war.* In this example, T and H share the same verb *won* but the word *won* in H should be aligned to *lose* in T because they share the same subject (*Japan*). So, we introduce features that capture predicate-argument structure-level contextual information: e.g. how many arguments are shared by the two predicates (NUM_SHARED_ARGS)? On the other hand, Ψ_S pays more attention to inferring the lexical semantic relations of edits. The features used in Ψ_p and Ψ_C work as rules to infer sentence-level semantic relations.

3.3 Learning the Model

The parameters $\boldsymbol{\lambda}$ of the proposed model are trained from sentence-level semantic relations via marginal-likelihood maximization $\mathscr{L}_{\boldsymbol{\lambda}} = \sum_n \log p(r_T^C = l^n | \boldsymbol{x}^n; \boldsymbol{\lambda})$. By applying this objective function, we expect that the proposed model is trained so as to prefer alignment edits and

Factor	Edit	Name	Description
Ψ_A	DEL, INS	TYPE	edit type of e
		SIZE	the number of *bunsetsus* in e
		SAME_NOMINATIVE_IN_{T,H}	1 if e has a nominative argument and the other sentence also contains a nominative argument and its lemmas are the same.
		SAME_CASE_IN_{T,H}	1 if e has an argument of some predicates and the other sentence also contains an argument and its cases are the same.
		{T,H}_CONTAINS_{H,T}_LEMMA	1 if the head word of the *bunsetsu* in e is also contained in the other sentence.
		POS_SEQ	POS sequence in e
		HEADPOS	head POS of e
Ψ_A	SUB	TYPE	edit type of e
		SIZE	the number of *bunsetsus* in e
		NUM_SHARED_ARGS	the number of shared arguments if both t and h in e are predicates
		PARTICLE_SAME	1 if t and t have the same particle
		JAPANESE_WORDNET	Set relation type if e matches an entry in Japanese WordNet (Bond et al., 2009).
		WIKIPEDIA_HYPERNYM-HYPONYM	1 if e matches an entry in (Sumida et al., 2008).
		VERB_ENTAILMENT_REL	1 if an entry in the verb entailment relation dictionary (Hashimoto et al., 2009) matches e
		VERB_RELATION_REL	1 if an entry in the verb relation dictionary (Matsuyoshi et al., 2008) matches e
		PARENT_NUM_SHARED_ARGS	the number of shared arguments of the parent of t and h if t and h are arguments
		BOTH_HAVE_A_ROLE	1 if each *bunsetsu* in e is an argument of a predicate.
		BOTH_HAVE_THE_SAME_ROLE	1 if each *bunsetsu* in e has the same case.
		HEAD_POS_SAME	1 if the POSs of the heads in e are the same
		POS_SEQ_SAME	1 if the POS sequences of chunks in e are the same
		EXACT_MATCH	1 if t and h are the same
		UNIGRAM_COSINE	return unigram cosine value if the cosine similarity of two chunks in e is greater than the pre-defined threshold
Ψ_S	DEL, INS	SIZE	number of *bunsetsus* in e
		HEAD_LEMMA	lemma of the head of *bunsetsu* in e
		HEAD_WORD_CLASS	Word class of the head of *bunsetsu* in e. The word class information is extracted from the dictionary provided by (Kazama et al., 2010)
		NEGATION	1 if the *bunsetsu* contains a negation
Ψ_S	SUB	SIZE	Pair of the number of *bunsetsus* in e
		HEAD_POS_PAIR	POS pair of the heads of *bunsetsus* in e
		HEAD_LEMMA_SAME	1 if the lemmas of the heads in e are the same
		POS_SEQ_SAME	1 if the POS sequences of chunks in e are the same
		JAPANESE_WORDNET	same as in Ψ_A
		WIKIPEDIA_HYPERNYM-HYPONYM	same as in Ψ_A
		VERB_ENTAILMENT_REL	same as in Ψ_A
		VERB_RELATION_REL	same as in Ψ_A
Ψ_P	–	MONOTONE_{UP/DOWN}	the context of e is *upward-monotone* or *downward-monotone*.
Ψ_C	–	COMPOSITION_RULE	1 if the tuple of semantic relations is included in a set of defined compositional rules.

Table 1: Features used for the model.

their semantic relations which infer the correct sentence-level semantic relations based on the composition rules of Natural Logic.

The partial differential of the objective function is

$$\frac{\partial L}{\partial \lambda_k} = \sum_n \left(\frac{\partial}{\partial \lambda_k} \log \sum_{\langle \boldsymbol{e}, \boldsymbol{r}_e \rangle \in \mathcal{E}} \sum_{\boldsymbol{r}: r_T^C = l} \exp\left(\sum_k \boldsymbol{\Psi}_k(\boldsymbol{e}, \boldsymbol{r}_e, \boldsymbol{r}_e^P, \boldsymbol{r}^C, \boldsymbol{x}) \right) - \frac{\partial}{\partial \lambda_k} \log Z(\boldsymbol{x}) \right) \quad (2)$$

Algorithm 1 The alignment algorithm.

input
an example $(\boldsymbol{x}_T, \boldsymbol{x}_H)$, number of iterations I, max size of edits M,
number of N-bests N, score difference δ, score function $\Psi_A(\boldsymbol{e}, \boldsymbol{x}) + \Psi_S(\boldsymbol{e}, \boldsymbol{r}_e, \boldsymbol{x})$
initialize
$\boldsymbol{e}_0 \leftarrow \phi$
$\forall x_T \in \boldsymbol{x}_T\ \boldsymbol{e}_0 \leftarrow \boldsymbol{e}_0 \cup e(x_T, DEL, \sqsubset)$
$\forall x_H \in \boldsymbol{x}_H\ \boldsymbol{e}_0 \leftarrow \boldsymbol{e}_0 \cup e(x_H, INS, \sqsupset)$
all alignments $\mathscr{E} \leftarrow \boldsymbol{e}_0$
while $(iter < I)$ **do**
  top alignments $\mathscr{E}_{top}$
  max score $score_{argmax}$
  repeat
    get top alignment $\boldsymbol{e}_{top}$ from $\mathscr{E}$
    if $s(\boldsymbol{e}_{top}) > score_{argmax}$ **then**
      $score_{argmax} \leftarrow s(\boldsymbol{e}_{top})$
    end if
    $\mathscr{E}_{top} \leftarrow \mathscr{E}_{top} \cup \boldsymbol{e}_{top}$
  until $score_{argmax} - s(\boldsymbol{e}_{top}) >= \delta$
  get successors $\mathscr{S}_i = \{\boldsymbol{e}_i^s\}_i$ for $\boldsymbol{e}_i \in \mathscr{E}_{top}$
  $\mathscr{E} \leftarrow \mathscr{E} \bigcup_i \mathscr{S}_i$
end while
return $\mathscr{E}$

where edits $\boldsymbol{e}$, their semantic relations $\boldsymbol{r}_e$, projected semantic relations $\boldsymbol{r}_e^P$ and joined relations excluding the sentence-level semantic relation are all hidden variables. Given $\boldsymbol{r}_e$, $\boldsymbol{r}_e^P$ and $\boldsymbol{r}^C$ can be identified uniquely by using projection and composition rules. Since the objective function is non-convex, estimated parameters can be local-optima.

In optimization, only the parameters in Ψ_A and Ψ_S are updated, and the parameters in Ψ_P and Ψ_C are left to initial values. In order to update the parameters, we need to calculate marginal probabilities of the alignments. However, unlike sequential or tree models, calculating exact values of alignments is prohibitively difficult. We use only N-bests provided by the extended MANLI algorithm to calculate an approximate partition function $\widetilde{Z}(\boldsymbol{x})$ instead of $Z(\boldsymbol{x})$, and approximate marginal probabilities.

3.4 Inference of Alignments

Given two sentences, the problem of alignment inference in our model is predicting the best edits and their semantic relations $\widehat{\langle \boldsymbol{e}, \boldsymbol{r}_e \rangle} = \arg\max_{\langle \boldsymbol{e}, \boldsymbol{r}_e \rangle \in \mathscr{E}} \sum_{\langle e_i, r_{e_i} \rangle \in \langle \boldsymbol{e}, \boldsymbol{r}_e \rangle} \Psi_A(e_i, \boldsymbol{x}; \boldsymbol{\lambda}) + \sum_{\langle e_i, r_{e_i} \rangle \in \langle \boldsymbol{e}, \boldsymbol{r}_e \rangle} \Psi_S(e_i, r_{e_i}, \boldsymbol{x}; \boldsymbol{\lambda})$ where $\mathscr{E}$ is a set of all possible edits and their semantic relations between two sentences. The original MANLI algorithm (MacCartney et al., 2008) only provides the best edits, so we extend the algorithm so as to provide not only edits, but also their semantic relations.

The extended version of MANLI is shown in Algorithm 1. Given two sentences, the algorithm starts at an initial alignment $\boldsymbol{e}_0$ which consists of deletion edits of *bunsetsus* in T and insertion edits of *bunsetsu* in H, and then searches for more good alignments by changing edits from a pair of a deletion and an insertion edit to a substitution edit, or changing semantic labels. The main differences between the original MANLI and our algorithm are: (1) alignments have their semantic relations, (2) keeps a set of alignments ordered by scores provided by Ψ_A and Ψ_S to provide N-bests. We omitted the annealing procedure which is included in the original

MANLI because our algorithm need to keep an ordered set of alignments based on scores. If we introduce a temperature value, we have to update all of the alignments in the set when the value is changed. However this is computationally expensive.

3.5 The Order of Composition

The composition order of semantic relations defined in Natural Logic is non-commutative. Let us consider joining an alternation ($|$) and a forward-entailment ($\sqsubset$). The pair of semantic relations frequently appear in contradiction examples. $\sqsubset$ joined with $|$ yields $|$, on the other hand, $|$ joined with $\sqsubset$ yields $\bigcup\{\equiv, ^\wedge, |, \cup, \#\}$. The former way of composition derives the desired result, however, the latter way derives an ambiguous result. We defined the order of composition so as to keep joined semantic relations unambiguous as far as possible. Our proposed model at first joins $\equiv$ and $\sqsubset$, then $|$, then $^\wedge$, and $\sqsupset$ in the end [4].

4 Experiments

4.1 Data

We developed a dataset for semantic relation recognition which includes a diverse selection of linguistic phenomena. Although there is an textual entailment recognition data set for Japanese (RITE (Shima et al., 2011)), we do not consider it an appropriate target for evaluation and instead construct our own dataset. Our motivation is as follows. Much of the progress made in textual entailment recognition has been on a set of phenomena that can be handled with methods of lexical and phrasal similarity, however, there are many other phenomena that have not been addressed.

Sammons et al. (2010) make a case for more detailed analysis of the linguistic phenomena important to textual entailment recognition so that their impact on existing approaches can be properly measured. In that spirit, we investigated textual entailment recognition phenomena and found that quantification, negation, and monotonicity require consideration of their semantic structure and are beyond the scope of similarity-based methods. Constructing systematic and robust models of handling these phenomena is the focus of this paper. It is reasonable to target these phenomena next because many of the remaining problems for textual entailment recognition require world knowledge and are thus problems of inference or AI. Existing datasets for textual entailment recognition are insufficient for our purposes because the phenomena they contain are too broad and they do not contain enough examples of the phenomena we are targeting to draw meaningful conclusions.

We selected the categories based on FraCaS (Cooper et al., 1996), the corpus developed by Bentivogli et al. (2010) and the categories discussed in (MacCartney, 2009): lexical semantic relation (e.g. synonym, antonym, hypernym-hyponym relation), quantifiers, modifiers, negation, coordination, relative clauses, apposition, temporal and numerical expressions, active/passive, factive verbs and functional relations.

The statistics of the dataset is shown in Table 4.1. The distribution of the categories is not balanced: the quantifier category accounts for approximately 30% of the total. One of our interests is whether the model can automatically capture behaviour of functional expressions such as quantifiers from sentence-level semantic relations. In order to conform this point, we developed many examples for quantifiers.

[4] NatLog uses a different strategy from ours. The system at first joins semantic relations of deletion edits, then substitution edits, next edits involve operators with non-default projectivity, and, finally, insertion edits.

Category	#	Category	#	Category	#
Quantifier	182	Coordination	27	Part-Whole	13
Numerical/Temporal	53	Argument Mismatch	26	Condition	8
Modifier	55	Negation + Lexical semantic Rel.	23	Apposition	7
Lexical semantic relation	44	Paraphrase	21	World Knowledge	3
Implicative/Factive	36	Predicate Mismatch	17	Other	37
Relation between entities/events	27	Coordination	27	Total	598

Table 2: Category statistics.

Category	Example	Sem. Rel.
Quantifier	*T: Almost all mammals have molars in the back of their rows of teeth.* *H: There are mammals that do not have molars.*	Forward-Ent.
Paraphrase	*T: Not all smokers get cancer.* *H: Even if you smoke, you might not get cancer.*	Paraphrase
Modifier	*T: There aren't many students.* *H: There aren't many students who have had a heat stroke.*	Forward-Ent.
Lexical semantic rel.	*T: Japan got a bronze medal in Team Fencing.* *H: Japan hasn't gotten a bronze medal in any sports.*	Contradiction
Implicative/Factive	*T: Earthquake-proofing prevented the house's collapse.* *H: The house did not collapse.*	Forward-Ent.
Coordination	*T: Tokyo has a population of 13,000,000 and Miyagi has a population of 2,300,000.* *H: Tokyo has a population of 2,300,000.*	Contradiction

Table 3: Some examples in the dataset (translated in English).

For each example, we annotated one of the four types of sentence-level semantic relations (paraphrase, forward-entailment, contradiction and independence), and alignment edits and their semantic relations in Natural Logic. In the dataset, the number of paraphrase examples is 97, forward-entailment is 313, contradiction is 100, and independence is 88. Table 4.1 shows some examples in the dataset. The dataset was developed by one annotator, who is a professionally trained linguist unaffiliated with this research project, and the set of annotated semantic relations does not always provide the correct semantic relation. 55.2% of the gold annotations derive correct sentence-level semantic relation (332 examples). The remaining examples include inconsistencies between sentence-level semantic relations and semantic relations of alignments, linguistic phenomena that the current model can not deal with (e.g. syntactic transformation, some quantifiers), etc.

Whereas there are seven types of relations in Natural Logic, our annotation uses only four types of relations. So in the experiments, we mapped *contradiction* to $\{\wedge, |\}$ and *other* to $\{\cup, \#\}$ in the training and the testing phase.

4.2 Settings

In order to explore the effectiveness of the proposed model, we evaluated the following approaches in the experiments.

Initial Weight Initial weights of the model are used for testing.

Resource-based Alignment Alignments are determined based on a surface-based similarity measures and lexical resources. In this setting, a pair of two phrases is aligned if the character-bigram cosine similarity is greater than a pre-defined threshold (we set it to 0.8), or the pair matches an entry in the lexical resources such as Japanese WordNet, Hypernym-Hyponym relations, Verb Entailment Relations, and Verb Relation Dictionary.

Semantic relations of alignment edits are determined as follows: $\equiv$ if the pair of the *bunsetsus* $\langle b_1, b_2 \rangle$ is the same, similar or is synonym, | if the pair is antonym, $\sqsubset$ if b_2 is the hypernym of b_1 or b_1 entails b_2, $\sqsupset$ if b_1 is the hypernym of b_2 or b_2 entails b_1. The *bunsetsus* not aligned by the similarity measure or the resources are converted to *deletion* or *insertion* edits, and their semantic relations are set to $\sqsubset$ and $\sqsupset$ respectively with exceptions described later.

Alignment Supervised The model is trained using gold alignments which have correct semantic relations defined in Natural Logic. In this setting, sentence-level semantic relations are not considered in training. As in the proposed model, we constructed the model using a log-linear discriminative model, and the model was trained log-likelihood maximization of gold alignments. The objective function used in training was $\mathcal{L}_{\boldsymbol{\lambda}} = \sum_n \log p(\boldsymbol{e}, \boldsymbol{r}_e | \boldsymbol{x}^n; \boldsymbol{\lambda})$.

Weakly Supervised (proposed) The model is trained by marginal likelihood maximization over sentence-level semantic relations.

The dataset we used in the experiments include the examples whose correct sentence-level semantic relations can not be derived from the pre-annotated semantic relations of alignment edits. It seems that these are hard to derive correct sentence-level semantic relations from the current possible edits. So, we conducted experiments on the examples whose correct sentence-level semantic relations can be derived from the gold alignments (hereafter, we say *reachable*).

For the factors Ψ_P and Ψ_C, we initialized the weights to 0.0 if the semantic relation tuple is covered by our projection rules and composition rules, and $-\infty$ otherwise. For the factors Ψ_A and Ψ_S, we set initial weights to some features [5]. In training of the model, we update the parameters in Ψ_A and Ψ_S, and the parameters in Ψ_P and Ψ_C are left to the initial values. Parameter updating was performed using stochastic gradient descent (SGD), and the number of iterations was set to 2. Also, we applied L_2 regularization. As for the alignment algorithm, the number of iterations was set to 40, and the number of N-bests was set to 10. For each edit type, we restricted the maximum size of units: only allows one-to-one for substitution, allows at most three units for insertion and deletion edits. Also, we constrained the types of semantic relations for each edit type. Substitution edits can have one of the five types of semantic relations: $\equiv$, $\sqsubset$, $\sqsupset$, $\wedge$ and | with an exception. If the lemma sequences of the two *bunsetsus* are the same, the edit can have only $\equiv$. Deletion edits and insertion edits can have $\sqsubset$ and $\sqsupset$ respectively with exceptions. They can have | if the head of *bunsetsu* matches an entry in the list of *counter-factive expressions* [6], and they can have $\equiv$ if the head of *bunsetsu* matches an entry in the list of *less-informative expressions* [7].

4.3 Evaluation Measures

We use the following measures in evaluation: **(1) Alignment (Unlabeled):** A predicted alignment is correct if there is a gold alignment which has the same span, but the semantic label is not considered, **(2) Alignment (Labeled):** A predicted alignment is correct only if there is a gold alignment which has the same span and their semantic relations are also the same, and **(3) Sem. Rel.:** Accuracy of sentence-level semantic relations.

[5] For instance, the weights of the combination feature "NEGATION=0" and "JAPANESE_WORDNET=antonym" are set to 1.0 if label is | and -1.0 otherwise

[6] A hand-crafted list which contains 13 entries.

[7] As with the list of *counter-factive expressions*, the list was hand-crafted, and contains 30 entries.

	Alignment (Unlabeled)			Alignment (Labeled)			Sem. Rel.
	Prec.	Rec.	F1	Prec.	Rec.	F1	Acc.
Initial Weights	42.6	62.5	50.6	37.5	54.9	44.5	31.8
Resource-based Alignment	45.6	63.3	53.0	38.6	53.5	44.9	35.0
Weakly Supervised (proposed)	67.1	67.8	67.4	51.3	51.9	51.6	**47.5**
Alignment Supervised	68.0	68.8	**68.4**	54.9	55.5	**55.2**	43.7
Gold Alignment	100.0	100.0	100.0	100.0	100.0	100.0	55.2
reachable examples only							
Initial Weights	45.6	65.5	53.8	41.7	59.9	49.2	38.5
Resource-based Alignment	48.4	66.7	56.1	42.6	58.8	49.4	37.7
Weakly Supervised (proposed)	72.4	73.1	72.7	59.2	59.8	59.5	**60.2**
Alignment Supervised	74.2	75.0	**74.6**	61.9	62.6	**62.3**	46.4

Table 4: Performance of alignment prediction and sentence-level semantic relation recognition.

4.4 Preprocessing

For each sentence, we conducted various forms of linguistic analysis: morphological analysis using MeCab (Kudo et al., 2004), syntactic parsing using the Japanese dependency parser, CaboCha (Kudo and Matsumoto, 2002) and predicate-argument structure analysis (Watanabe et al., 2010) to provide a basis for alignment and semantic relation classification.

4.5 Results

Table 4 shows the experimental results of 10-fold cross validation for alignment prediction and sentence level semantic relation recognition. We can see that while the proposed method is less successful at reproducing gold standard alignments, it greatly outperforms Supervised Learning for sentence-level semantic relation recognition[8]. We expected Supervised Learning to perform best on reachable examples, which should have the most straightforward connection between alignment semantic relation labels and sentence level semantic relations. Nevertheless, our proposed method achieved the best performance on this dataset as well. These results support our theory that gold standard alignment data is necessary for semantic relation recognition. Indeed, alignment labels appear to degrade performance in several cases.

Table 5 shows the sentence-level performance for each semantic relation type. This breakdown shows that the proposed method is particularly good at Contradiction and Forward-Entailment relations, outperforming all other methods on all data sets. When considering reachable examples only, it is also the top-performing method for Paraphrase detection as well. Resource-based Alignment and Initial Weights both perform poorly, producing significantly worse results than the supervised methods in every evaluation setting with the exception of Contradiction on reachable examples only and Independence on both data sets.

The poor performance by Resource-based Alignment and Initial Weights is likely due to inaccurate alignments, especially of functional expressions (e.g. *sometimes - not always*). Since deletion and an insertion edits are assigned $\sqsubset$ and $\sqsupset$ respectively by default and joining them yields independence (#), these methods over-produce Independence relations. Most of the errors in Alignment Supervised are caused by lower precision for alternation (|). Since alternation relations can greatly impact the sentence-level semantic relation prediction, this severely impacted the overall performance of the supervised model.

Table 6 shows the performances of semantic relation classification of alignments for each type. As discussed before, supervised alignment is the most successful at recovering gold standard

[8] We compared the sentence-level semantic relation recognition results of Weakly Supervised and Alignment Supervised with the McNemar test, and the difference was statistically significant ($p < 0.01$).

	Paraphrase			Forward-Entailment		
	Prec.	Rec.	F1	Prec.	Rec.	F1
Resource-based Alignment	67.4	29.9	41.4	73.6	28.4	41.0
Initial Weights	61.1	11.3	19.1	81.9	27.5	41.2
Weakly Supervised (proposed)	46.1	54.6	50.0	73.9	55.3	**63.2**
Alignment Supervised	60.5	47.4	**53.2**	72.7	39.9	51.6
	Contradiction			Independence		
Resource-based Alignment	19.2	15.0	16.9	22.1	86.4	35.2
Initial Weights	41.4	12.0	18.6	18.6	92.1	31.0
Weakly Supervised (proposed)	25.0	43.0	**31.6**	33.3	17.1	22.6
Alignment Supervised	21.3	42.0	28.3	36.4	54.6	**43.6**
reachable examples only						
	Paraphrase			Forward-Entailment		
Resource-based Alignment	52.9	20.9	30.0	76.3	33.7	46.8
Initial Weights	60.0	20.9	31.0	85.7	34.9	49.6
Weakly Supervised (proposed)	59.5	51.2	**55.0**	62.3	85.5	**72.1**
Alignment Supervised	53.1	39.5	45.3	61.2	58.7	59.9
	Contradiction			Independence		
Resource-based Alignment	29.8	20.6	24.4	23.2	89.8	36.8
Initial Weights	63.2	17.7	27.6	20.6	95.9	33.9
Weakly Supervised (proposed)	73.9	25.0	**37.4**	60.9	28.6	38.9
Alignment Supervised	19.0	26.5	22.1	62.1	36.7	**46.2**

Table 5: The details of the results of sentence-level entailment relation recognition.

	Equivalence (≡)			Forward-Entailment (⊏)		
	Prec.	Rec.	F1	Prec.	Rec.	F1
Resource Alignment	60.7	76.0	67.5	25.5	49.8	33.8
Initial Weights	69.0	76.2	72.4	22.8	53.3	31.9
Weakly Supervised	66.0	89.0	75.8	30.1	25.3	27.5
Alignment Supervised	70.0	89.5	78.6	35.2	29.3	32.0
	Backward-Entailment (⊐)			Alternation (|), Negation (^)		
Resource Alignment	11.8	47.0	18.1	68.4	14.3	23.7
Initial Weights	9.7	54.8	16.4	91.2	5.6	10.5
Weakly Supervised	27.1	13.9	18.3	28.7	15.4	20.0
Alignment Supervised	34.3	15.1	20.9	23.3	17.2	19.8
reachable examples only						
	Equivalence (≡)			Forward-Entailment (⊏)		
	Prec.	Rec.	F1	Prec.	Rec.	F1
Resource Alignment	69.1	76.5	72.6	27.4	53.8	36.3
Initial Weights	75.3	76.9	76.1	25.2	57.0	35.0
Weakly Supervised	72.8	88.7	80.0	30.5	37.5	33.6
Alignment Supervised	76.8	89.5	82.7	35.1	38.0	36.5
	Backward-Entailment (⊐)			Alternation (|), Negation (^)		
	Prec.	Rec.	F1	Prec.	Rec.	F1
Resource Alignment	8.3	52.5	14.3	70.6	19.1	30.0
Initial Weights	7.2	57.6	12.8	100.0	8.3	15.4
Weakly Supervised	25.0	11.9	16.1	83.3	7.9	14.5
Alignment Supervised	28.1	15.3	19.8	40.3	19.8	26.6

Table 6: The details of the performances of alignment prediction.

alignments and semantic relation labels. However, it is interesting to note that while Resource Alignment performs competitively at alignment prediction (it rivals Alignment Supervised on Forward-Entailment and outperforms all other methods on Alternation/Negation), it performs drastically worse on sentence-level semantic relation recognition, sometimes with an f-score that is more than 20 points lower than the best performing method. These results suggest that it is important to jointly model alignment prediction and sentence-level semantic relation recognition so that globally optimal alignments are promoted.

5 Related Work

There are a number of existing works which explore the use of latent variable or structure models for recognizing textual entailment. Chang et al. (2010) proposed a discriminative linear model where alignments are treated as hidden structures, and the sentence-level semantic relation is derived based on the best latent alignment structure. They formulated the problem of predicting the best hidden structure as an Integer Linear Programming problem, where domain knowledge is encoded as constraints. Wang and Manning (2010) proposed a latent variable model where the model provides a conditional distribution of a sequence of edits, which can be seen as a transformation-based approach. In the model, edits are treated as hidden variables that populate a positive set and a negative set in the search space. Sentence-level semantic relations are predicted based on the sum of the scores of edit sequences in the positive set and the negative set.

The differences between our proposed model and theirs are that the number of semantic relations and compositionality. Both Wang and Manning (2010) and Chang et al. (2010) consider only *entailment* and *non-entailment*, while our proposed model identifies a rich set of relations: *paraphrase*, *forward entailment*, *backward entailment*, *contradiction*, and *independence*.

Furthermore, as discussed in Section 2, our model exhibits compositionality by incorporating Natural Logic at two different levels. First, it incorporates information about upward and downward monotonicity into a projection layer, allow it to handle flips in entailment direction caused by scope of negation that can influence the final sentence-level semantic relation. In addition, it considers the result of combining projected semantic relations of alignment edits, allowing it to handle complex interactions between linguistic phenomena in sentences. The alignment models of Wang and Manning (2010) and Chang et al. (2010) do not consider the interaction between alignments that we model with Natural Logic making it difficult for them to classify examples that contain complex semantic structures.

Conclusion

In this paper, we proposed a novel latent variable model for compositional entailment relation recognition. We gave the proposed model compositionality by incorporating a set of semantic relations and their composition rules of Natural Logic. The model has ability to predict local correspondences (alignments) between sentences, the semantic relations, and the sentence-level semantic relation simultaneously. The model can be trained from only sentence-level semantic relations by using marginal-likelihood maximization. In evaluation, our proposed method outperformed a supervised alignment method on a sentence-level semantic relation recognition task, and detailed analysis on that task and an alignment prediction task did not provide evidence that gold standard alignment labels contributed to semantic relation recognition performance and instead suggests that they can be detrimental to performance if used in a manner that prevents the learning of globally optimal alignments.

A future research direction we are investigating is extending the model so as to deal with structural transformations. The current model has a big drawback: the model assumes that all of sentence-level semantic relations can be derived from only *bunsetsu* -level transformations. We would like to explore how to incorporate transformation rules (used in e.g. (Stern et al., 2011)) into the proposed model.

Acknowledgements

This work is supported by the PRESTO program of JST and the Grants-in-Aid for Scientific Research No. 23240018, No. 23700157 and No. 23700159.

References

Bentivogli, L., Cabrio, E., Dagan, I., Giampiccolo, D., Leggio, M. L., and Magnini, B. (2010). Building textual entailment specialized data sets: a methodology for isolating linguistic phenomena relevant to inference. In *Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC'10)*.

Bond, F., Isahara, H., Fujita, S., Uchimoto, K., Kuribayashi, T., and Kanzaki, K. (2009). Enhancing the Japanese WordNet. In *Proceedings of the 7th Workshop on Asian Language Resources, ACL-IJCNLP 2009*, pages 1–8.

Chang, M., Goldwasser, D., Roth, D., and Srikumar, V. (2010). Discriminative learning over constrained latent representations. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 429–437.

Cooper, R., Crouch, D., van Eijck, J., Fox, C., van Genabith, J., Jaspers, J., Kamp, H., Milward, D., Pinkal, M., Poesio, M., and Pulman, S. (1996). Using the framework. Technical report, Technical Report LRE 62-051 D-16, The FraCaS Consortium.

Dagan, I., Glickman, O., and Magnini, B. (2005). The Pascal Recognising Textual Entailment Challenge. In *Proceedings of the PASCAL Challenges Workshop on Recognising Textual Entailment*.

Hashimoto, C., Torisawa, K., Kuroda, K., Murata, M., and Kazama, J. (2009). Large-scale verb entailment acquisition from the web. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing (EMNLP 2009)*, pages 1172–1181.

Heilman, M. and Smith, N. (2010). Tree Edit Models for Recognizing Textual Entailments, Paraphrases, and Answers to Questions. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 1011–1019.

Jijkoun, V. and de Rijke, M. (2005). Recognizing textual entailment using lexical similarity. In *Proceedings of the PASCAL Challenges Workshop on RTE*.

Kazama, J., Saeger, S. D., Kuroda, K., Murata, M., and Torisawa, K. (2010). A bayesian method for robust estimation of distributional similarities. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*.

Kudo, T. and Matsumoto, Y. (2002). Japanese dependency analysis using cascaded chunking. In *CoNLL 2002: Proceedings of the 6th Conference on Natural Language Learning 2002 (COLING 2002 Post-Conference Workshops)*, pages 63–69.

Kudo, T., Yamamoto, K., and Matsumoto, Y. (2004). Appliying conditional random fields to japanese morphological analysis. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, pages 230–237.

Lakoff, G. (1970). George Lakoff - Linguistics and Natural Logic. *Synthese*, 22:151–271.

MacCartney, B. (2009). *Natural Language Inference*. PhD thesis.

MacCartney, B., Galley, M., and Manning, C. (2008). A Phrase-Based Alignment Model for Natural Language Inference. *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 802–811.

MacCartney, B. and Manning, C. D. (2008). Modeling semantic containment and exclusion in natural language inference. In *Proceedings of the 22nd International Conference on Computational Linguistics (Coling 2008)*, pages 521–528.

Matsuyoshi, S., Murakami, K., Matsumoto, Y., , and Inui, K. (2008). A database of relations between predicate argument structures for recognizing textual entailment and contradiction. In *Proceedings of the 2nd International Symposium on Universal Communication (ISUC2008)*, pages 366–373.

Sammons, M., Vydiswaran, V., and Roth, D. (2010). Ask not what textual entailment can do for you... In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 1199–1208.

Sammons, M., Vydiswaran, V., Vieira, T., Johri, N., Chang, M., Goldwasser, D., Srikumar, V., Kundu, G., Tu, Y., and Small, K. (2009). Relation alignment for textual entailment recognition.

Shima, H., Kanayama, H., Lee, C.-W., Lin, C.-J., Mitamura, T., Miyao, Y., Shi, S., and Takeda, K. (2011). Overview of NTCIR-9 RITE: Recognizing Inference in TExt. In *Proceedings of NTCIR-9 Workshop Meeting*, pages 291–301.

Stern, A., Lotan, A., Mirkin, S., Shnarch, E., Kotlerman, L., Berant, J., and Dagan, I. (2011). Knowledge and Tree-Edits in Learnable Entailment Proofs. In *Proceedings of the Fourth Text Analysis Conference (TAC 2011)*.

Sumida, A., Yoshinaga, N., and Torisawa, K. (2008). Boosting Precision and Recall of Hyponymy Relation Acquisition from Hierarchical Layouts in Wikipedia. In *Proceedings of the 6th International Language Resources and Evaluation (LREC'08)*, pages 2462–2469.

Valencia, V. S. (1991). *Studies on Natural Logic and Categorial Grammar*. PhD thesis.

van Benthem, J. (1988). The semantics of variety in categorial grammars. pages 33–55.

van Benthem, J. (1991). Language in action: Categories, lambdas and dynamic logic.

Wang, M. and Manning, C. (2010). Probabilistic tree-edit models with structured latent variables for textual entailment and question answering. pages 1164–1172.

Wang, R. and Zhang, Y. (2009). Recognizing textual relatedness with predicate-argument structures. In *EMNLP-2009*.

Watanabe, Y., Asahara, M., and Matsumoto, Y. (2010). A structured model for joint learning of argument roles and predicate senses. In *Proceedings of the 48th Annual Meeting of the Association of Computational Linguistics*, pages 98–102.

Strategies for Mixed-Initiative Conversation Management using Question-Answer Pairs

Wilson Wong, Lawrence Cavedon, John Thangarajah, Lin Padgham
RMIT University, Melbourne, Australia
{wilson.wong,lawrence.cavedon,john.thangarajah,lin.padgham}@rmit.edu.au

ABSTRACT
One of the biggest bottlenecks for conversational systems is large-scale provision of suitable content. In this paper, we present the use of content mined from online question-and-answer forums to automatically construct system utterances. Although this content is mined in the form of question-answer pairs, our system is able to use it to formulate utterances that drive a conversation, not just for answering user questions as has been done in previous work. We use a collection of strategies that specify how and when the question-answer pairs can be used and augmented with a small number of generic hand-crafted text snippets to generate natural and coherent system utterances. Our experiments involving 11 human participants demonstrated that this approach can indeed produce relatively natural and coherent interaction.

KEYWORDS: conversational system; question-answer pairs; conversational strategies.

1 Introduction

A major bottleneck for any kind of *conversational system* is provision of sufficient content, and if one wishes to avoid repetition and to cover the many possible user inputs, there must be a substantial amount of content. In the case of chatbots such as ALICE (Wallace, 2009), the content is custom-crafted to be as generic and deflective as possible to cover the needs of open-ended conversations. This approach, which reduces the amount of content that the system needs despite the relatively broad range of user inputs, is essentially only capable of content-free, small talk. This kind of conversational ability is, however, inadequate for virtual characters that need to contribute relevant content in their conversations. In systems that require domain-specific content, for example about health in a virtual nurse (Bickmore et al., 2009) or about children-related topics in an intelligent toy (Chen et al., 2011), the custom-crafting of content to cover the depth and breadth of the domain as well as to keep it current becomes an impractical task. This work reports on our use of an abundant type of data, sourced from the Web, to equip a conversational agent with the content it needs to engage users in a coherent and natural way over a wide range of domains.

To overcome the bottleneck of custom- or hand-crafting conversational content, researchers have recently investigated mining online resources such as Web pages returned by search engines (e.g., `Yahoo!` (Shibata et al., 2009)). The problem with this approach, however, is that most content on the Web is not suitable for conversational agents, for various reasons. The verbose, non-colloquial and monologue nature of much of Web text is not a good match for the characteristics of human-human dialogue. However, there are some other sources on the Web that have more potential for this purpose, such as human-written comments on news websites (e.g., `NYTimes.com` (Marge et al., 2010)) and online forums (e.g., `RottenTomatoes.com`

Proceedings of COLING 2012: Technical Papers, pages 2821–2834,
COLING 2012, Mumbai, December 2012.

(Huang et al., 2007), `2ch.net` (Inoue et al., 2011)). We use text of this nature in the current work for obtaining content with which our conversational agent constructs its utterances.

In this paper, we propose the use of question-answer (QA) pairs from community-driven question-and-answer websites such as `AskKids.com` and `Answers.com` as content for our conversational agent. QA pairs from the Web are essentially individual pairings of questions and the corresponding answers contributed by online communities. By nature, the QA pairs extracted from a particular source on a certain topic are disjointed, in that they do not have any temporal or structural information that could immediately lend themselves to straightforwardly building conversations. We see the potential of QA pairs for use in a conversational agent as they are a reasonable embodiment of human-human communication, unlike text extracted from other sources such as Wikipedia or online news articles. The abundance and self-contained nature of QA pairs means that they can be extracted easily over a wide range of topics covered by the respective online communities. QA pairs have also been used in the past with some success for interactive question answering (IQA), driven solely by the user's information needs. We developed an IQA system (Wong et al., 2011) in the past that supports interactivity and does not require inputs to be formulated as questions. This system, however, still interprets inputs as questions and service them with answers. The approach in this paper, on the other hand, is able to share the initiative with the human user for determining conversation content and direction, and is able to engage with the user using a range of different *conversational strategies* such as 'question asking', 'fact telling' and 'question answering'. Context is maintained, similar to our previous work (Wong et al., 2011), but instead of using this purely to identify a question, which is then answered, it is now used to identify a QA pair of relevance for crafting system utterances to further a conversation.

The key contributions of this paper lie in the conversational strategies that we describe that specify how the different parts of the QA pairs can be used and combined with a small number of generic hand-crafted fragments to generate natural system utterances. These system contributions, when interplayed with cooperative user inputs, will produce seemingly coherent conversations. Unlike existing dialogue systems that rely on deep language processing to achieve 'understanding' from input (e.g., (Schulman et al., 2011)), our conversational agent relies only on shallow analysis for efficiency to extract and assign weights to terms and to identify domain information for building *conversational context*. To examine the level of coherence and naturalness that our system can achieve in the absence of standard dialogue management, custom language resources, and deep semantic analysis of user inputs, we perform a pilot study involving 11 human assessors. The results, which show that our approach does indeed produce natural and coherent interaction relative to the human assessment, are discussed in Section 5. Prior to that, we look at the state of the art in conversational agents in terms of their dialogue management approaches and language resource requirements in Section 2. In Section 3, we provide some background on the system architecture and the overall process, including the structure of our QA pairs, the processing of input and the building of context. We then describe in Section 4 the way in which we use the QA pairs according to several strategies, to form a variety of system utterances, using these to build natural, reasonably coherent conversations. After the system evaluation in Section 5, we conclude in Section 6 with discussion of future work to incorporate this into a more complete virtual companion, integrating these conversational agent abilities into a fully fledged companion with a range of capabilities.

2 Related Work

Mining content from the Web to support human-computer interaction applications (e.g., question answering (Radev et al., 2001; Clarke et al., 2001; Yao et al., 2012), interactive question answering (Wong et al., 2011), conversational systems (Huang et al., 2007; Wu et al., 2008; Shibata et al., 2009)) has been investigated since the advent of the read-write Web. In particular, the practice of mining the Web to alleviate the bottleneck of conversational content acquisition is increasing in popularity. Huang et al. (Huang et al., 2007), for example, proposed a supervised technique for extracting `<thread-title><reply>` pairs from online forums as 'knowledge' for chatbots. The authors used `RottenTomatoes.com` as their source of title-reply pairs. The extraction is done in a cascaded manner, given the semi-structured and relatively noisy nature of text in online forums. All replies relevant to the thread title are first extracted using an SVM classifier based on features related to the structure and content. The candidate pairs are then ranked according to the quality of the content, and the top *n* pairs are selected as chatbot 'knowledge'. However, no details were provided on how the pairs could be used in a chatbot. Yao et al. (Yao et al., 2012), on the other hand, describe a technique for extracting factoid QA pairs from standard text, such as Wikipedia[1]. In their approach, natural language processing techniques are used to create a QA pair from a declarative sentence. These QA pairs are then used by virtual characters to answer queries from a user. As such, they are only using the QA pairs reactively, whereas our novel proposal is to also use them proactively. Moreover, by mining our content from social media sites, we are more likely to obtain more conversational fragments, as opposed to the strictly factoid style QA pairs that Yao et al. are interested in. However, they do demonstrate encouraging performance for their techniques and it may be possible to use such an approach to further increase our content with such (factoid) QA pairs.

Overall, despite the rise in interest in deriving content from the Web for conversational systems, there is little research on how this content can be used more effectively to maximise the traits that should ideally be present in human-computer dialogue. These traits include the coherence of system-generated outputs with respect to preceding exchanges, the naturalness of system utterances (e.g., is the utterance too lengthy or formal for supposedly casual conversations), and the ability to support mixed-initiative interactions (i.e., no strict adherence to whether the system or the user leads the conversations). In the next section, we present our approach to a system which mines the Web for QA pairs, and specifies how and when this content can be used to generate coherent, natural system contributions to mixed-initiative conversation.

3 System Overview

An overview of the main components of our conversational system is shown in Figure 1. We briefly describe the collection of QA pairs, and the *Input Analysis*, *Context Management* and *QA pair Retrieval and Ranking* components in this section. We focus in more detail in Section 4 on the *Strategiser*, which is the major novel contribution of this paper.

3.1 QA Pair Collection

The content for generating system utterances is primarily a set of QA pairs, mined from community-driven question-and-answer websites. During output generation, the different parts of the selected QA pairs are augmented by a small number of strategy-specific fragments, which we describe together with the strategies, to create natural system utterances. As our focus

[1]They actually use the Simple English version of Wikipedia.

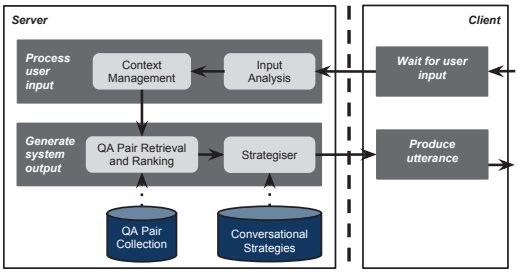


Figure 1: Conversational system architecture

is a companion for children, in this work we have used websites such as `AskKids.com` and `Answers.com` for collecting our QA pairs. This collection is constructed offline, using an interface similar to that described in (Wong et al., 2011) which allows the system administrator to specify the source (e.g., *"askkids*"), domain (e.g., *"animal*") and a seed word (e.g., *"lion*") to represent a topic in the domain for localising and indexing QA pairs. Each QA pair consists of the question and the answer to that question, and some metadata, which are the domain tag, the topic tag and the source tag. These QA pairs are then organised into a database according to the metadata. The tags are used to move between topics of the same domain during a conversation, and to define the scope (i.e., subset of fragments) that the conversational agent has to deal with at any one time. Currently, the collection contains a total of 23,295 QA pairs across 150 topics in the animal domain.

3.2 Input Analysis

Analysis of input utterances is relatively shallow, deliberately avoiding any attempt at sophisticated natural language understanding. The input is first tagged with parts of speech using FastTag[2] for its speed. Noun phrases are then identified using simple regular expression patterns. Pronouns are resolved to the most prominent entities from previous inputs, using a method loosely similar to the backward looking centering approach (Mitkov, 2001). The phrases and words are then assigned weights to reflect their content bearing property. These weighted phrases and words are referred to as terms. Our system has two different weighting schemes implemented, the deviation from Poisson approach (Church and Gale, 1995) and *tf-idf*, with the former being the default. The higher the weight of a term, the more content-bearing it is. Content-bearing terms play a more discriminatory role in selecting relevant QA pairs for utterance generation during the retrieval and ranking process. Stopwords, on the other hand, are removed by virtue of them being non-content bearing. In addition, a dictionary-based approach, operating over the input string, is used to detect if an input is an explicit question or a request for session termination.

3.3 Context Management

Context, which is essentially a collection of weighted terms decayed over time, is used to select QA pairs which are sufficiently relevant to the user inputs to generate system utterances. By

[2] markwatson.com/opensource

collecting the weighted terms from the user inputs (and potentially system output), and then decaying them over time, we maintain sufficient contextual information to make good choices of QA pairs for building system utterances. Each time a new input is processed, the corresponding weighted terms are combined into the existing conversational context. For example, if the current input is the $5th$ utterance by the user, then the context at that turn would contain all the terms and their weights extracted from the previous four inputs. The context management process at turn 5 would then assimilate the terms from the recent input into the current context to create a revised conversational context for turn 6.

The rule for combining new terms to the context is as follows. If a term is already present in the context, the weight associated with the recent occurrence is added to the term's existing decayed weight in the context to reinforce what we perceive as an important term. If the term is absent from the context, the new term and its weight are added to the context. In other words, recurrence is an important factor for a term to maintain its prominence in the context, where the more times a term occurs in the conversation, the more significant it will become through the compounding of its weights. During the process of combining, the weights of all other terms in the context that are not encountered in the most recent input are decayed based on the turn in the conversation, the decay factor and the part of speech. Those term used less recently thus have a greatly reduced weight. Different parts of speech are decayed at different rates, with nouns decaying less quickly than verbs, which decay slower than adjectives, adverbs and other parts of speech. Terms are decayed according to $v' = v \exp(-t\lambda\alpha)$ where v' is the decayed weight, v is the original weight, λ is the decay factor set to 0.8 during our experiments. The α value has been set to the following empirically-selected values to reflect the bias that we introduced according to a term's part of speech: 0.25 for nouns, 0.75 for adjectives and adverbs, 1.25 for verbs, and 2.50 for others.

3.4 QA Pair Retrieval and Ranking

QA pair retrieval and ranking are performed to determine the QA pair that has the highest relevance to the conversational context. Initially, a set of all candidate QA pairs containing at least one term from the context is retrieved. The relevance of the candidates with respect to the conversational context is then determined in terms of (1) the extent of the overlap of words in the question and answer parts of QA pairs with those in the conversational context, and (2) the string similarity between the question component and the user input. Edit distance (Levenshtein, 1966) is used to determine the similarity between the user input and the question part of the candidate QA pairs. As for overlap, the more highly weighted terms from the conversational context that appear in a QA pair, the higher the pair's score will be. The sum of the weights of terms from the context that appear in the question as well as the answer of a QA pair is determined. This sum is augmented by the location of appearance, where term matches in the questions are scored twice as high as matches in the answers. At the end of retrieval and ranking, the top scoring QA pair is selected for utterance generation.

4 Conversational Strategies

The core of our system's abilities are two *reactive* and four *proactive* strategies. These strategies attempt to recreate the types of human-to-human communications that are commonly encountered. The strategies is also divided into *progression* or *conclusion* depending on their roles, to either progress or conclude a conversation. The content used by these strategies to produce utterances are the QA pairs and a small number of specialised *hand crafted fragments*

which include speech disfluencies to produce more natural outputs (Marge et al., 2010). These fragments are assigned tags and are contained in the strategy library. This library can be extended to increase the variety of specialised fragments, independently of the actual strategies.

4.1 Reactive Strategies

This group of strategies is characterised by the agent following the user's lead. There are two strategies in this group, the first being a *reactive progression* strategy called UAQ, and a *reactive conclusion* strategy called UEC. The details are discussed below.

UAQ (User Asks Questions) Strategy: This strategy is initiated when the input parsing detects that the user has asked a question. In this case the keywords from the parsed input, weighted as described in *Context Management*, are used to select a QA pair where the context has an optimal match with the question portion of the QA pair. The answer portion of the pair (or if it is too long, the initial part) is then used as output. If it is desired to continue this strategy beyond a single interaction in the conversation, then subsequent inputs are essentially treated as refinements of the initial question. This is essentially the approach used in our IQA system (Wong et al., 2011). We do not use any specialised fragments to augment the selected QA pair in this strategy.

UEC (User Ends Conversation) Strategy: This strategy is used when input processing detects signs that the user wants to conclude the conversation. When a termination is detected, the Strategiser removes all pending outputs in the queue and adopts this strategy to access the associated fragments shown in Table 1, which are used to generate farewell messages.

1	OK. Nice chatting with you. Cya.	[UEC]
2	Bye.	[UEC]

Table 1: Some fragments for the UEC strategy

4.2 Proactive Strategies

We have five proactive strategies, two for progressing and three for concluding a conversation. The *proactive progression* strategies are the ones that we make most use of in our conversational agent. In these strategies the system takes the initiative by either asking a question or sharing some information. The three *proactive conclusion* strategies deal with certain circumstances that necessitate the conclusion of conversation sessions, namely, (1) lack of user participation, and (2) system's inability to carry on the conversation. These conclusion strategies are the result of the system interpreting various signs and initiating the ending. The conclusion strategies do not use any components of QA pairs for generating system utterances.

SSK (System Shares Knowledge) Strategy: This strategy attempts to simulate conversations involving the sharing of knowledge between two participants in a chat. In the SSK strategy, the system assumes the role of imparting knowledge, with the user on the receiving end. This strategy uses the answer component of QA pairs to formulate system utterances to create this effect. To illustrate, consider the following QA pair:

```
Q: What is a panda?
```

```
A: The Panda is the bear-like black and white mammals which lives in
China. These cute and cuddly like bears eats mainly bamboo shoots
and are becoming extinct. Some experts refer to them as bears while
others believe they are more like a raccoon.
```

The SSK strategy uses the first sentence in the answer and augments it with a selection of fragments such as *"Did you know that $X*" and *"I just learnt that $X*". In this example, the first system utterance generated using this strategy, which will be placed in a queue, is *"Did you know that the panda is the bear-like black and white mammals which lives in China.*". The subsequent sentences in the answer are stored in the queue for generation but their order of use will depend on the evolving conversation context in the next n iterations. For example, assume that the system sends the first utterance in the queue to the user, and the user provides *"They look like a racoon*" as the next input. Since this new input is not an explicit question and the queue of the next things to utter is not empty, the strategy remains the same, i.e., SSK, and the system utterance generation process does not invoke the QA pair retrieval and ranking process. The Strategiser is used to reorder the utterances in the queue based on their overlap with the current conversational context that contains the term *"racoon*". Since the overlap of the words in the second sentence in the queue (i.e., the third sentence in the answer of the selected QA pair) with the context is likely to be higher (due to the word *"racoon*"), this sentence will be selected, augmented with the appropriate fragments and moved to the front of the queue as the next system utterance. This next utterance will appear as *"And some experts refer to them as bears while others believe they are more like a raccoon*". In the third iteration, only one sentence from the 3-sentence answer remains. Unless the next user input is a question, the system will generate its next utterance as *"What's more is that these cute and cuddly like bears eats mainly bamboo shoots and are becoming extinct*". This process of reordering the sentences that the system utters takes place for all sentences in the answer of the selected QA pair.

1	Did you know that $\$X$?	[SSK-1]
2	I just learnt that $\$X$.	[SSK-1]
3	What's more, $\$X$.	[SSK-2]
4	What's more is that $\$X$.	[SSK-2]
5	And $\$X_3$... Well... $\$X$.	[SSK-2]
6	And $\$X_3$... Umm... $\$X$.	[SSK-2]

Table 2: Some fragments for the SSK strategy

Table 2 shows the list of fragments associated with the SSK strategy. Fragments labelled [SSK-1] are used by the strategy to prepend to the first sentence of the selected answer, while those labelled [SSK-2] are used for augmenting the subsequent sentences. Fragments 5 and 6 are three examples containing false starts, where $\$X_3$ and $\$X_4$ represents the first three and four words in $\$X$, respectively.

SAQ (System Asks Questions) Strategy: This strategy attempts to recreate a conversational scenario where a conversation participant explicitly asks the other participant a question. In this strategy, the system poses as the participant asking the question. Both the question as well as the answer components of QA pairs are used to generate the system utterances in this strategy.

1	Can you tell me, $\$X$?	[SAQ-1]
2	Tell me, $\$X$?	[SAQ-1]
3	Erm... $\$X$.	[SAQ-2]
4	$\$X_3$... Well... $\$X$.	[SAQ-2]

Table 3: Some fragments for the SAQ strategy

When this strategy is selected, the question component of the selected QA pair is used to construct the first system utterance. Fragments shown in Table 3 which are labelled [SAQ-1] are used to augment the question. The second utterance is then constructed using the first three sentences of the corresponding answer and a fragment tagged with [SAQ-2]. With regard to [SAQ-2] fragments, 3 contains fillers while 4 contains a false start to add to the 'natural' effect of the resulting utterances. To illustrate, consider the QA pair above. The first and second utterances in the queue would appear as *"Can you tell me, what is a panda?*" and *"The Panda is... Well... The Panda is the bear-like black and white mammals which lives in China. These cute and cuddly like bears eats mainly bamboo shoots and are becoming extinct. Some experts refer to them as bears while others believe they are more like a raccoon.*". The use of only the first three sentences in the answer to generate the second system utterance is motivated by the average sentences in answers of QA pairs from AskKids.com being around three.

UNI (User Not Interested) Strategy: This strategy is chosen by the Strategiser when input processing detects three empty input sentences in a row, likely to be the result of the expiration of the 60-second timer imposed by the input waiting process at the client side. The system assumes the three successive timeouts to be a sign of the user losing focus to other tasks, or lacking interest in pursuing the conversation further. Similar to the UEC strategy, the Strategiser cancels all pending utterances when this strategy is chosen, and uses one of the fragments lablled [UNI] shown in Table 4 to generate the system's final utterance.

1	You don't seem very interested in continuing the chat. Let's do this some other time. Bye!	[UNI]
2	It seems that you're in the middle of something. Let's talk some other time. Cya.	[UNI]

Table 4: Some fragments for the UNI strategy

LTT (Lost Train of Thought) and SEC (System Ends Conversation) Strategies: The second circumstance that necessitates the conclusion of a conversation, from the system's point of view, is the repeated inability of the retrieval and ranking modules to select a QA pair that the Strategiser can work on using the conversational context maintained by the system. There are two main causes of this: (1) the user's successive use of non-content bearing words in his or her inputs, and (2) the exhaustion of QA pairs. Two strategies are involved in this circumstance. Whenever the Strategiser does not receive a selected QA pair from the retrieval and ranking modules, the LTT strategy is used to generate utterances with fragments, shown in Table 5, designed to prompt the user to repeat the recent input(s). When this strategy is used, the conversational context is emptied and rebuilt using the latest input with the hope that the new context can be used to select some useful QA pairs. In the event that the LTT strategy is being selected for the third time in a row, the system will consider this as a sign that it is unable to further pursue the conversation. In such cases, the Strategiser uses the SEC strategy

to conclude the current session. Table 6 shows the fragments associated with the SEC strategy for ending a conversation.

1	Oops. I got distracted. You were saying?	[LTT]
2	Sorry, can you please repeat what you just said?	[LTT]
3	Where were we? I was talking to someone else.	[LTT]

Table 5: Some fragments for the LTT strategy

1	I've got something that I have to attend to now. Talk to you later.	[SEC]
2	Sorry. I'm tired now. Let's continue this some other time. Bye.	[SEC]
3	We'll need to stop here now. I've got to run. Bye.	[SEC]

Table 6: Some fragments for the SEC strategy

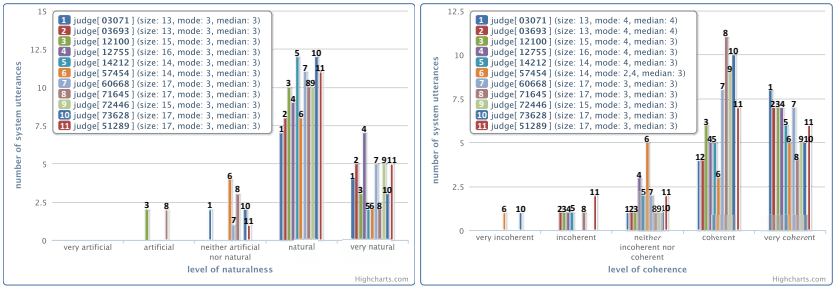
5 Evaluation

We conducted a pilot study to assess the naturalness and coherence of utterances generated by our conversational agent using only QA pairs and 41 generic fragments. We first discuss the experimental setup, before moving on to present the results of humans' judgements of our system outputs. Error analysis is also performed, where we analyse the causes of those system utterances that were perceived as less natural and coherent by the judges. An actual interaction between the system and one of the judges is presented at the end to illustrate a typical conversation with the conversational agent.

5.1 Experimental Setup

The domain that we have selected to test our conversational agent is that of *"animal*". The question-answer pairs are downloaded from two sources, namely, `AskKids.com` and `Answers.com`, solely for this experiment. 150 topics in the *"animal*" domain were manually identified and used as seed words, where 17,790 QA pairs were retrieved from `Answers.com` and 5,505 from `AskKids.com`. We were only able to download QA pairs about 96 out of the total 150 topics from `AskKids.com`. To ensure that the Strategiser has the option of using QA pairs from both sources during utterance generation, a topic is randomly selected by the system from amongst the 96 seed words to initiate every conversation with the participants.

For the experiment, we obtained the assistance of 11 human participants. For privacy reasons, the participants' names, genders and other personal information were not collected, and their names were never used by the conversational agent during conversations. The experiments were conducted over the Internet where the participants were provided with a text interface accessible via a Web browser. The interface is made up of two main parts, a user input field at the bottom, and a panel to display the exchanges between the user and the system. The instructions that we provide to the participants were as follows: (1) We requested the participants to limit their conversations to a 10-minute duration; and (2) At the end of every conversation, the participants were requested to rate the system utterances for naturalness and coherence using two 5-point Likert scales (the naturalness scale has scores that range from 0-`very artificial` to 4-`very natural`, while the coherence scale ranges from 0-`very incoherent` to 4-`very coherent`).



(a) Distribution of "naturalness"

(b) Distribution of "coherence"

Figure 2: Distribution of human judgements

5.2 Results

A total of 168 system utterances were recorded for all 11 conversations involving the 11 participants, with 16 − 17 system utterances on average per conversation.

Naturalness of system utterances: Figure 2(a) shows the distribution of the naturalness rating of system-generated outputs. Out of the 168 system utterances, only about 10% (17 utterances) are rated at or below level 2, with 13 system utterances being judged as `neither artificial nor natural` and 4 as `artificial`. The other 26% (43 utterances) and 64% (108 utterances) are judged as `very natural` and `natural`, respectively. In other words, the chart is left-skewed, with the majority (90%) of the judgements tending towards the `very natural` end of the scale, with both the mode and median for all 11 conversations being at level 3, which is `natural`.

Coherence of system utterances: Figure 2(b) summarises the human judgements of coherence of system-generated outputs. The trend appears to be quite similar to the judgements of naturalness in terms of skewness. However, instead of being concentrated at level 3, the judgements of coherence are slightly more spread over the neighbouring levels 2 and 4. The modes and medians shown in the legend in Figure 2(b) also demonstrate this spread. About 17% (29 utterances) of all system outputs were considered as `neither coherent nor incoherent` (20 utterances), `incoherent` (7 utterances) or `very incoherent` (2 utterances). At the same time, the numbers of utterances rated as `very coherent` (67 utterances) and as `coherent` (72 utterances) are approximately equal and when tallied up constitute 83% of all system-generated outputs.

Ability to maintain coherence: While the majority of the system-generated outputs are coherent (i.e., 83% of utterances are `coherent` or better), we are also interested in whether coherence of the conversations is sustained throughout, or whether coherence deteriorates with conversation length. Figure 3 shows coherence of individual system utterances over time for all 11 conversations. The figure shows that the coherence of the system contributions was consistently above level 2 (i.e., `neither incoherent nor coherent`), with 9 sporadic valleys reaching level 1 and 0.

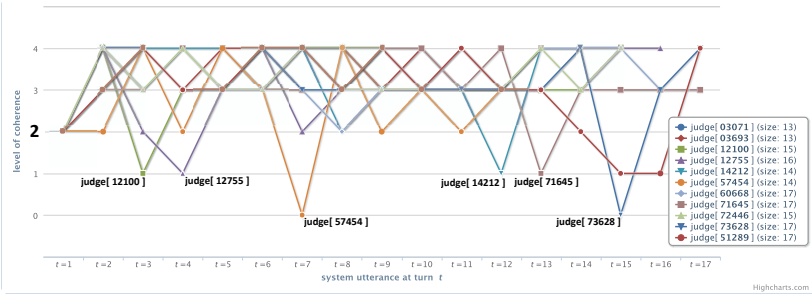


Figure 3: Level of coherence over the course of a conversation

causes	1	2	total
repetition of utterances	3	2	5
verbose utterances	0	4	4
perceived non-responsiveness to user questions	0	4	4
quality of QA pairs and wrongly judged utterances	1	3	4

Table 7: The causes behind the system utterances that were rated as `artificial` (score 1) or `neither artificial nor natural` (score 2) on the naturalness scale by the judges.

5.3 Error Analysis

After collecting judgements from the human participants, we analysed chat logs for naturalness and coherence ratings of the individual system utterances. We identified a number of causes behind the average and poorly-rated outputs: we discuss these here. Possible ways to remove these problems are discussed in Section 6.

Causes of artificial utterances: We identified four main causes behind the 17 system utterances that were judged as `artificial` or `neither artificial nor natural`, as summarised in Table 7. The first cause, i.e., repetition occurred when QA pairs which are different on the surface but semantically similar were used in succession to generate system utterances. For example, there were two distinct QA pairs with the questions *"What is cougar*" and *"What is the cougar?*" in our collection. During the system's conversation with judge 71645, these two pairs were used to generate the two different utterances *"Can you tell me, what is cougar?*" and *"Tell me, what is the cougar?*", four turns apart. From the judge's point of view, these utterances appeared as unnatural in that such blatant repetitions would not normally be encountered in human-human dialogue. The second cause, not surprisingly, is the artificiality of verbose utterances. The number of words in a typical system output ranges between 8 to 20. There are, however, a small number of utterances that go beyond this range, with some approaching or exceeding length 60. In this experiment, 8 utterances were considered to be lengthy, and of these, 4 were rated down by the participants (were rated as `neither artificial nor natural`) that encountered them during their interactions with the system. The third cause is the user perceiving the system as not responding to their questions. This problem occurred due to the system's more restrictive interpretation of what

causes	0	1	2	total
alternation between topics	0	4	2	6
utterances which are out-of-place	1	1	1	3
perceived non-responsiveness to user questions	1	0	3	4
issues with context management	0	2	3	5
initial system utterance	0	0	11	11

Table 8: The causes behind the system utterances that were rated as `very incoherent` (score 0), `incoherent` (score 1) or `neither incoherent nor coherent` (score 2) on the coherence scale by the judges.

constitutes a question. Out of the 4 utterances in the `quality of QA pairs and wrongly judged utterances` row of Table 7, 2 were the result of poorly-written QA pairs by the online human contributors. As an example, one of the sentences in the answer of a selected QA pair was incomplete, producing the system utterance *"And the interesting bit is that the giant panda lives in Asia and is most commonly known [sic]*", which was rated as 1 for naturalness by judge 12100. As for the other two utterances rated as `neither artificial nor natural` in the same row, the causes were unclear.

Causes of incoherent utterances: Next, we discuss the causes that contributed to some of the system utterances being judged poorly on the coherence scale. Table 8 summarises these causes. The first cause is the generation of utterances that give the users the impression of abrupt transitions. This may happen when the system moves to a new topic that is semantically less related (hence the abruptness), or alternates between topics due to the presence of multiple influential terms within the same domain in the context. The second cause is the generation of utterances that appear out of context. This may be due to the ways the QA pairs were written (poorly) or structured (i.e., the sentences in the answers may not lend well to the generation of individual, successive system utterances). Consider the following QA pair as an example:

```
Q: What do american cockroaches eat?
A: Just about everything! American cockroaches eat plants, garbage,
   people food, glue found in book bindings and other glues and
   pet foods are just a few.
```

This particular QA pair was used as part of the SSK strategy to produce the utterance *"Do you know that just about everything?*" when interacting with judge 03693. The utterance, which appeared in the following context, was in no way related to the previous exchanges and was ill-formed. For this reason, the judge rated its coherence as 1.

```
SYSM: erm... some types of cockroaches can fly but not all of them.
      the american cockroach can fly but the german one cannot.
      these are the most common cockroaches found.
USER: american cockroach?
SYSM: do you know that just about everything?
...
```

The third cause is the system's inability to detect and handle questions that were not formulated using typical *wh*-words. This cause also negatively affects the naturalness of system utterances,

as previously discussed. The fourth cause is due to problems with context management. As we have discussed before, the LTT strategy is used to recover from inadequate context for QA pair selection. This can happen when users repeatedly provide inputs that do not contain words that are useful for selecting QA pairs. Unfortunately, some of the judges do perceive this system-initiated solicitation for more useful inputs as unnatural. As for the 11 utterances judged as `neither incoherent nor coherent` on the fifth row of Table 8, these outputs were simply the first utterances generated by the system in their respective conversations. Since these utterances do not have any preceding inputs or outputs to be benchmarked against for coherence, these level 2 ratings were expected.

6 Discussion

Conversational systems that are required to engage users in conversations that cover the breadth and depth of certain topics face the bottleneck of custom- or hand-crafting the necessary content. Mining conversational content from the Web is increasingly being seen as a promising solution to this problem. However, the verbose, non-colloquial and monologue nature of typical Web text means that such content is not straightforwardly usable for responding to users by conversational systems for various reasons. In this work, we propose the use of question-answer (QA) pairs mined from community-driven question-and-answer websites as content for a conversational system. The main contribution of this paper lies in the conversational strategies we have defined that specify how and when the different parts of QA pairs can be used and augmented with a small number of generic fragments to generate natural system utterances. The coherence of system contributions in a conversation is managed using context. To assess the naturalness and coherence of system-generated utterances, we conducted a small pilot study involving 11 human participants. Out of the 168 system outputs, over 80% were judged as natural and coherent by the participants. The coherence of system contributions is generally maintained throughout the course of all 11 conversations, with sporadic incoherences. We also analysed the causes behind the $10-20$% of artificial and incoherent system outputs.

Limitations and potential improvements on the naturalness of system utterances: The two main causes that contributed to a number of outputs being judged as artificial were the appearance of repetitive and lengthy utterances. The problem of repetition arises when QA pairs with different surface patterns that are semantically similar are selected within the same conversation to generate system utterances. One possible way to overcome this is to use a string similarity measure to compare the questions of potential QA pairs against the QA pairs that have already been used. Candidate pairs that are similar to the previously used ones on the surface can then be excluded from future use. As for coping with lengthy utterances, a range of approaches are available, from detecting sentence boundaries with some heuristics for recombining them, to more elaborate summarisation techniques.

Limitations and potential improvements on the coherence of system utterances: Incoherent utterances are caused mainly by the lack of management of the implicit transitions between topics in the same domain during a conversation. The problem of topic transitioning can be managed, for example, using local *wellformedness* constraints, e.g., semantic relatedness measures. The other causes of poor coherence and artificial outputs, i.e., the non-responsiveness to certain questions (that have surface patterns not recognised by the system) and the quality of QA pairs, are more difficult to address. Adding new patterns to support more types of questions or using punctuation to detect questions may seem to be straightforward solutions. However,

our intention to potentially incorporate speech input will render the latter option useless, while the former solution will increase false positives during question detection. In regard to the quality of QA pairs, manual intervention is necessary to proof-read and edit the pairs. Currently, no manual effort is required to make our system operational, except providing seed terms to populate the QA pair collection.

Acknowledgment

This work is partially supported by the Australian Research Council and Real Thing Entertainment Pty. Ltd. under Linkage grant number LP110100050.

References

Bickmore, T., Pfeifer, L., and Jack, B. (2009). Taking the time to care: Empowering low health literacy hospital patients with virtual nurse agents. In *Proceedings of the 27th CHI*, Boston, USA.

Chen, Z., Liao, C., Chien, T., and Chan, T. (2011). Animal companions: Fostering children's effort-making by nurturing virtual pets. *British Journal of Educational Technology*, 42(1):166–180.

Church, K. and Gale, W. (1995). Inverse document frequency (idf): A measure of deviations from poisson. In *Proceedings of the ACL 3rd Workshop on Very Large Corpora*, pages 121–130.

Clarke, C., Cormack, G., and Lynam, T. (2001). Exploiting redundancy in question answering. In *Proceedings of the 24th International Conference on Research and Development in Information Retrieval*.

Huang, J., Zhou, M., and Yang, D. (2007). Extracting chatbot knowledge from online discussion forums. In *Proceedings of the 20th IJCAI*, Hyderabad.

Inoue, M., Matsuda, T., and Yokoyama, S. (2011). Web resource selection for dialogue system generating natural responses. In *Proceedings of the 14th HCI*, pages 571–575, Orlando, Florida, USA.

Levenshtein, V. (1966). Binary codes capable of correcting deletions, insertions, and reversals. *Soviet Physics Doklady*, 10(8):707–710.

Marge, M., Miranda, J., Black, A., and Rudnicky, A. (2010). Towards improving the naturalness of social conversations with dialogue systems. In *Proceedings of the 11th SIGDIAL*, University of Tokyo, Japan.

Mitkov, R. (2001). Outstanding issues in anaphora resolution. In *Computational Linguistics and Intelligent Text Processing*, pages 110–125. Springer.

Radev, D., Qi, H., Zheng, Z., Zhang, Z., Fan, W., Blair-Goldensohn, S., and Prager, J. (2001). Mining the web for answers to natural language questions. In *Proceedings of the 10th CIKM*.

Schulman, D., Bickmore, T., and Sidner, C. (2011). An intelligent conversational agent for promoting long-term health behavior change using motivational interviewing. In *Proceedings of the AAAI Spring Symposium*, Stanford University.

Shibata, M., Nishiguchi, T., and Tomiura, Y. (2009). Dialog system for open-ended conversation using web documents. *Informatica*, 33(3):277–284.

Wallace, R. (2009). The anatomy of a.l.i.c.e. In Epstein, R., Roberts, G., and Beber, G., editors, *Parsing the Turing Test*. Springer.

Wong, W., Thangarajah, J., and Padgham, L. (2011). Health conversational system based on contextual matching of community-driven question-answer pairs. In *Proceedings of the 20th ACM International Conference on Information and Knowledge Management (CIKM)*, pages 2577–2580, Glasgow, UK.

Wu, Y., Wang, G., Li, W., and Li, Z. (2008). Automatic chatbot knowledge acquisition from online forum via rough set and ensemble learning. In *Proceedings of the IFIP International Conference on Network and Parallel Computing*, Shanghai, China.

Yao, X., Tosch, E., Chen, G., Nouri, E., Artstein, R., Leuski, A., Sagae, K., and Traum, D. (2012). Creating conversational characters using question generation tools. *Dialogue and Discourse*, 3(2):125–146.

Factored Language Model based on Recurrent Neural Network

Youzheng Wu Xugang Lu Hitoshi Yamamoto Shigeki Matsuda
Chiori Hori Hideki Kashioka
National Institute of Information and Communications Technology (NiCT)
3-5 Hikari-dai, Seika-cho, Soraku-gun, Kyoto, Japan, 619-0289
{youzheng.wu,xugang.lu,hitoshi.yamamoto,shigeki.matsuda}@nict.go.jp
{chiori.hori,hideki.kashioka}@nict.go.jp

Abstract
Among various neural network language models (NNLMs), recurrent neural network-based language models (RNNLMs) are very competitive in many cases. Most current RNNLMs only use one single feature stream, i.e., surface words. However, previous studies proved that language models with additional linguistic information achieve better performance. In this study, we extend RNNLM by explicitly integrating additional linguistic information, including morphological, syntactic, or semantic factors. Our proposed RNNLM is called a factored RNNLM that is expected to enhance RNNLMs. A number of experiments are carried out that show the factored RNNLM improves the performance for all considered tasks: consistent perplexity and word error rate (WER) reductions. In the Penn Treebank corpus, the relative improvements over n-gram LM and RNNLM are 29.0% and 13.0%, respectively. In the IWSLT-2011 TED ASR test set, absolute WER reductions over RNNLM and n-gram LM reach 0.63 and 0.73 points.

Title and Abstract in another language (Chinese)

基于递归神经网络的因素语言模型

在多种神经网络语言模型（NNLM）中，递归神经网络语言模型（RNNLM）在很多场合都表现出优异的性能。目前，大多数的RNNLM只使用词汇这一种特征。然而，以往的研究表明利用了语言学特征的语言模型可进一步提高模型性能。本文通过引入语言学特征，包括形态，语法，或语义特征，对RNNLM进行扩展。我们称之为基于递归神经网络的因素语言模型（fRNNLM）。一系列的实验表明：本文提出的fRNNLM在多个任务中均显著地提高了RNNLM的性能：即一致地降低了困惑度和单词错误率（WER）。在宾州树库上，fRNNLM的困惑度相对相对于n元语言模型和RNNLM分别降低了29.0%和13.0%。在IWSLT-2011 TED语音识别测试集上，fRNNLM的单词错误率分别提高了0.63（相对于n元语言模型）和0.73（相对于RNNLM）。

Keywords: Factored Language Model, Recurrent Neural Network, Large Vocabulary Continuous Speech Recognition.

Keywords in Chinese: 因素语言模型, 递归神经网络, 大词汇量语音识别.

Proceedings of COLING 2012: Technical Papers, pages 2835–2850,
COLING 2012, Mumbai, December 2012.

1 Introduction

Language models (LM) are a critical component of many application systems such as automatic speech recognition (ASR), machine translation (MT) and optical character recognition (OCR). In the past, statistical back-off n-gram language models with sophisticated smoothing techniques have gained great popularity because of their simplicity and good performance. In n-gram language models, words are represented in a discrete space: the vocabulary. Standard back-off n-gram language models predict the following word based on the previous n-1 words, which can be expressed as,

$$p(w_i|w_{i-1},...,w_1) \approx p(w_i|w_{i-n+1},...,w_{i-2},w_{i-1}) \quad (1)$$

Even though n is usually limited to three or four, the number of parameters in a back-off n-gram LM is still enormous. Assuming the vocabulary size is $64K$, a 4-gram language model needs to estimate $64K^2$ bigrams, $64K^3$ trigrams and $64K^4$ 4-grams. Due to data sparseness, many are not observed during the training phase. This means that n-gram LMs have poor generalization to low-frequency and unseen n-grams. This problem becomes more severe as the vocabulary size increases. Many interesting approaches have been proposed to overcome it in large vocabulary continuous speech recognition (LVCSR) and statistical machine translation systems, especially smoothing techniques (Chen and Goodman, 1996), class n-gram language models (Brown et al., 1992), topic language models (Gildea and Hofmann, 1999; Hsu and Glass, 2006), structured language models (Chelba and Jelinek, 2000), maximum entropy language models (Rosenfeld, 1996) and random forests language models (Xu and Jelinek, 2004).

Among these techniques, one of the most successful schemes is the neural network language model (NNLM), such as the feed-forward NNLM (Bengio et al., 2003; Schwenk, 2007; Kuo et al., 2012), the recurrent NNLM (RNNLM) (Mikolov et al., 2010, 2011b) and the deep NNLM (Arisoy et al., 2012). Compared to other LMs, recurrent NNLMs, which are state-of-the art (Mikolov et al., 2011a; Arisoy et al., 2012), embed words in a continuous space in which probability estimation is performed using artificial neural networks consisting of input layer, single or multiple hidden layers, and output layer. Due to consistent improvement in terms of perplexity and word error rate and their inherently strong generalization, they have become an increasingly popular choice for LVCSR and statistical MT tasks.

Many of these RNNLMs only use one single feature stream, i.e., surface words, which are limited to generalize over words without using linguistic information, including morphological, syntactic, or semantic. In surface word RNNLMs, such words as “prices" and “price" and “developed" and “developing" are completely independent training instances. In this paper, we integrate additional linguistic information into a RNNLM, called a factored RNNLM, which can further improve the generalization of RNNLM using multiple factors (or features) of words (stems, lemmas, parts-of-speech, etc.) instead of surface forms of words as input to recurrent neural networks. Let us use an example to illustrate the shortcomings of surface word RNNLM. In extreme cases, the training data might only contain the following sentence: “difference between developed countries and developing countries". During training in the RNNLM that treats each word as a token in itself, the bi-gram “developing countries" is a completely unseen instance. However, for our factored RNNLM that incorporates stem features, “developing countries" belongs to seen instances in a sense because it shares the same stem bi-gram “develop countri" with the previous bi-gram “developed countries." This coincides with our intuition; “developed" and “developing" should add knowledge to each other during training. Our factored RNNLM may be more effective for such morphologically rich

languages as Czech, Arabic, or Russian. In this paper, we only evaluate it on English, and our experiments show that it significantly enhances performance measured in perplexity and WER.

To the best of our knowledge, few studies have been done on this perspective. Our approach provides the following advantages:

- It predicts the following word based on the entire history (due to a recurrent connection between input and hidden layers) in the low-dimensional representation (due to the neural network architecture).
- It integrates the additional rich information of words in particular morphological and syntactic features to overcome the data sparseness problem caused by limited in-domain training data, such as in academic lecture ASR and MT tasks.
- It simultaneously interpolates all possible factors and the entire history in stead of backing-off to fewer factors and shorter context, which can address the optimization problem well in factored n-gram language models.
- Since it converges faster than RNNLM due to the integration of additional features, it can save several days of training if the training data are large.

This paper is organized as follows: We introduce related studies in Section 2. In Section 3, we describe our proposed factored RNNLM in detail. Section 4 shows the performance of our model as measured by both perplexity and WER. We finally summarize our findings and outline future plans in Section 5.

2 Related work

Recently, deep neural networks are experiencing significant improvements in the fields of image processing, acoustic modeling (Seide et al., 2011), language modeling, etc. Neural network language models to LVCSR were first presented in (Bengio et al., 2003), which was a feed-forward NNLM with a fixed-length context consisting of projection, input, hidden, and output layers. Arisoy et al. (2012) proposed a deep NNLM that uses multiple hidden layers instead of single hidden layer in feed-forward NNLMs. Furthermore, several speedup techniques such as shortlists, regrouping and block models have been proposed (Schwenk, 2007). Feed-forward NNLMs, which predict following word w_i based on any possible context of length n-1 history, remain a kind of n-gram language model.

Recurrent NNLM (RNNLM) (Mikolov et al., 2010, 2011b), which has different architecture at the input and output layers, can be considered as a deep neural network LMs because of its recurrent connections between input and hidden layers, which enable RNNLMs to use their entire history. Compared with feed-forward NNLMs, recurrent NNLMs reduce computational complexity and have relatively fast training due to the factorization of the output layer. Other experiments (Mikolov et al., 2011a; Arisoy et al., 2012; Kuo et al., 2012) demonstrated that RNNLM significantly outperforms feed-forward NNLM. Therefore, this paper uses RNNLM as a baseline and improves it by incorporating additional information other than surface words, such as morphological or syntactic features.

Although few studies incorporate morphological and syntactic features into RNNLM, using multiple features in language modeling is not novel. For example, Duh and Kirchhoff (2004) presented

a factored back-off n-gram LM (FLM) that assumes each word is equivalent to a fixed number (K) of factors, i.e., $W \equiv f^{1:K}$, and produces a statistic model of the following form: $p(f_i^{1:K}|f_{i-n+1:i-1}^{1:K})$. The standard back-off in an n-gram LM first drops the most distant word (w_{i-n+1} in the case of Eq. (1)), and then the second most distant word etc. until the unigram is reached. However, the factors in FLM occur simultaneously, i.e., without forming a temporal sequence, so the order in which they should be dropped is not immediately obvious. In this case, FLM creates a large space of back-off graphs that cannot be exhaustively searched. Duh and Kirchhoff (2004) employed a genetic algorithm (GA) that, however, provides no guarantee of finding the optimal back-off graph. Our factored RNNLM addresses this optimization problem well, as described in Section 3. In addition, Emami and Jelinek (2004); Alexandrescu and Kirchhoff (2006); Kuo et al. (2009); Collins et al. (2005) introduced various syntactic features into their feed-forward NNLMs and discriminative language models. Table 1 summarizes FLM, RNNLM, and our approach from three points of view.

	Conditioning variables	History	Pros and Cons
FLM	Word and its linguistic features	n-1 preceding history	Better than n-gram LM due to linguistic features; Creating a large space of models that cannot be searched exhaustively.
RNNLM	Word	Entire history	Further enhancing FLM due to RNN architecture; Conditioning variables are only words, no morphological or syntactic linguistic features are used.
factored RNNLM	Word and its linguistic features	Entire history	Combining the above merits, but more parameters and computation complexity, which actually does not cause problems, as described in Section 4.4.

Table 1: Comparison of FLM, RNNLM, and factored RNNLM

Koehn and Hoang (2007) introduced various features from linguistic tools or word classes into phrase-based MT models for better translation performance.

3 Factored RNNLM

The architecture of our factored RNNLM is illustrated in Fig. 1. It consists of input layer x, hidden layer s (state layer), and output layer y. The connection weights among layers are denoted by matrixes U and W. Unlike RNNLM, which predicts probability $P(w_i|w_{i-1}, s_{i-1})$, our factored RNNLM predicts probability $P(w_i|F(w_{i-1}), s_{i-1})$ of generating following word w_i and is explicitly conditioned on a collection or bundle of K factors of one preceding word. It is implicitly conditioned on the factors of the entire history by the delay copy of hidden layer s_{i-1}. Here, $F(w_{i-1})$ is the vector concatenated from K factor vectors f_{i-1}^k ($k = 1, ..., K$), f_{i-1}^k stands for the k-th factor vector encoded from the k-th factor of preceding word w_{i-1}, and the functions of factor extraction $f^k(\cdot)$ are used to extract the corresponding factors. A word's factors can be anything, including the word itself, its morphological class, its root, and any other linguistic features. An example is shown in Table 2.

In the input layer, the extracted factors are encoded into the factor vectors using the 1-of-n coding.

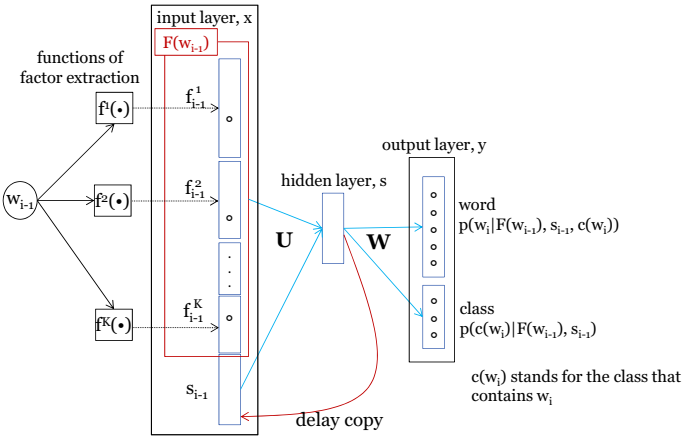


Figure 1: Architecture of factored recurrent NNLM.

Assume, for example, that the factor extracted by function $f^k(w_{i-1})$ is the m-th element in the k-th factor vocabulary, which is then encoded to $|f^k|$-dimension vector f^k_{i-1} by setting the m-th element of the vector to 1 and all the other elements to 0. Here, $|f^k|$ stands for the size of the k-th factor vocabulary. The K factor vectors are concatenated into $F(w_{i-1})$ as expressed in Eq. (2). Finally, the input layer is formed by concatenating factor vectors $F(w_{i-1})$ of the preceding word w_{i-1} and hidden layer s_{i-1} at the preceding time step, as shown in Eq. (3).

Word:	difference	between	developed	countries	and	developing	countries
Lemma:	difference	between	developed	country	and	developing	country
Stem:	differ	between	develop	countri	and	develop	countri
Part-of-speech[1]:	NN	IN	JJ	NNS	CC	VBG	NNS

Table 2: An example of factor sequences.

$$F(w_{i-1}) = [f^1_{i-1},\ f^2_{i-1},\ ...,\ f^K_{i-1}] \tag{2}$$

$$x_i = [F(w_{i-1}),\ s_{i-1}] \tag{3}$$

Using the concatenation vector, our proposed factored RNNLM can simultaneously integrate all factors and the entire history in stead of backing-off to fewer factors and a shorter context. The weight of each factor is represented in connection weight matrix U. Therefore, it can address the optimization problem well in factored n-gram LM (Duh and Kirchhoff, 2004). In the special case that f^1_{i-1} is a surface word factor vector and f^k_{i-1} $(k = 2, ..., K)$ are dropped, the factored RNNLM goes back to the RNNLM.

[1] http://www.cis.upenn.edu/~treebank/

The hidden layer employs a sigmoid activation function:

$$s_i^m = f(\sum_j (x_i^j \times u_{mj})) \quad \forall m \in [1, H]$$
$$f(z) = \frac{1}{1+e^{-z}} \tag{4}$$

where H is the number of hidden neurons in the hidden layer and u_{mj} is an element in matrix U denoting the corresponding connection weight.

Like (Goodman, 2001; Mikolov et al., 2011b), we assume that each word belongs to exactly one class and divide the output layer into two parts: the first estimates the posterior probability distribution over all classes,

$$y_c^l = g(\sum_j (s_i^j \times w_{lj})) \quad \forall l \in [1, C] \tag{5}$$

where C is the number of predefined classes. The second computes the posterior probability distribution over the words that belong to class $c(w_i)$, the one that contains predicted word w_i:

$$y_w^o = g(\sum_j (s_i^j \times w_{oj})) \quad \forall o \in [1, nc(w_i)] \tag{6}$$

where $nc(w_i)$ is the number of words belonging to class $c(w_i)$ and w_{lj} and w_{oj} are the corresponding connection weights.

To ensure that all outputs are between 0 and 1, and their sum equals to 1, the output layer employs a softmax activation function shown below:

$$g(z_d) = \frac{e^{z_d}}{\sum_x e^{z_x}} \tag{7}$$

Finally, probability $P(w_i|F(w_{i-1}), s_{i-1})$ is the product of two posterior probability distributions:

$$\begin{aligned} P(w_i|F(w_{i-1}), s_{i-1}) &= P(c(w_i)|F(w_{i-1}), s_{i-1}) \times P(w_i|F(w_{i-1}), s_{i-1}, c(w_i)) \\ &= y_c^l|_{l=classid(c(w_i))} \times y_w^o|_{o=wordid(w_i)} \end{aligned} \tag{8}$$

The architecture of splitting the output layer into two parts can greatly speedup the training and the test processes of RNNLM without sacrificing much performance. Many word clustering techniques can be employed. In this paper, we map words into classes with frequency binning (Mikolov et al., 2011b), which proportionally assigns words to classes based on their frequencies. The pseudo codes are shown in Fig. 2.

3.1 Training

To use the factored RNNLM, connection weight matrixes U and W must be learned. To learn them, training is performed with the back-propagation through time (BPTT) algorithm (Boden, 2002) by minimizing an error function defined in Eq. (9).

$$L = \frac{1}{2} \times \sum_{i=1}^{N} (t_i - p_i)^2 + \gamma \times (\sum_{lk} u_{lk}^2 + \sum_{tl} w_{tl}^2) \tag{9}$$

```
#vocab[i].cn denotes the number of the i-th word that occurs
#vocab[i].classid denotes the class index of the i-th word
#nclass is the number of classes predefined
 double df=0, a=0, b=0;
 for (i=0; i<|V|; i++) b+=vocab[i].cn;
 for (i=0; i<|V|; i++) {
    df+=vocab[i].cn/b;
    if (df>1) df=1;
    if (df>(a+1)/nclass) {
       vocab[i].classid=a;
       if (a<nclass-1) a++;
    }
    else {
       vocab[i].classid=a;
    }
 }
```

Figure 2: Frequency binning. $|V|$ is the word vocabulary's size.

where N is the number of training instances, t_i denotes the desired output; i.e., the probability should be 1.0 for the predicted word in the training sentence and 0.0 for all others. The first part of this equation is the summed squared error between the output and the desired probability distributions, and the second part is a regularization term that prevents RNNLM from over-fitting the training data. γ is the regularization term's weight, which is determined experimentally using a validation set.

The training algorithm randomly initializes the matrixes and updates them with Eq. (10) over all the training instances in several iterations. In Eq. (10), ψ stands for one of the connection weights in the neural network and η is the learning rate. After each iteration, it uses validation data for stopping and controlling the learning rate. Usually, the factored RNNLM needs 10 to 20 iterations.

$$\psi^{new} = \psi^{previous} - \eta \times \frac{\partial L}{\partial \psi} \tag{10}$$

3.2 Free parameter & time complexity

To better understand the differences between RNNLM and our factored RNNLM, we compare them in terms of the number of free parameters and computational complexity of one training step in Table 3. τ is the amount of steps used in BPTT.

	Free Parameter	Computational Complexity
RNNLM (1)	$(\|V\|+H)\times H + H\times(C+\|V\|)$	$(1+H)\times H\times\tau + H\times\|V\|$
fRNNLM (2)	$(\|f^1\|+...+\|f^K\|+H)\times H + H\times(C+\|V\|)$	$(K+H)\times H\times\tau + H\times\|V\|$
Difference (2)-(1)	$(\|f^1\|+...+\|f^K\|-\|V\|)\times H$	$(K-1)\times H\times\tau$

Table 3: RNNLM vs. factored RNNLM (fRNNLM).

From this table we can observe that the factored RNNLM has more free parameters and larger computational complexity. If the factored RNNLM only employs word factor (f^1) and POS factors

(f^2), then, it has $39 \times H$ additional free parameters. The additional computational complexity is $(K-1) \times H \times \tau$. In experiments, H is usually set to $300-1000$, τ is usually set to 4, $|V|$ is usually set to several hundreds of thousands. This means that $H \times |V| \gg (K-1) \times H \times \tau$, and the increased complexity can be neglected. Owing to the additional free parameters, our factored RNNLM converges faster and reduces training time. Section 4.4 shows the exact running time spent on experiments.

4 Experiments

In this section, we show the performance of our factored RNNLM as measured by perplexity. After analyzing these results, we present the performance measured by word error rate when the factored RNNLM is used in a LVCSR system. In our experiments, we mainly compare our factored RNNLMs with a 4-gram LM with modified Kneser-Ney smoothing (Chen and Goodman, 1996) and RNNLM (Mikolov et al., 2011b). In the factored RNNLM, we investigate four commonly used types of factors: word, stem[2], lemma[3] and part-of-speech (POS).

For perplexity results, we use the WSJ portion of Penn Treebank (LDC99T42). The WSJ portion is divided into training (sections 00-20), heldout (sections 21-22), and test (sections 23- 24) sets containing 930K, 101K, and 97K words respectively. The vocabulary is limited to 10K words. This setting is the same as that used by other studies (Xu and Jelinek, 2004; Mikolov et al., 2011b). The sizes of the factor vocabularies in the training set are shown in Table 4[4]. Note that the word vocabulary (10001 in Table 4) contains 10K words and one special token "<unk>" denoting words not in the vocabulary.

Factors	Word	Lemma	Stem	POS
Sizes	10001	7356	6892	37

Table 4: Statistics of factor vocabularies.

4.1 Impacts of factors

This experiment analyzes the contribution from each factor to the factored RNNLM in terms of the perplexities on the heldout and test sets. We set the number of hidden neurons in the hidden layer and the number of classes in the output layer for both the RNNLM and factored RNNLM to 320 and 300. Table 5 shows the experimental results. fRNNLM$_{wslp}$ denotes the factored RNNLM incorporating the word, stem, lemma, and POS factors, and so forth, the ratio is computed using $\frac{|U|_{factored\ RNNLM} - |U|_{RNNLM}}{|U|_{RNNLM}}$ that indicates the percentage of additional parameters in matrix U against the RNNLM. Subscript numbers are the relative improvements over RNNLM.

From this table, we observe the following: (1) All of the factored RNNLMs significantly improve their performances. For example, the improvement of fRNNLM$_{wsp}$ against the RNNLM on the test set reaches 14.4%. (2) No significant differences are found among the factored RNNLMs with various combinations of factors. The contributions from stem and lemma factors are less than 1.0%. In particular, it is not necessary to use both stem and lemma because they are very similar and obviously do not complement each other. (3) Although the size of the parts-of-speech is the

[2] `http://tartarus.org/~martin/PorterStemmer/`

[3] `http://lemmatizer.org/turglem-english-description`

[4] We directly use manually tagged parts-of-speech in the Penn Treebank corpus. Section 4.6 investigates automatically tagged parts-of-speech.

	Ratio	Heldout	Test
4-gramLM		156.26	156.41
RNNLM	-	146.94	145.63
fRNNLM$_{wp}$	0.4%	$128.14_{12.8\%}$	$126.47_{13.1\%}$
fRNNLM$_{wsp}$	67.5%	$127.09_{13.4\%}$	$124.63_{14.4\%}$
fRNNLM$_{wlp}$	72.0%	$126.81_{13.7\%}$	$124.76_{14.3\%}$
fRNNLM$_{wslp}$	138.8%	$126.06_{14.2\%}$	$124.76_{14.3\%}$

Table 5: Impacts of factors measured by perplexities.

smallest (only 37, Table 4), they have the largest impact on our factored RNNLM. The main reason may lie in that syntactic factor (POS) has stronger complementariness to the surface word factor, while morphological factors (stem and lemma) are too similar to the word itself, limiting such complementariness. Therefore, in the following experiments we only use word, stem, and POS in our factored RNNLM.

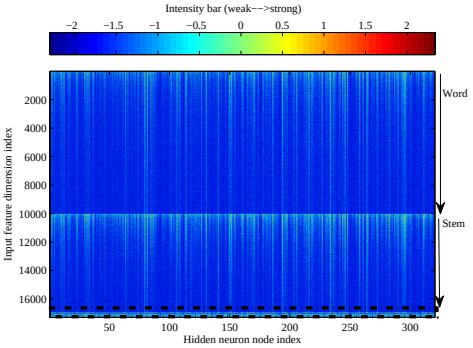
For a better understanding of the contribution of each factor to the factored RNNLM, we do a quantitative analysis of the connection weight values. The basic assumption in this analysis is that if one feature has a strong correlation or contribution to the factored RNNLM, the connections between the input features to the hidden neurons have large values (either positive or negative corresponding to positive or negative correlations). We show connection weight matrix U (corresponding to the logs of the absolute values of neural connection weights) in Figs. 3 (a) and (b). The horizontal and vertical axis denote the hidden neurons and the input feature dimensions. Since feature streams (word, stem, POS and history) are organized in blocks in matrix U, we mark each feature stream in blocks on the right vertical axis. In these figures, the connection intensity is marked by color, the brighter the color, the stronger the connection. From these figures, we can see that the POS feature stream shows the strongest connection intensity among all feature streams. The POS feature stream contributes the most to the factored RNNLM. However, RNNLM (Mikolov et al., 2011b) does not use it. In addition, the feature stream of the history also shows relatively strong intensity that confirms that the entire history is important.

4.2 Hidden neurons

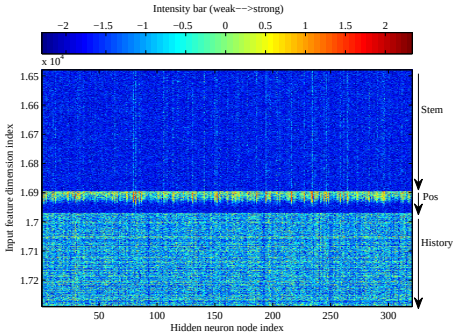
In this subsection, we evaluate the impacts from various numbers of hidden neurons in the hidden layer. Table 6 shows the results of the heldout set and the relative gains over the RNNLM. The experiments prove that factored RNNLMs consistently reduce perplexity. With increasing hidden neurons, both RNNLM and fRNNLM$_{wsp}$ enhance performance. The biggest improvement over RNNLM is 13.4%. The convergence column denotes the difference of the fRNNLM and RNNLM iterations, showing that factored RNNLM converges using fewer iterations. For example, RNNLM converges after 15 iterations, while fRNNLM$_{wsp}$ takes 12 iterations.

4.3 Convergence study

Figure 4 demonstrates the training progress of RNNLM and fRNNLM$_{wsp}$. In the same way, the number of hidden neurons in the hidden layer and the number of classes are set to 320 and 300, respectively. From this figure, we can observe that fRNNLM$_{wsp}$ significantly outperforms RNNLM at all iterations, especially at iterations 1-4 where the improvements exceed 20.0% and iterations



(a) Connection weight intensities of word, stem, POS and history



(b) Locally amplified view in rectangle marked position.

Figure 3: Neural connection weight intensity: between input feature and hidden neural nodes.

#Hidden neurons	RNNLM	fRNNLM$_{wsp}$	Gain	Convergence
60	163.71	147.00	10.2%	-3
120	152.33	133.07	12.6%	-2
240	147.74	128.75	12.8%	-2
320	146.94	127.09	13.4%	-1
480	143.18	126.70	11.5%	-2
640	142.22	126.04	11.4%	-1
1000	141.91	125.76	11.4%	0

Table 6: Impact from hidden layer on heldout data set.

5-10 where they exceed 15.0%, the final improvement reaches 13.5%. In other words, the relative

improvements decrease with increasing iterations.

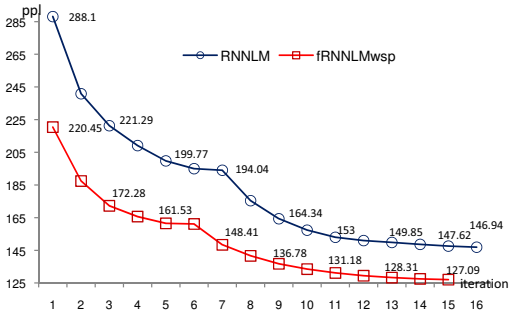


Figure 4: Convergence curve.

4.4 Running-time analysis

This subsection analyzes the time complexity of the two RNNLMs. Table 7 shows the training time of an iteration, the training time of all iterations, and the test time on a PC with 1006G of memory and 24 2.66Ghz CPUs with 144 cores. We observe the following: (1) No significant difference of elapsed time is found between RNNLM and fRNNLM$_{wsp}$ during an iteration of training and test stage. (2) RNNLM requires more time than fRNNLM$_{wsp}$ because it takes 18 iterations to reach a convergence and fRNNLM$_{wsp}$ uses 16 iterations. This experiment shows that although fRNNLM$_{wsp}$ has more free parameters and time complexities (shown in Table 3), it saves time owing to its fast convergence.

	An iteration during training	All iterations during training	During test
RNNLM	48.92m	880m19s	29.18s
fRNNLM$_{wsp}$	49.58m	792m39s	29.35s

Table 7: Elapsed time during training and test. m=minute, s=second.

4.5 Hybrid LM

In the experiments described above, RNNLMs are compared to a 4-gram back-off n-gram language model with modified Kneser-Ney smoothing trained using the SRILM toolkit (Stolcke, 2002). It is also useful to interpolate the recurrent neural network with a back-off n-gram language model to reduce the perplexity and the word error rate. In the following this interpolated model will be denoted by a hybrid language model. Table 8 compares the hybrid RNNLMs in terms of perplexity.

This table demonstrates that the hybrid factored RNNLM also outperforms the hybrid of RNNLM, as we expected. For example, the perplexity reductions of n-gram+fRNNLM over n-gramLM+RNNLM on the heldout and test sets are 8.8% and 9.4%, respectively, and n-gramLM+fRNNLM largely improves the 4-gramLM on the heldout and test sets by 28.9% and 29.6%.

	Heldout	Test
4-gramLM	156.26	156.41
RNNLM	146.94	145.63
fRNNLM$_{wsp}$	127.09	124.63
4-gramLM+RNNLM	121.89	121.62
4-gramLM+fRNNLM$_{wsp}$	$111.20_{+8.8\%}$	$110.19_{+9.4\%}$

Table 8: Perplexities of hybrid language models.

4.6 N-best re-scoring

To evaluate the factored RNNLM in the context of large vocabulary speech recognition, we use the data sets for the IWSLT-2011 large vocabulary continuous speech recognition shared task (Federico et al., 2011) to recognize TED talks published on the TED website[5]. TED talks touch on the environment, photography and psychology without adhering to a single genre. This task reflects the recent increase of interest in automatically transcribing lectures to make them either searchable or accessible.

For LM, the IWSLT-2011 campaign defines a closed set of publicly available English texts, including a small collection of TED transcriptions (in-domain corpus) and a large collection of news sentences (general-domain). All training data are preprocessed by a non-standard-word-expansion tool that converts non-standard words (such as CO2 or 95%) to their pronunciations (CO two, ninety five percent). The most frequent 100K words are extracted from the preprocessed corpora, which, with the CMU.v0.7a pronunciation dictionary[6], are used as the LM vocabulary. Our vocabulary contains 157K entries with an OOV rate of 0.78% on the test2010 data set. For the re-scoring test, we use the IWSLT data sets of tests 2010 and 2011. Their statistics are shown in Table 9.

LM training data			
	#sentences		#words
in-domain	124K		2,063K
general-domain	115,101K		2,458,626K
Test sets			
data	#talks	#utterances	#words
test2010	11	1664	27.0K
test2011	8	818	12.4K

Table 9: Summary of IWSLT2011 data sets

The acoustic models are trained on 170h speech segmented from 788 TED talks that were published prior to 2011. We employ two types of schemes, a Hidden Markov Model (HMM) and a Subspace Gaussian Mixture Model (SGMM) for each context-dependent phone and train them with the Kaldi toolkit (Povey et al., 2011). HMM consists of 6.7K states and 240K Gaussians that are discriminatively trained using the boosted Maximum Mutual Information criterion. SGMM con-

[5] http://www.ted.com/
[6] http://www.speech.cs.cmu.edu/cgi-bin/cmudict

sists of 9.2K states. In addition, we apply speaker adaptive training with feature space maximum likelihood linear regression on top of the HMM and SGMM. The acoustic feature vectors have 40 dimensions. For each frame, we extract 13 static MFCCs, splice 9 adjacent frames, and apply LDA to reduce its dimension with maximum likelihood linear transform. For the in-domain and general-domain corpora, modified Kneser-Ney smoothed 3- and 4-gram LMs are constructed using SRILM (Stolcke, 2002), and interpolated to form a baseline of 3- and 4-gram LMs by optimizing the perplexity of the development data set.

First, we employ a Kaldi speech recognizer (Povey et al., 2011) to decode each utterance using the trained AM and the 3-gram LM. Second, we use the 4-gram LM for lattice re-scoring and generate n-best lists. The n-best size is at most 100 for each utterance. Finally, we use RNNLM and factored RNNLM to re-score the n-best. Note that since it is very time consuming to train RNNLM and factored RNNLM on large data, we only use the in-domain corpus for training them, and the corpus is automatically tagged with parts-of-speech[7] before training fRNNLM$_{wp}$ and fRNNLM$_{wsp}$. The best re-scoring results measured by word error rate are demonstrated in Table 10. We also conduct utterance-level significance tests.

	test2010(%)	**test2011(%)**
4-gram LM	14.34	15.32
4-gram+RNNLM	14.12	15.22
4-gram+fRNNLM$_{wp}$	13.57$^{\dagger}{}_{0.55}$	14.64$^{\dagger}{}_{0.58}$
4-gram+fRNNLM$_{wsp}$	13.65$^{\dagger}{}_{0.47}$	14.59$^{\dagger}{}_{0.63}$

Table 10: n-best re-scoring performance in word-error-rate. Subscript numbers are the absolute improvements over RNNLM. † indicates significantly better results than RNNLM at the p = 0.01 level using a two-sided t-test.

The experimental results show that fRNNLM$_{wp}$ and fRNNLM$_{wsp}$ significantly improves upon 4-gram LM and RNNLM. For example, the absolute improvements of fRNNLM$_{wsp}$ over the 4-gram LM on the sets of tests 2010 and 2011 are 0.69 and 0.73 points, respectively. However, fRNNLM$_{wsp}$ doest not significantly outperforms fRNNLM$_{wp}$. Table 11 demonstrates the re-scoring results sampled from RNNLM and fRNNLM$_{wp}$. This table shows that the results of fRNNLM$_{wp}$ are more grammatically fluent. Fig. 5 illustrates the absolute improvements of fRNNLM$_{wp}$ over RNNLM for each talk in the sets of tests 2010 and 2011. Our approach improves most talks, expect talks 824 and 1183.

Conclusion

In this paper we follow the architecture of a state-of-the-art recurrent neural network language model (RNNLM) and present a factored RNNLM by integrating additional morphological, syntactic, and/or semantic information into RNNLM. Our approach, which is a hybrid of factored n-gram LM and RNNLM, addresses the problems in them. In experiments, we investigate the influences of four commonly used types of features on our factored RNNLM: word, stem, lemma and part-of-speech. We carry out many experiments to evaluate the factored RNNLM performance and analyze the influencing factors. Our experimental results prove that factored RNNLM consistently outperforms n-gram LM and RNNLM for all considered tasks.

[7] `http://www.nactem.ac.uk/tsujii/software.html`

model	**result**
Reference	or we'll be here all day with my childhood stories
RNNLM	* THE WORLD WE'RE all day with my childhood stories
fRNNLM$_{wp}$	or WILL be here all day with my childhood stories
Reference	but don't worry if you can't see it so well
RNNLM	* * TILLER if you can't see it so well
fRNNLM$_{wp}$	* don't worry if you can't see it so well
Reference	and so you're standing there and everything else is dark but there's this portal that you wanna jump in
RNNLM	and so you're * STAYING IN ANYTHING else * TO START there's this portal that you WANT TO jump in
fRNNLM$_{wp}$	and so you're * STAYING IN ANYTHING else is dark but there's this portal that you WANT TO jump in
Reference	AND by the way here are four doctors in your part of the united states who offer it and their phone numbers
RNNLM	* by the way here are four doctors in your part of the united states who * OFFERED and their phone numbers
fRNNLM$_{wp}$	* by the way here are four doctors in your part of the united states who offer it and their phone numbers

Table 11: Re-scoring results sampled from RNNLM and fRNNLM$_{wp}$. * denotes deletion errors, capitalized words denote substitution errors, and underlined words show their differences.

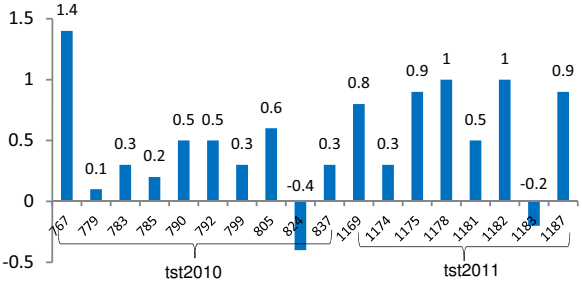


Figure 5: Absolute improvement on each talk.

Recently, syntactic parse trees are used in many advanced LMs (Chelba and Jelinek, 1998; Khudanpur and Wu, 2000; Xu et al., 2002; Collins et al., 2005; Rastrow et al., 2012). For future work, we intend to investigate topic information (Shi et al., 2012) and richer syntactic structure features into factored RNNLM, such as context-free rule productions, constituent/head features, and head-to-head dependencies that can be extracted using parser tools. Second, neural networks are notorious for being time consuming during training, future studies will also focus on speeding up the training of factored RNNLM using graphical processing units (Schwenk et al., 2012). Furthermore, factored RNNLMs need to be evaluated on other tasks like MT and with other languages such as Czech, Arabic, and Turkish.

References

Alexandrescu, A. and Kirchhoff, K. (2006). Factored neural language models. In *Proceedings of the NAACL 2006*, pages 1–4, New York, USA.

Arisoy, E., Sainath, T. N., Kingsbury, B., and Ramabhadran, B. (2012). Deep neural network language models. In *Proceedings of NAACL-HLT 2012 Workshop: Will We Ever Really Replace the N-gram Model? On the Future of Language Modeling for HLT*, pages 20–28, Montreal, Canada.

Bengio, Y., Ducharme, R., Vincent, P., and Jauvin, C. (2003). A neural probabilistic language model. *Journal of Machine Learning Research*, pages 1137–1155.

Boden, M. (2002). A guide to recurrent neural networks and backpropagation. In *The Dallas Project, Sics Technical Report*.

Brown, P. F., deSouza, P. V., Mercer, R. L., Pietra, V. J. D., and Lai, J. C. (1992). Class-based n-gram models of natural language. *Computational Linguistics*, 18(4):467–470.

Chelba, C. and Jelinek, F. (1998). Exploiting syntactic structure for language modeling. In *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics*, pages 225–231, Montreal, Canada.

Chelba, C. and Jelinek, F. (2000). Structured language modeling. *Computer Speech and Language*, 13(4):283–332.

Chen, S. F. and Goodman, J. T. (1996). An empirical study of smoothing techniques for language modeling. In *Proceedings of ACL 1996*, pages 310–318, California, USA.

Collins, M., Roark, B., and Saraclar, M. (2005). Discriminative syntactic language modeling for speech recognition. In *Proceedings of ACL 2005*, pages 507–514, Michigan, USA.

Duh, K. and Kirchhoff, K. (2004). Automatic learning of language model structure. In *Proceedings of the International Conference on Computational Linguistics (COLING) 2004*, pages 148–154, Geneva, Switzerland.

Emami, A. and Jelinek, F. (2004). Exact training of a neural syntactic language model. In *Proceedings of ICASSP 2004*, pages 245–248, Montreal, Canada.

Federico, M., Bentivogli, L., Paul, M., and Stuker, S. (2011). Overview of the iwslt 2011 evaluation campaign. In *Proceedings of IWSLT 2011*, pages 11–27, San Francisco, USA.

Gildea, D. and Hofmann, T. (1999). Topic-based language models using em. In *Proceedings of Eurospeech 1999*, pages 2167–2170.

Goodman, J. (2001). Classes for fast maximum entropy training. In *Proceedings of ICASSP 2001*, Utah, USA.

Hsu, B.-J. P. and Glass, J. (2006). Style and topic language model adaptation using hmm-lda. In *Proceedings of EMNLP 2006*, pages 373–381, Sydney, Australia.

Khudanpur, S. and Wu, J. (2000). Maximum entropy techniques for exploiting syntactic, semantic and collocational dependencies in language modeling. *Computer Speech and Language*, pages 355–372.

Koehn, P. and Hoang, H. (2007). Factored translation models. In *Proceedings of EMNLP 2007*, pages 868–876, Prague, Czech Republic.

Kuo, H.-K. J., Arisoy, E., Emami, A., and Vozila, P. (2012). Large scale hierarchical neural network language models. In *Proceedings of Interspeech 2012*.

Kuo, H.-K. J., Mangu, L., Emami, A., Zitouni, I., and Lee, Y.-S. (2009). Syntactic features for arabic speech recognition. In *Proceedings of Automatic Speech Recognition & Understanding (ASRU) 2009*, pages 327–332, Merano, Italy.

Mikolov, T., Anoop, D., Stefan, K., Burget, L., and Cernocky, J. (2011a). Empirical evaluation and combination of advanced language modeling techniques. In *Proceedings of INTERSPEECH 2011*, pages 605–608, Florence, Italy.

Mikolov, T., Karafiat, M., Burget, L., Cernocky, J. H., and Khudanpur, S. (2010). Recurrent neural network based language model. In *Proceedings of INTERSPEECH 2010*, pages 1045–1048, Makuhari, Japan.

Mikolov, T., Kombrink, S., Burget, L., Cernocky, J., and Khudanpur, S. (2011b). Extensions of recurrent neural network language model. In *Proceedings of ICASSP 2011*, pages 5528–5531, Prague, Czech Republic.

Povey, D., Ghoshal, A., Boulianne, G., Burget, L., Glembek, O., Goel, N., Hannemann, M., Motlicek, P., Qian, Y., Schwarz, P., Silovsky, J., Stemmer, G., and Vesely, K. (2011). The kaldi speech recognition toolkit. In *IEEE 2011 Workshop on Automatic Speech Recognition and Understanding*. IEEE Signal Processing Society. IEEE Catalog No.: CFP11SRW-USB.

Rastrow, A., Dredze, M., and Khudanpur, S. (2012). Fast syntactic analysis for statistical language modeling via substructure sharing and uptraining. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics*, pages 175–183, Jeju, Korea.

Rosenfeld, R. (1996). A maximum entropy approach to adaptive statistical language modeling. *Computational Linguistics*, 10(3):187–228.

Schwenk, H. (2007). Continuous space language models. *Computer Speech and Language*, 21(3):492–518.

Schwenk, H., Rousseau, A., and Attik, M. (2012). Large, pruned or continuous space language models on a gpu for statistical machine translation. In *Proceedings of NAACL-HLT 2012 Workshop: Will We Ever Really Replace the N-gram Model? On the Future of Language Modeling for HLT*, pages 11–19, Montreal, Canada.

Seide, F., Li, G., Chen, X., and Yu, D. (2011). Feature engineering in context-dependentdeep neural networks for conversational speech transcription. In *Proceedings of ASRU 2011*, Hawaii, USA.

Shi, Y., Wiggers, P., and Catholijn, M. J. (2012). Towards recurrent neural networks language models with linguistic and contextual features. In *Proceedings of Interspeech 2012*.

Stolcke, A. (2002). Srilm - an extensible language modeling toolkit. In *Proceedings of INTERSPEECH 2002*, pages 901–904, Colorado, USA.

Xu, P., Chelba, C., and Jelinek, F. (2002). A study on richer syntactic dependencies for structured language modeling. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 191–198, Philadelphia, USA.

Xu, P. and Jelinek, F. (2004). Random forests in language modeling. In *Proceedings of EMNLP 2004*, pages 325–332, Barcelona, Spain.

Multi-View AdaBoost for Multilingual Subjectivity Analysis

Min Xiao *Yuhong Guo*
Department of Computer and Information Sciences
Temple University, Philadelphia, PA, USA
minxiao@temple.edu, yuhong@temple.edu

ABSTRACT
Subjectivity analysis has received increasing attention in natural language processing field. Most of the subjectivity analysis works however are conducted on single languages. In this paper, we propose to perform multilingual subjectivity analysis by combining multi-view learning and AdaBoost techniques. We aim to show that by boosting multi-view classifiers we can develop more effective multilingual subjectivity analysis tools for new languages as well as increase the classification performance for English data. We empirically evaluate our two multi-view AdaBoost approaches on the multilingual MPQA dataset. The experimental results show the multi-view AdaBoost approaches significantly outperform existing monolingual and multilingual methods.

KEYWORDS: Multi-view learning, AdaBoost, Multilingual subjectivity analysis.

Proceedings of COLING 2012: Technical Papers, pages 2851–2866,
COLING 2012, Mumbai, December 2012.

1 Introduction

Subjectivity analysis has received increasing interest in natural language processing (NLP) area (Banea et al., 2010; Alm, 2011; Abdul-Mageed and Diab, 2011; Abdul-Mageed et al., 2011). Subjectivity refers to the expression of emotions, sentiments, opinions, beliefs, speculations, evaluations, as well as other private states (Banfield, 1982; Wiebe, 1994). Subjectivity classification aims to distinguish whether a given text expresses subjective or objective meaning (Wiebe and Cardie, 2005; Abdul-Mageed et al., 2011; Banea et al., 2008, 2010). Subjectivity analysis has been intensively studied, particularly motivated by the prevalent need for opinion-related applications, including mining opinions from product reviews (Pang et al., 2002; Hu and Liu, 2004) or political news (Abbott et al., 2011), and recognizing stances in online debates (Somasundaran and Wiebe, 2009, 2010). Moreover, many NLP tasks employ subjectivity analysis as an additional layering to filter data. Research that benefited from this phase ranges from conversation summarization (Seki et al., 2005; Carenini et al., 2008) and information extraction (Riloff et al., 2005) to text semantic analysis (Wiebe and Mihalcea, 2006) and question answering (Li et al., 2008; Yu and Hatzivassiloglou, 2003).

Although subjectivity analysis has been widely studied in NLP area, much work has only focused on English data. Recently, some researchers propose to carry out subjectivity analysis in a multilingual framework based on machine translation, where resources or tools of subjectivity analysis developed in one language are used to support developing resources or tools in another language (Mihalcea and Banea, 2007; Banea et al., 2008, 2010). The approaches in (Mihalcea and Banea, 2007; Banea et al., 2008) however only exploit the translated target-language-view of the data to develop a subjectivity analysis tool, which is a waste of resources in a multilingual setting since possible parallel views of the data are ignored. Banea et al. (2010) propose to overcome this shortcoming by conducting subjectivity analysis based on concatenated multilingual input feature vectors. This simple feature combination method nevertheless is still very preliminary in exploring the capacity of multi-view learning for subjectivity analysis on multilingual data.

In this paper, we propose to use multi-view AdaBoost approaches for multilingual subjectivity analysis, which combine the advantages of both multi-view learning and AdaBoost learning in one integrated framework. By exploring multi-view learning, we expect to exploit the complementary discriminative information in different language views. By incorporating the multi-view learning into an AdaBoost framework, we expect to further boost the classification accuracy of the integrated models. Based on different strategies of exploring multilingual information, we develop two approaches in this paper: Multi-View Majority Voting AdaBoost (*MVAB1*) and Multi-View Weighted Voting AdaBoost (*MVAB2*).

To demonstrate the effectiveness of the proposed approaches, we empirically evaluate them on a multilingual subjectivity analysis dataset, the Multilingual Multi-Perspective Question Answering (MPQA) corpus. To justify the robustness of our boosting framework, we conduct experiments using two types of base classifiers, Support Vector Machines (SVM) and Naïve Bayes (NB). The experimental results show that the proposed approaches can significantly outperform other comparison methods for multilingual subjectivity analysis. Overall, the contributions of this paper can be summarized as below:

- We propose two multi-view AdaBoost algorithms, Multi-View Majority Voting AdaBoost and Multi-View Weighted Voting AdaBoost, which can be widely used for multilingual classification tasks when parallel corpora or machine translation is available.

- Experimentally, we evaluate our approaches on Multilingual MPQA corpus and obtain a subjectivity classifier with *accuracy* as high as 78.19% and *macro F1* as high as 77.44% over all six languages.

The remainder of the paper is organized as follows. Related work is presented in Section 2. In Section 3, we present the multilingual subjectivity analysis problem and two proposed multi-view AdaBoost approaches. In Section 4, we present the experimental results and discussions. We then conclude the paper.

2 Related Work

The importance of subjectivity analysis has been widely acknowledged by language analysts, including computational linguists. Due to the availability of data resources, much work on subjectivity analysis has focused on English data alone. However, recently, some work tries to generate resources and develop tools for other languages by transferring labeled English subjectivity resources and corresponding analysis tools.

Mihalcea and Banea (2007) proposed to build subjectivity classifiers for Romanian data by leveraging the resources and tools available in English. They developed a lexicon-based approach and a corpus-based approach. For the lexicon-based approach, they first created a target-language subjectivity lexicon by translating the existing annotated English subjectivity lexicon via bilingual dictionaries and then trained a rule-based classifier relying on the translated lexicon. For the corpus-based approach, they first manually translated an automatically annotated English corpus into Romanian language and projected the subjectivity annotations correspondingly, and then trained a statistical classifier on the resulting corpus. They empirically evaluated their approaches on MPQA corpus and SemCor corpus (Miller et al., 1993), showing that the corpus translations preserve subjectivity more reliably than the lexicon translations. Nevertheless, the requirement for manual translation is a big restriction for potential usage of the proposed approaches.

Banea et al. (2008) then proposed to generate resources and tools for new languages (Spanish and Romanian) using machine translation and cross-lingual annotation projections. Specifically, they used machine translation to transfer the manually or automatically annotated training data from the source language (English) into the target languages, and projected the subjectivity annotations of the transferred data across language correspondingly. Then they employed statistical machine learning techniques such as Support Vector Machines and Naïve Bayes to produce a subjectivity classifier on the translated corpus in the target language. The advantage of this approach is that it does not need any original target language data for training. Thus it can be widely used for any new target language as long as a source-target-language machine translation engine is available. Nonetheless, the subjectivity analysis tool developed by this approach is dependent on the quality of machine translation since only translated data is used in training.

Banea et al. (2010) proposed to combine multiple language spaces altogether in an expanded feature space. Specifically they combined the original English feature vector of an instance and its translated feature vectors in different target languages together into one feature vector, and then used the training instances expressed in this expanded feature space to train multilingual subjectivity classifiers. They empirically evaluated their approach on MPQA corpus, by translating English sentences into five other languages. Their empirical results showed that multiple languages can complement each other to greatly increase subjectivity classification

performance for target languages as well as for English source data, comparing to training subjectivity classifiers on the target language alone. Nevertheless, the parallel texts in multiple languages can be approximately taken as label-conditionally independent multiple views of the same set of data objects, and their simple feature space expanding method is still far from fully exploiting this multilingual information. Thus, in this work we investigate new multi-view AdaBoost approaches to improve the performance of multilingual subjectivity analysis.

Multilingual views have also been exploited in sentiment analysis (Wan, 2009; Lu et al., 2011). Wan (2009) used co-training on bilingual views (Chinese and English) generated from machine translation to perform sentiment analysis on Amazon product reviews. Their approach however requires in-domain data from the target language for training. Lu et al. (2011) developed a maximum entropy based statistical model to jointly train two monolingual sentiment classifiers using an EM-algorithm. They also only studied the bilingual situation with experiments on English and Chinese. Combining multi-view learning and boosting has been studied in a few different ways on application problems outside of NLP field, including a semi-supervised boosting method for object category recognition and visual object tracking (Saffari et al., 2010), and an embedded two-view AdaBoost method for UCI data (Xu and Sun, 2010). But our work is the first one that combines multi-view learning and AdaBoost learning to address supervised subjectivity analysis with multiple languages.

3 Multi-View AdaBoost for Multilingual Subjectivity Analysis

In this section, we introduce two multi-view AdaBoost approaches, which combines the advantages of both multi-view learning and boosting learning to achieve better multilingual subjectivity classifiers. Below, we will first describe the general framework of multilingual subjectivity analysis and briefly introduce the AdaBoost algorithm, and then present the two proposed multi-view AdaBoost approaches.

3.1 Multilingual Subjectivity Analysis

Banea et al. (2010) pointed out that training a subjectivity classifier on the resulting target monolingual corpus alone, though works, is not good enough. There are two main reasons. First, in order to correctly predict labels based on statistical information, a sufficient amount of training data is needed, which may not be available in the monolingual corpus. Second, in the monolingual corpus alone, some discriminative features present in the test data may not appear in the training data and therefore their information cannot be used to generate an effective classifier. In both cases, multilingual subjectivity analysis can have advantages.

Below we demonstrate the problems of learning with monolingual corpus using examples from the multilingual MPQA dataset. Following (Banea et al., 2010)'s suggestions, we assume that only the words in italics carry potential subjective meaning and their surrounding contexts would be objective if without them. Therefore, their association with an either subjective or objective sense imparts the same label to the whole segment.

We explore the first data sparseness problem through the following two examples (En 1 and En 2) from the English version of the MPQA dataset as well as their respective translations in Spanish (Es 1 and Es 2):

> "En 1: The source said that the ministry would soon deliver copies of the report to the various ministries *concerned*, especially the Interior and Municipalities Ministry,

prior to relaying its observations to the State Department in Washington."

"Es 1: La fuente dijo que el ministerio pronto entregar copias del informe a los distintos ministerios *interesados*, en particular el Ministerio del Interior y de los Municipios, antes de transmitir sus observaciones al Departamento de Estado en Washington."

"En 2: Still, the overseers of the prison are *concerned* that detainees aren't getting enough pita bread with their meals, and they're planning to make the food spicier, just the way the prisoners like it back home. "

"Es 2: Sin embargo, los supervisores de la prisión tienen la *preocupación* de que los detenidos no están recibiendo suficiente pan pita con sus comidas, y que está pensando en hacer la comida spicier, sólo la forma en la que los reclusos como para volver a casa."

We focus on the word *concerned*. In the first example (En 1), it is used with an objective sense, which means a group of ministries defined earlier in the context. While in the second example (En 2), *concerned* serves as a subjective carrier. If we train a monolingual classifier on the English data alone, due to the data sparseness paradigm, our machine learning model may not distinguish between the word's subjective and objective senses when inferring a label for the whole sentence. However, the corresponding translations of *concerned* in Spanish are functionally different because of the surrounding context. We denote the respective translations in the Spanish context (Es 1 and Es 2) using the italic form. If we take the Spanish translation into consideration during training, we may obtain a classifier, which can potentially differentiate between the senses and predict the correct sentence label.

Next, we explore the second problem with two other examples below (En 3 and En 4) extracted from the MPQA dataset and their corresponding machine translations in Romanian:

"En 3: What is the point of engaging in dialogue with a government that is only interested in buying time while it fervently escalates a campaign of bludgeoning its citizens in the hope of frightening voters into *supporting* Mugabe? "

"Ro 3: Ce este pe punctul de angajarea in dialog cu un guvern doar ca este interesat de cumpararea timp in timp ce aceasta inflacarare e o campanie de forta cetatenilor sai in speranta de infricosator electoratul in *sprijinirea* lideri? "

"En 4: According to the sources, the EU which was barred by the Government from observing the election because some of its members were openly *supporting* the MDC, sent a Ms Maria Macchiaverna to "support the financial management of our assistance" to the Sadc Parliamentary Forum and ZESN."

"Ro 4: Potrivit surselor, UE care a fost oprita de guvernul de la observarea alegerilor pentru ca unii dintre membrii sai s-au deschis *sustinerea* mdc, a trimis un MS Maria macchiaverna sa "sprijin financiar de gestiune noastra de asistenta" la parlamentare sadc Forumului si zesn. "

In both of the two examples (En 3 and En 4), *supporting* carries subjective senses. However, the corresponding translations of *supporting* in Romanian for En 3 and En 4 are different: *sprijinirea* (in Ro 3) and *sustinerea* (in Ro 4). If we train a monolingual classifier on Romanian corpus alone, and the training set contains *sprijinirea* but not *sustinerea*, it is hard to infer the correct label for a context containing *sustinerea* by leveraging information from *sprijinirea*. However, if we adopt a multilingual classifier, we may be able to predict a correct label for the

context containing *sustinerea* by using the English information, or the associations between *supporting* and *sprijinirea*, as well as *supporting* and *sustinerea*.

As suggested in these examples above, exploiting multilingual information can compensate the shortcomings of learning from monolingual data. Multilingual subjectivity analysis has been previously studied in (Banea et al., 2008, 2010). We propose to conduct multilingual subjectivity analysis following the general framework suggested in these works. Assume we have manually annotated subjectivity corpus in English, and aim to develop subjectivity resources and tools for other languages, such as Arabic, French, German, Romanian, and Spanish, for which there are few text processing resources and tools to date. The multilingual subjectivity analysis framework contains three steps:

- Translating English sentences into target languages by using machine translation.
- Projecting subjectivity annotations from English to translated target languages.
- Producing subjectivity analysis tools on the resulting labeled corpora by using statistical machine learning techniques.

Banea et al. (2010) empirically studied the multilingual subjectivity analysis problem and provided a simple solution by expanding the feature space with multiple languages. In this work, we propose to further improve multilingual subjectivity analysis by exploiting multi-view learning in the framework of AdaBoost. In addition to the multilingual analysis problem discussed above, our approaches have the following two standpoints. First, Amini et al. (2009) provided theoretical bounds for multi-view classifiers and showed that additional views generated by machine translation may significantly improve classification performance. Second, within an AdaBoost framework, the algorithms can deal with hard examples which are difficult to be recognized using base multi-view learners. Moreover, an exponential loss function is guaranteed to be minimized over the multilingual data.

3.2 AdaBoost

Boosting is a general method for improving accuracy of any given learning algorithm by combining many weak or base learners. A well-known boosting algorithm is AdaBoost, which was introduced by (Freund and Schapire, 1997). AdaBoost takes a set of training instances $(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)$ as input, where each x_i belongs to some instance space $\mathscr{X}$, and each label y_i is in some label space $\mathscr{Y}$. For a binary classification problem, we assume $\mathscr{Y} = \{+1, -1\}$. AdaBoost calls a given base learning algorithm repeatedly in a series of rounds $t = 1 \cdots T$ and maintains a set of weights or a distribution over the training instances during the rounds. All weights are initialized equally; but in each round, the weights of misclassified instances are increased so that the weak learner will be forced to focus on the hard instances in the training set in the next round.

In each round, the weak learner trains a base classifier $h_t : \mathscr{X} \rightarrow \mathscr{Y}$ over the training instances with the weighted distribution D_t. For the binary classification problem, the base learner's job is to minimize the error

$$\epsilon_t = Pr_{i \sim D_t}[h_t(x_i) \neq y_i] \tag{1}$$

Once the base classifier h_t has been found, AdaBoost chooses a parameter α_t that intuitively

Algorithm 1 Multi-view Majority Voting AdaBoost

Input: A multi-view binary training set $\{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)\}$, with $y_i \in \{+1, -1\}$.
Output: The final classifier H.
Initialize: $D_1(i) = \frac{1}{n}$.
for $t = 1$ **to** T **do**

- Separately train a set of single view classifiers $\{h_{tv}\}$ using distribution D_t.
- Compute the base classifier h_t using the majority voting scheme in Eq. (5).
- Set $\epsilon_t = Pr_{i \sim D_t}[h_t(\mathbf{x}) \neq y_i]$
- Set $\alpha_t = \frac{1}{2} \ln \frac{1-\epsilon_t}{\epsilon_t}$.
- Update $D_{t+1}(i) = D_t(i) \frac{e^{-\alpha_t y_i h_t(\mathbf{x}_i)}}{Z_t}$, where Z_t is a normalization factor.

end for
$H(\mathbf{x}) = sign(\sum_{t=1}^{T} \alpha_t h_t(\mathbf{x}))$

measures the confidence or importance that it assigns to h_t. For binary h_t, α_t is set as

$$\alpha_t = \frac{1}{2} \ln(\frac{1-\epsilon_t}{\epsilon_t}) \tag{2}$$

in (Freund and Schapire, 1997). The distribution D_t is then updated as

$$D_{t+1}(i) = \frac{D_t(i) e^{-\alpha_t y_i h_t(x_i)}}{Z_t} \tag{3}$$

where Z_t is a normalization factor.

The final classifier H produced by AdaBoost is a weighted majority vote of the T base classifiers

$$H(x) = sign(\sum_{t=1}^{T} \alpha_t h_t(x)) \tag{4}$$

where α_t is the weight assigned to h_t.

3.3 Multi-View AdaBoost for Multilingual Subjectivity Analysis

In the multilingual setting, each instance (sentence) is described using feature sets from multiple languages, where each feature set from one language can be treated as one view of the instance. To address multilingual subjectivity analysis in an AdaBoost framework, we propose to integrate multi-view learning into the AdaBoost. In particular, we develop two approaches using different multi-view learning strategies: a multi-view majority voting AdaBoost (MVAB1) and a multi-view weighted voting AdaBoost (MVAB2).

3.3.1 Multi-View Majority Voting AdaBoost

Each view of the multi-view data is expect to be able to learn a classifier independently. Combining different views to achieve a better classification model is the key idea of multi-view learning. Due to the noise and strength of different views, a majority voting scheme has been shown to be effective in multi-view learning (Amini et al., 2009). Amini et al. (2009) proposed to use multi-view majority voting to perform multilingual text classification on parallel data.

Algorithm 2 Multi-view Weighted Voting AdaBoost

Input: A multi-view binary training set $\{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)\}$, with $y_i \in \{+1, -1\}$.
Output: The final classifier H.
Initialize: $D_1(i) = \frac{1}{n}$.
for $t = 1$ **to** T **do**

- Separately train a set of single view classifiers $\{h_{tv}\}$ using distribution D_t.
- Compute the weights $\{\beta_v\}$ by minimizing the weighted least square loss in Eq. (7).
- Compute the base classifier h_t using the weighted voting scheme in Eq. (6).
- Set $\epsilon_t = Pr_{i \sim D_t}[h_t(\mathbf{x}_i)) \neq y_i]$
- Set $\alpha_t = \frac{1}{2} \ln \frac{1-\epsilon_t}{\epsilon_t}$.
- Update $D_{t+1}(i) = D_t(i) \frac{e^{-\alpha_t y_i h_t(\mathbf{x}_i)}}{Z_t}$, where Z_t is a normalization factor.

end for
$H(\mathbf{x}) = sign(\sum_{t=1}^{T} \alpha_t h_t(\mathbf{x}))$

They derived a generalization error bound for classifiers learned on examples with multiple artificially views created using machine translation. They empirically evaluated their approach on a comparable multilingual corpus, Reuters RCV1/RCV2, showing that additional views obtained using a machine translation system can significantly increase classification performance, especially when few labeled data are available for training.

We propose to use multi-view majority voting to produce a base learner within the AdaBoost framework, aiming to improve subjectivity classification performance. Given a weighted training set with multiple views, a multi-view instance can be expressed as $\mathbf{x} = (x^1, x^2, \ldots, x^V)$, where each sub-vector x^v provides a representation of the same object in one feature space $\mathcal{X}^v$. In multilingual subjectivity analysis, each view is the textual representation of instances in a given language (e.g. Arabic, English, French, German, Romania, and Spanish). The Multi-view Majority Voting AdaBoost approach is then carried out in the following way. At each round t, a set of view-specific binary classifiers $\{h_{tv}(x^v)\}$ can be separately trained using specific views on the weighted training data with a distribution D_t. The base classifier is then obtained using a majority voting scheme:

$$h_t(\mathbf{x}) = sign(\sum_{v=1}^{V} h_{tv}(x^v)) \tag{5}$$

where $h_{tv}(x^v) \in \{1, -1\}$. The remaining steps of the AdaBoost procedure for training instance reweighting and final combination parameter determination are same as the standard AdaBoost for binary classifications. The overall procedure of the algorithm is described in Algorithm 1. With this algorithm, we expect to integrate the strengths of the subjectivity analysis data in different languages and boost the subjectivity classification performance.

3.3.2 Multi-View Weighted Voting AdaBoost

The multi-view majority voting scheme assumes the set of view-specific classifiers contribute equally for the final classifier. This assumption is too strong in many cases. We thus propose to pursue a more advanced multi-view combination scheme, where the combination classifier is a linear weighted combination of the view-specific classifiers. This leads to our next approach,

a Multi-view Weighted Voting AdaBoost approach. Similar to Multi-view Majority Voting AdaBoost, at each round t of Multi-view Weighted Voting AdaBoost, we separately train a set of single view classifiers $\{h_{tv}\}$ over the training instances with a weighted distribution D_t, one for each language (view) v. But instead of taking a majority vote, we set the combination base classifier h_t as a linear combination of the single view classifiers with a set of weight parameters $\{\beta_v\}$; i.e.,

$$h_t(\mathbf{x}) = sign(\sum_{v=1}^{V} \beta_v h_{tv}(x^v)) \tag{6}$$

where $0 \leq \beta_v \leq 1$ and $\sum_{v=1}^{V} \beta_v = 1$. In order to obtain the weight parameters, we train this linear combination model on the weighted training set, using the outputs of the single-view classifiers as features. Specifically, we minimize the following weighted least square loss on the training instances to to obtain the β values:

$$L = \sum_{i=1}^{n} D(i)(\sum_{v=1}^{V} \beta_v h_{tv}(x_i^v) - y_i)^2 \tag{7}$$

With the obtained base classifier h_t in Eq. (6), we can then find its importance weight α_t and update the distribution D_t. We proceed with this procedure for T rounds and output the final hypothesis H by combining the importance weighted multilingual base classifiers. The overall algorithm is presented in Algorithm 2.

4 Experiments

In this section, we empirically evaluate our proposed approaches for the task of subjectivity analysis on Multilingual Multi-Perspective Question Answering corpus. We first introduce the dataset and describe implementations, and then present the experimental results.

4.1 Multilingual Dataset

We use the same dataset as studied in (Banea et al., 2010). This is a multilingual subjectivity analysis dataset constructed from the Multi-Perspective Question Answering (MPQA) corpus. The original MPQA corpus contains 535 English newswire articles, which are collected from various sources. Each article is manually annotated for subjectivity labels (Wiebe and Cardie, 2005). Although the original corpus is labeled at the phrase and clause levels, we adopt the sentence-level annotations as suggested by (Banea et al., 2008, 2010). If the sentence contains at least one private state, whose opinion strength is higher or medium, we regard it as *subjective*. Otherwise, we see it as *objective*. Thus, we collected 9732 sentences. Among the 9732 sentences in this corpus, 5380 of them are annotated as subjective, while the rest 4352 sentences are labeled as objective.

In order to obtain comparable corpora to MPQA in other languages, Banea et al. (2010) translated the original English (En) newswire articles into five other languages, namely Arabic (Ar), French (Fr), German (De), Romanian (Ro) and Spanish (Es), using machine translation. To generate subjectivity labeling for target languages, they projected the original sentence-level subjectivity annotation from the source English data onto the target language data. Thus, we get a multilingual subjectivity analysis dataset with six languages. [1]

[1]The original English MPQA corpus can be downloaded from http://www.cs.pitt.edu/mpqa.

Based on this multilingual corpus, we first performed the following preprocessing steps as (Banea et al., 2010) employed. We removed all the numbers, diacritics, all punctuation marks except ' and -. We kept ' and - because they may mark contractions, such as in "they're" (for English) and "s-ar" (for Romanian). Although Arabic has a different encoding, we treated it in a similar way as the Roman language. We then used the library (Lingua::AR::Word PERL Library) to map Arabic script to Roman-alphabet letters. In the multilingual corpus, there are six languages. Then for each language, we removed the rare words and selected the top 20% of (unigram) features to use as suggested by (Banea et al., 2010). We used term presence as weighting scheme for all features, which means if the sentence contains one specific feature, then its corresponding value for that feature is 1, otherwise, the value is 0. We did this for two reasons. First, (Pang et al., 2002) explored different feature weighting schemes for sentiment classification, showing that term presence is better than term frequency in sentiment classification tasks. Second, (Banea et al., 2010) adopted term presence as a weighting scheme in their experiments.

4.2 Implementation

In the experiments, we compared the empirical performance of the following approaches for subjectivity analysis, including the two proposed approaches.

- TDe, TEn, TEs, TFr, TRo: (Banea et al., 2008) proposed to train a subjectivity classifier for a new language on the translated data in the target language alone. We use *TDe* to denote the method that trains a subjectivity classifier on the target German language alone. Similarly, *TEn, TEs, TFr, TRo* denote that we use English, Spanish, French and Romania as the target language respectively.
- MLS: (Banea et al., 2010) proposed to train a multilingual subjectivity classifier by combining all different languages together to expand the feature space. We denote it as a MultiLingual Space method (MLS).
- MVMV: The multi-view majority voting method developed in (Amini et al., 2009).
- MVAB1: The multi-view majority voting AdaBoost approach.
- MVAB2: The multi-view weighted voting AdaBoost approach.

The last four approaches are multilingual approaches and the others are single language based approaches. For all these approaches, we experimented with two different classifiers: Naïve Bayes (NB) and Support Vector Machines (SVM), which are also used in previous studies on multilingual subjectivity analysis (Banea et al., 2008, 2010). We chose them due to their robust performances and the diversity of their learning methodologies. For Naïve Bayes, we used the multinomial model (McCallum and Nigam, 1998). For Support Vector Machines, we used the LIBLINEAR package (Fan et al., 2008). The LIBLINEAR is an open source library and works very efficiently on large sparse data sets. For LIBLINEAR, we set the tradeoff parameter c with the default value, $c = 1$. For the two boosting approaches, MVAB1 and MVAB2, we set the maximum iteration number T as 50.

4.3 Experimental Results

We take all the subjective sentences as positive instances and all the objective sentences as negative instances. The six single view approaches are experimented on each of the six target lan-

Table 1: Average results for the comparison approaches based on SVM classifiers.

Method	SubjP	SubjR	SubjF	ObjP	ObjR	ObjF	AllP	AllR	AllF	MAcc
TEn	75.92	73.78	74.82	68.57	70.96	69.73	72.25	72.37	72.28	72.53
TRo	75.01	73.76	74.37	67.80	69.24	68.51	71.41	71.50	71.44	71.75
TEs	74.04	73.43	73.71	68.26	68.95	68.57	71.15	71.19	71.14	71.39
TFr	75.04	73.00	73.99	67.21	69.48	68.31	71.13	71.24	71.15	71.47
TDe	72.97	71.93	72.44	65.91	67.05	66.46	69.44	69.49	69.45	69.75
TAr	72.70	72.06	72.35	65.76	66.47	66.08	69.23	69.26	69.22	69.55
MLS	76.72	76.00	76.34	70.45	71.29	70.84	73.59	73.65	73.59	73.89
MVMV	76.78	77.99	77.37	72.79	71.36	72.06	74.79	74.68	74.72	75.01
MVAB1	77.95	79.15	78.53	74.68	73.29	73.95	76.32	76.22	76.24	76.47
MVAB2	78.62	79.39	78.98	75.03	74.12	74.54	76.83	76.75	76.76	76.97

guages alone. The four multilingual approaches use all the parallel texts in the six languages. We performed ten-fold cross validation on the multilingual dataset as did in (Banea et al., 2010). Two sets of such experiments are conducted, with Support Vector machines and Naïve Bayes as base classifiers respectively. The average test results of different approaches are reported in Table 1 for SVM and in Table 2 for Naïve Bayes.

Each test result is evaluated with 10 measurements denoted with the following abbreviation style: *Obj* represents objective, *Subj* represents subjective, and *All* stands for overall macro measures, computed over the objective and subjective classes; *P*, *R*, *F*, and *MAcc* correspond to precision, recall, F1-measure and macro-accuracy, where

$$Precision = \frac{\#\text{ of correct positive predictions}}{\#\text{ of positive predictions}},$$

$$Recall = \frac{\#\text{ of correct positive predictions}}{\#\text{ of positive instances}},$$

$$F1 = 2\,\frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}.$$

From Table 1, we can see that all the four multilingual methods consistently outperform the single-view methods across all languages in terms of all 10 measurements. This verifies the hypothesis that training on translated target language alone is not enough and multilingual subjectivity analysis is useful. Among the four multilingual methods, *MVMV* performs slightly better than *MLS* in terms of subjective precision and objective recall, while *MVAB1* and *MVAB2* significantly outperform both *MLS* and *MVMV* in terms of all ten measurements. The *MVAB2* performs slightly better than *MVAB1* in terms of all ten measurements. Comparing to *MLS*, *MVAB1* improves the accuracy by 2.58%, and improves the macro F1-measure by 2.65%; *MVAB2* improves the accuracy by 3.08%, and improves the macro F1-measure by 3.17%. Comparing to *MVMV*, *MVAB1* improves the accuracy by 1.46% and improves the macro F1-measure by 1.52%; *MVAB2* improves the accuracy by 1.96%, and improves the macro F1-measure by

Table 2: Average results for the comparison approaches based on Naïve Bayes classifiers.

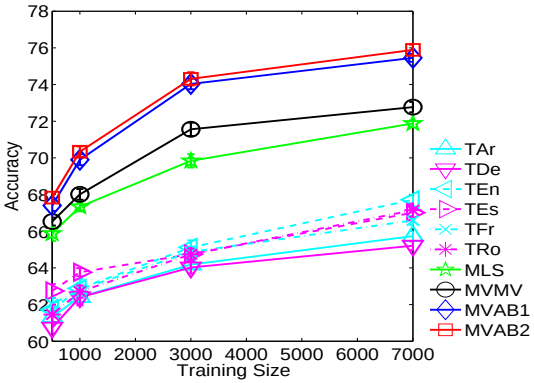
Method	SubjP	SubjR	SubjF	ObjP	ObjR	ObjF	AllP	AllR	AllF	MAcc
TEn	74.01	83.64	78.53	75.89	63.68	69.25	74.95	73.66	73.89	74.72
TRo	73.50	82.06	77.54	74.08	63.40	68.33	73.79	72.73	72.94	73.72
TEs	74.02	82.84	78.19	75.11	63.05	69.14	74.57	73.44	73.66	74.44
TFr	73.83	83.03	78.16	75.19	63.61	68.92	74.51	73.32	73.54	74.35
TDe	73.26	83.49	78.04	75.32	62.30	68.19	74.29	72.90	73.12	74.02
TAr	71.98	81.47	76.43	72.62	60.78	66.17	72.30	71.13	71.30	72.22
MLS	75.43	83.66	79.33	76.64	66.30	71.10	76.04	74.98	75.21	75.89
MVMV	75.91	84.56	79.98	77.47	66.38	71.46	76.69	75.47	75.72	76.49
MVAB1	76.95	85.49	80.98	78.92	67.91	72.98	77.93	76.70	76.98	77.68
MVAB2	77.74	85.73	81.53	78.96	68.52	73.34	78.35	77.13	77.44	78.19

2.04%. It is worth mentioning that both *MVAB1* and *MVAB2* increase the overall precision and recall levels to above 76%. Using a paired t-test, all these improvements were found to be significant at p=0.001. Those improvements demonstrate that our Multi-view AdaBoost approaches are more effective than simply expanding the feature space with multiple languages, or only using a simple multi-view majority voting strategy.

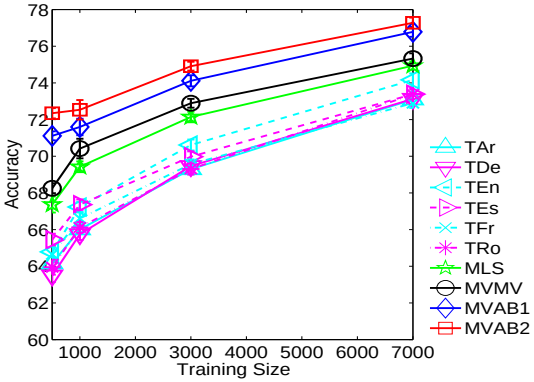
From Table 2 we can see that again the four multilingual methods outperform the single language methods in terms of macro precision, recall and F1-measure as well as macro accuracy. However, *MLS* achieves almost the same performance in terms of subjective recall measurement as the single language methods, *TEn* and *TDe*, which implied that more advanced multilingual models are needed. The two proposed multi-view AdaBoost approaches, *MVAB1* and *MVAB2*, outperform all the other comparison methods in terms of macro precision, and F1-measure as well as macro accuracy. Even in term of recall for subjective and objective classes, both of *MVAB1* and *MVAB2* outperform all other methods. Comparing to *MLS*, *MVAB1* improves the accuracy by 1.79%, and improves the macro F1-measure by 1.77%; *MVAB2* improves the accuracy by 2.30%, and improves the macro F1-measure by 2.23%. Comparing to *MVMV*, *MVAB1* improves the accuracy by 1.19%, and improves the macro F1-measure by 1.26%; *MVAB2* improves the accuracy by 1.70%, and improves the macro F1-measure by 1.72%. All the improvements were found to be significant at p=0.001 by using a paired t-test. These results demonstrate that our proposed multi-view AdaBoost approaches are robust to different types of base classifiers, and their advantages can be maintained. This again suggests the proposed multi-view AdaBoost approaches provide a more effective framework to exploit multilingual information for multilingual subjectivity analysis than the existing methods.

4.4 Impact of Training Size

Next we studied the impact of the training size for different approaches. Among the 9732 sentences, we randomly selected 2732 (about 1500 subjective sentences and 1232 objective sentences) as test data. From the rest 7000 sentences, we randomly selected training instances with a range of different sizes $m \in \{500, 1000, 3000, 7000\}$. For each training size, we repeated



(a) Results based on SVM classifiers.



(b) Results based on Naïve Bayes classifiers.

Figure 1: Average classification accuracies ± standard deviations across a range of different training sizes. The top figure presents results based on Support Vector Machine classifiers and the bottom figure presents results based on Naïve Bayes classifiers.

the experiments over 10 runs by randomly selecting the training instances from the 7000 sentences. We again experimented with the same two base classifiers: SVM and Naïve Bayes (NB). We compared the four multilingual methods with the six single language methods. We reported the average test accuracies and standard deviations in Figure 1. We can see that for both SVM-based classifiers and Naïve Bayes-based classifiers, the four multilingual methods consistently outperform the six single language methods across the range of different training sizes. The improvements of multilingual methods over the single language methods are clearly significant across all range of training sizes for both SVM-based classifiers and Naïve Bayes-based classifiers, which justified that using multilingual information enables every single

Table 3: Results for Statistical Significance (McNemar's) test.

Null Hypothesis	p-value (SVM, Trainsize=500)	p-value (NB, Trainsize=7000)
MVAB1 vs. TEn	6.3×10^{-10}	4.1×10^{-9}
MVAB1 vs. MLS	7.9×10^{-7}	8.5×10^{-8}
MVAB1 vs. MVMV	3.5×10^{-4}	2.7×10^{-5}
MVAB2 vs. TEn	1.3×10^{-10}	1.8×10^{-9}
MVAB2 vs. MLS	2.6×10^{-7}	3.1×10^{-8}
MVAB2 vs. MVMV	1.2×10^{-4}	1.9×10^{-5}

language to reach a high accuracy that is not individually obtainable. Among the four multilingual methods, our proposed two multi-view AdaBoost methods, *MVAB1* and *MVAB2*, significantly outperform the other two simple multi-view methods, *MLS* and *MVMV*. With the more sophisticated view weighted training, *MVAB2* outperforms *MVAB1*. More specifically, for the SVM-based classifiers, when the training size is 3000, *MVAB1* increases the accuracy by 4.19% over *MLS* and by 2.47% over *MVMV*; *MVAB2* increases the accuracy by 4.50% over *MLS* and by 2.75% over *MVMV*. Their advantages over the single language methods are even much larger. *MVAB1* increases the accuracy of the best single language method (over English) by 8.91%. *MVAB2* increases the accuracy of the best single language method (over English) by 9.19%. For NB-based classifiers, our proposed approaches achieve their best improvements when the training size is small (500). In this case, *MVAB1* increases the accuracy by 6.33% comparing to the best single language method over English, increases the accuracy by 3.74% comparing to *MLS*, and increases the accuracy by 2.86% comparing to *MVMV*. *MVAB2* performs even better, and it increases the accuracy by 7.56% comparing to the best single language method over English, increases the accuracy by 4.98% comparing to *MLS*, and increases the accuracy by 4.10% comparing to *MVMV*.

To justify the significance of those improvements across ranges, we used a McNemar paired test for labeling disagreements (Gillick and Cox, 1989) with $p < 0.05$ being significant. We focus on the improvements the two proposed approaches obtained over other methods. Since *TEn* works the best among models trained with monolingual corpus, we select it as the representative for all six monolingual methods. From Figure 1, we can see that the two proposed approaches achieve the smallest improvement when the training size is very small (500) for SVM-based classifiers, and when the training size is very big (7000) for NB-based classifiers. We thus select those results to conduct significance test. We report the p values in Table 3. From Table 3, we can see that all the improvements made by *MVAB1* and *MVAB2* over other methods are statistically significant. These results again demonstrate the efficacy of the proposed multi-view AdaBoost framework on multilingual subjectivity analysis.

Conclusion

In this paper, we proposed to integrate multi-view learning into the AdaBoost framework to perform multilingual subjectivity analysis, with the aim of developing more effective subjectivity analysis tools for both English and languages beyond English. Our experimental results on multilingual MPQA corpus show that the approaches developed within our proposed framework can significantly outperform other existing methods.

References

Abbott, R., Walker, M., Anand, P., Fox Tree, J. E., Bowmani, R., and King, J. (2011). How can you say such things?!?: recognizing disagreement in informal political argument. In *Proc. of the Workshop on Languages in Social Media*.

Abdul-Mageed, M. and Diab, M. T. (2011). Subjectivity and sentiment annotation of modern standard arabic newswire. In *Proc. of the 5th Linguistic Annotation Workshop*.

Abdul-Mageed, M., Diab, M. T., and Korayem, M. (2011). Subjectivity and sentiment analysis of modern standard arabic. In *Proc. of the Annual Meeting of the Association for Computational Linguistics (ACL)*.

Alm, C. O. (2011). Subjective natural language problems: motivations, applications, characterizations, and implications. In *Proc. of the Annual Meeting of the Association for Computational Linguistics (ACL)*.

Amini, M., Usunier, N., and Goutte, C. (2009). Learning from multiple partially observed views - an application to multilingual text categorization. In *Advances in Neural Information Processing Systems (NIPS)*.

Banea, C., Mihalcea, R., and Wiebe, J. (2010). Multilingual subjectivity: are more languages better? In *Proc. of the International Conference on Computational Linguistics (COLING)*.

Banea, C., Mihalcea, R., Wiebe, J., and Hassan, S. (2008). Multilingual subjectivity analysis using machine translation. In *Proc. of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Banfield, A. (1982). *Unspeakable sentences: Narration and representation in the language of fiction*. Routledge and Kegan Paul.

Carenini, G., Ng, R., and Zhou, X. (2008). Summarizing emails with conversational cohesion and subjectivity. In *Proc. of the Association for Computational Linguistics: Human Language Technologies (ACL-HLT)*.

Fan, R., Chang, K., Hsieh, C., Wang, X., and Lin, C. (2008). Liblinear: A library for large linear classification. *Journal of Machine Learning Research (JMLR)*, 9:1871–1874.

Freund, Y. and Schapire, R. E. (1997). A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, 55(1):119–139.

Gillick, L. and Cox, S. (1989). Some statistical issues in the comparison of speech recognition algorithms. In *Proc. of the Inter. Conf. on Acoustics, Speech, and Signal Processing (ICASSP)*.

Hu, M. and Liu, B. (2004). Mining and summarizing customer reviews. In *Proc. of the ACM SIGKDD international conference on Knowledge discovery and data mining (KDD)*.

Li, B., Liu, Y., Ram, A., Garcia, E. V., and Agichtein, E. (2008). Exploring question subjectivity prediction in community qa. In *Proc. of the Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*.

Lu, B., Tan, C., Cardie, C., and Tsou, B. (2011). Joint bilingual sentiment classification with unlabeled parallel corpora. In *Proc. of the Annual Meeting of the Association for Computational Linguistics (ACL)*.

McCallum, A. and Nigam, K. (1998). A comparison of event models for naive bayes text classification. In *Proc. of AAAI-98 Workshop on Learning for Text Categorization*.

Mihalcea, R. and Banea, C. (2007). Learning multilingual subjective language via cross-lingual projections. In *Proc. of the Annual Meeting of the Association of Computational Linguistics (ACL)*.

Miller, G. A., Leacock, C., Tengi, R., and Bunker, R. T. (1993). A semantic concordance. In *Proc. of the DARPA workshop on Human Language Technology (HLT)*.

Pang, B., Lee, L., Rd, H., and Jose, S. (2002). Thumbs up? sentiment classification using machine learning techniques. In *Proc. of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Riloff, E., Wiebe, J., and Phillips, W. (2005). Exploiting subjectivity classification to improve information extraction. In *Proc. of the National Conference on Artificial intelligence (AAAI)*.

Saffari, A., Leistner, C., Godec, M., and Bischof, H. (2010). Robust multi-view boosting with priors. In *Proc. of the European conference on computer vision (ECCV)*.

Seki, Y., Eguchi, K., Kando, N., and Aono, M. (2005). Multi-document summarization with subjectivity analysis at DUC 2005. In *Proc. of 2005 Document Understanding Conference*.

Somasundaran, S. and Wiebe, J. (2009). Recognizing stances in online debates. In *Proc. of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*.

Somasundaran, S. and Wiebe, J. (2010). Recognizing stances in ideological on-line debates. In *Proc. of the NAACL HLT 2010 Workshop on Computational Approaches to Analysis and Generation of Emotion in Text*.

Wan, X. (2009). Co-training for cross-lingual sentiment classification. In *Proc. of the Annual Meeting of the Association for Computational Linguistics (ACL)*.

Wiebe, J. and Cardie, C. (2005). Annotating expressions of opinions and emotions in language. In *Language Resources and Evaluation*.

Wiebe, J. and Mihalcea, R. (2006). Word sense and subjectivity. In *Proc. of the Joint Conference of the International Committee on Computational Linguistics on Computational Linguistics and the Association for Computational Linguistics (COLING-ACL)*.

Wiebe, J. M. (1994). Tracking point of view in narrative. *Computational Linguistics*, 20(2):233–287.

Xu, Z. and Sun, S. (2010). An algorithm on multi-view adaboost. In *Proc. of the international conference on Neural information processing: theory and algorithms*.

Yu, H. and Hatzivassiloglou, V. (2003). Towards answering opinion questions: separating facts from opinions and identifying the polarity of opinion sentences. In *Proc. of the 2003 conference on Empirical methods in natural language processing (EMNLP)*.

Semi-supervised Representation Learning for Domain Adaptation using Dynamic Dependency Networks

Min Xiao *Yuhong Guo* *Alexander Yates*
Department of Computer and Information Sciences
Temple University, Philadelphia, PA, USA
{minxiao, yuhong, yates}@temple.edu

ABSTRACT
Recently, various unsupervised representation learning approaches have been investigated to produce augmenting features for natural language processing systems in the open-domain learning scenarios. In this paper, we propose a dynamic dependency network model to conduct semi-supervised representation learning. It exploits existing task-specific labels in the source domain in addition to the large amount of unlabeled data from both the source and target domains to produce informative features for NLP tasks. We empirically evaluate the proposed learning technique on the part-of-speech tagging task using Wall Street Journal and MEDLINE sentences and on the syntactic chunking task using Wall Street Journal corpus and Brown corpus. Our experimental results show that the proposed semi-supervised learning model can produce more effective features than unsupervised representation learning methods for open-domain part-of-speech taggers and syntactic chunkers.

KEYWORDS: Domain Adaptation, Representation Learning, POS Tagging, Syntactic Chunking.

Proceedings of COLING 2012: Technical Papers, pages 2867–2882,
COLING 2012, Mumbai, December 2012.

1 Introduction

Existing supervised natural language processing (NLP) systems are highly domain-dependent, whose performance degrades significantly when tested on a new domain. Previous works in a variety of NLP tasks, like part-of-speech (POS) tagging (Blitzer et al., 2006; Huang and Yates, 2010; Blitzer et al., 2011), syntactic chunking (Huang and Yates, 2009; Carreras and Màrquez, 2005), named entity recognition (NER) (Daumé III, 2007; Turian et al., 2010; Daumé III and Marcu, 2006), or parsing (Sekine, 1997; McClosky et al., 2010) show that the performance of supervised NLP systems drops a lot on domains whose vocabulary differs from the vocabulary of the training data.

The major reason that causes the increasing of test error on out-of-domain texts is the traditional representation used in the supervised NLP systems. Most NLP systems use the lexical features for predictions. Though it works very well for various in-domain NLP tasks, they perform poorly when tested on a different domain. There are two main reasons. First, the source and target domains may have very different vocabularies, thus some test words may never appear during the training phase. For example, "sequencing", "metastases" and "genomic" show up frequently as lexical features in biomedical text but rarely in newswire articles. A classifier trained on newswire data thus will have seen few training examples related to sentences with lexical features "sequencing", "metastases" and "genomic" (Ben-David et al., 2010, 2007). Second, the prediction function based on lexical features may change across domains. For example, "signaling" appears in "signaling that ..." from a Wall Street Journal (WSJ) article primarily as a present participle (VBG) (Marcus et al., 1993), but predominantly as a noun in "signaling pathway" from a MEDLINE text (PennBioIE, 2005).

Recently, various unsupervised representation learning techniques are proposed to induce generalizable latent features across domains by exploiting large amount of unlabeled data from both the source and the target domains. Blitzer et al. (2006) and Huang and Yates (2009, 2010) show that their learned representations can yield significant improvements for out-of-domain POS taggers or syntactic chunkers. However, the latent features produced by these unsupervised representation learning techniques provide no task-specific discriminative information over the labels of NLP tasks.

To tackle this issue, in this paper we propose a semi-supervised Dynamic Dependency Network (DDN) model to induce task-specific discriminative latent features across domains. In addition to exploiting large amount of unlabeled data from two domains, the DDN model will also leverage the already-existing task labels from the source domain. It combines the advantages of semi-supervised learning methods from (Blitzer et al., 2006; Daumé III, 2007) with the sequence models from (Huang and Yates, 2009, 2010), while maintaining desirable properties like computational tractability and modeling flexibility to incorporate many features. This model is more appealing than unsupervised representation learning techniques when a target NLP task is known. Moreover, though we perform representation learning in a semi-supervised manner, we only exploit the existing labeled data in the source domain. Thus our model can be applied to arbitrary new domains without any extra annotation effort. The proposed model is empirically evaluated for out-of-domain POS tagging systems on articles from WSJ and MEDLINE, and for out-of-domain syntactic chunking systems on articles from WSJ and Brown corpora. It is shown to outperform unsupervised representation learning techniques. Overall, the contributions of this paper include

- We propose a novel probabilistic graphical model, Dynamic Dependency Networks

(DDNs), which is computationally tractable for inference and training.

- We demonstrate how to apply DDNs on cross-domain semi-supervised representation learning for sequence labeling systems.

- Our empirical results show that DDN-based semi-supervised representation learning is superior to unsupervised representation learning for out-of-domain POS tagging and syntactic chunking.

The remainder of the paper is organized as follows. The next section discusses previous work. Section 3 describes representation learning. Section 4 presents the proposed DDN model and semi-supervised representation learning. Section 5 presents experimental results for out-of-domain POS tagging systems. Section 6 presents empirical results for out-of-domain syntactic chunking systems. We then conclude the paper.

2 Previous Work

Most previous work for domain adaptation tasks has focused on the setting where some labeled data is available in the target domain (Daumé III and Marcu, 2006; Daumé III, 2007; Jiang and Zhai, 2007; Dredze et al., 2010; Daumé III et al., 2010). Daumé III and Marcu (2006) proposed to tackle domain adaptation tasks by training three separate models to distinguish source-specific, target-specific and general information using maximum entropy classifiers. Jiang and Zhai (2007) adopted instance weighting method for semi-supervised domain adaptation by removing misleading training instances in the source domain, assigning more weights to labeled data, and augmenting training data using target instances with predicted labels. Daumé III (2007) proposed to perform supervised domain adaptation with feature augmentation for various NLP tasks. Daumé III et al. (2010) used co-regularization to incorporate unlabeled data for semi-supervised domain adaptation. In contrast, we investigate a more practical setting for domain adaptation where we have no labeled data in the target domain.

Recently, various unsupervised representation learning techniques have been proposed to tackle domain adaptation tasks by exploiting large amount of unlabeled data from two domains (Ando and Zhang, 2005; Blitzer et al., 2006; Huang and Yates, 2009, 2010; Blitzer et al., 2011). Blitzer et al. (2006) proposed a structural correspondence learning (SCL) method to seek for generalizable features by modeling the correlation between pivot features and non-pivot features. Turian et al. (2010) empirically evaluated Collobert and Weston embeddings (Collobert and Weston, 2008), Brown clusters, and HLBL embeddings (Mnih and Hinton, 2009) of words on both syntactic chunking and named entity recognition tasks. Their experimental results demonstrated that those three word representations can improve the performance of out-of-domain named entity recognition systems and in-domain syntactic chunking systems. Huang and Yates (2009) employed Hidden Markov Models (HMMs) to induce hidden states of the sentence words as latent features. Later, Huang and Yates (2010) proposed to learn a multi-dimensional feature representation by simultaneously train multiple HMMs with different initializations. Though unsupervised representation learning achieves good empirical performance for out-of-domain NLP tasks, it underutilizes the source data, since it completely neglects the existing task-specific labels when performing representation learning. The DDN model we propose in this work can suitably address this problem by exploiting task labels when performing semi-supervised representation learning.

3 Representation Learning

A *representation* is a set of features describing instances in a classification problem. Let $\mathbb{X}$ be the set of all instances. For example, for a sequence labeling task in NLP, $\mathbb{X}$ is the set of all sentences. Let $\mathbb{Z}$ be the label set of the classification problem. For POS tagging, $\mathbb{Z}$ is the set of all sequences of part-of-speech tags. For syntactic chunking, $\mathbb{Z}$ is the set of all sequences of syntactic chunks. Let $f : \mathbb{X} \rightarrow \mathbb{Z}$ be the prediction function. A representation is a function $R : \mathbb{X} \rightarrow \mathbb{Y}$, for some suitable feature space $\mathbb{Y}$ (such as $\mathbb{R}^d$). A *domain* $\mathcal{D}$ is defined as a distribution over the instance set $\mathbb{X}$. An open-domain system learns a classification model from a set of training instances $(R(x), f(x))$, where each instance $x \in \mathbb{X}$ is drawn from a *source* domain $\mathcal{D}_s$ and expressed in a representation space defined by function R, and classifies test instances drawn from a separate *target* domain $\mathcal{D}_t$.

It has been shown in recent theoretical work that the performance of domain adaptation greatly depends on the data representation employed, and traditional data representations in NLP prevent learning systems from generalizing appropriately across domains (Ben-David et al., 2010). Previous work by Ben-David et al. (2007) uses Vapnik-Chervonenkis (VC) theory (Vapnik, 1995) to prove theoretical bounds on an open-domain learning machine's performance. It demonstrates that the choice of representation is crucial for domain adaptation. It is customary in VC theory that a good choice of representation must allow a learning machine to achieve low error rates during training.

In light of Ben-David et al.'s theory findings, traditional representations in NLP are inadequate or problematic for domain adaptation. Traditional representations in NLP tasks are lexical features based on local context. Although many previous studies have shown that lexical features allow learning systems to achieve impressively low error rates during training, they also make texts from different domains look very dissimilar and create domain divergence problems. For example, a sentence containing "CEO" may be common in a domain of newswire text but scarce or nonexistent in a different domain like biomedical articles. Likewise, a sentence containing "path-way" is almost certainly from a biomedical literature rather than from a newswire article. Thus with traditional representations of NLP, a prediction model trained in one source domain can hardly work well in a different target domain.

At the same time, traditional representations contribute to *data sparsity*, a lack of sufficient training data for the relevant parameters of the system. In traditional supervised NLP systems, there are parameters for each word type in the data, or perhaps even combinations of word types. Since vocabularies can be extremely large, this leads to an explosion in the number of parameters. As a consequence, for many of their parameters, supervised NLP systems have zero or only a handful of labeled examples. No matter how sophisticated the learning technique, it is difficult to estimate parameters without relevant data. Because vocabularies differ across domains, domain adaptation greatly exacerbates this issue of data sparsity.

Huang and Yates (2009) show how to use language models, HMMs, to induce latent-variable states as generalizable features for various open-domain NLP tasks, such as POS tagging and syntactic chunking. These learned representations have proven to meet the criteria for open-domain representations. It would be difficult to tell two domains apart based on the HMM labels since the same HMM states may generate many similar words from a variety of domains. However, these unsupervised representations are not specifically discriminative for any NLP tasks. This is the main motivation of the research in this paper. Unsupervised representation learning based on HMMs nevertheless serves as one of the comparisons in our experiments.

4 Dynamic Dependency Network for Semi-supervised Representation Learning

In this section, we present a Dynamic Dependency Network (DDN) model to incorporate task-specific label information in the source domain for semi-supervised representation learning. A dynamic dependency network is a dynamic extension of dependency networks (Heckerman et al., 2000) for modeling data with sequential observations and labels. Dependency networks are cyclic directed graphical models. Similar to directed acyclic Bayesian networks, dependency networks allow simple local parameter estimations given fully observed data. But by dropping acyclicity constraints, dependency networks are more flexible on modeling interdependencies between variables than acyclic Bayesian networks. Following the same principle of Dynamic Bayesian Networks (DBNs) (Murphy, 2002), we extend dependency networks into sequential models to form Dynamic Dependency Networks. Although with directed cycles a DDN model will lose the ability of handling time series data that requires time forward directed arcs (not vice versa), it has increased the capacity of modeling word or label interdependencies within local contexts of sentences, comparing to DBNs. Figure 1 demonstrates an example of the DDN models we will use for semi-supervised representation learning.

In this DDN model (Figure 1), the variables are partitioned into three interconnected sequences $X = \{X_1, \ldots, X_T\}$, $Y = \{Y_1, \ldots, Y_T\}$ and $Z = \{Z_1, \ldots, Z_T\}$, representing observations, hidden states and labels respectively. Similar to the HMM model used in (Huang and Yates, 2009), the state sequence is hidden in our model and the state variable Y_t at location t takes values from a predefined set of state values; the observation sequence X is produced from the observed sentence; given Y_t, we assume X_t is conditionally independent of $X_{t'}$ for $t \neq t'$. But in addition to the two layers, X and Y in HMMs, our DDN model adds another task-specific label layer Z. For example, for the POS tagging task, Z will be the sequence of POS tags. Moreover, we take the bi-directional sequential dependency between labels into consideration by connecting each neighbor pairs of labels using bi-directional arcs. At each location t, X_t, Y_t are both parents of Z_t, since we assume both the sentence observation and the hidden state representation determine the sequence label. This DDN model maintains the same inference complexity as the HMM, since only the state sequence Y is latent during training. While by allowing bi-directional arcs over the label sequence Z, it has a natural capacity of modeling and incorporating task-specific label information for representation learning. By incorporating the label sequence Z into the model, we expect to identify more task discriminative latent sequence representations.

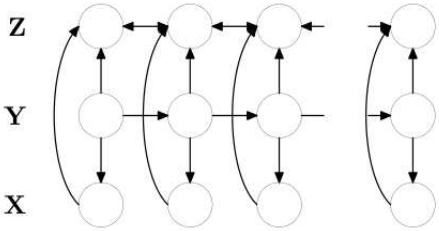


Figure 1: A Dynamic Dependency Network (DDN)

4.1 Training and Inference

Although we have an additional bi-directional Z layer in DDNs, the structures over the hidden layer Y, and between Y and the observed sequence X are similar to in HMMs. Thus inference over the hidden states and parameter learning in DDNs are as tractable as in HMMs. Assume that we are given a data set of N i.i.d. samples, $\{(X^i, Z^i)\}$ for $i = 1, 2, \ldots, N$, where X^i is the ith sentence and Z^i is the corresponding sequence of labels, e.g., POS tags, for X^i. Given the training data, its log-likelihood is

$$L(\theta) = \sum_{i=1}^{N} \log P(X^i, Z^i|\theta)$$

where θ denotes the set of model parameters.

Let $q(Y)$ be any non-zero distribution over hidden variables Y, we can get a lower bound for $L(\theta)$. For notational convenience, we will drop the superscript i in the following formulas.

$$\begin{aligned} \ell(\theta) &= \log \sum_{Y} q(Y) \frac{P(X, Y, Z|\theta)}{q(Y)} && (1) \\ &\geq \sum_{Y} q(Y) \log \frac{P(X, Y, Z|\theta)}{q(Y)} \\ &= L(\theta) - D_{KL}(q(Y) \,\|\, P(Y|X, Z, \theta)) && (2) \end{aligned}$$

where $D_{KL}(\cdot)$ denotes the Kullback-Leibler divergence measure. We denote the objective in (2) as $F(q, \theta)$. We then conduct training by maximizing $F(q, \theta)$ using iterative Expectation-Maximization (EM) algorithm (Baum et al., 1970; Dempster et al., 1977). For the $(k+1)$th iteration, in the E-step, we update q given fixed θ^k from previous iteration by

$$q^{k+1} = \arg\max_{q} F(q, \theta^k) \quad (3)$$

which has the following solution when the KL divergence becomes zero

$$q^{k+1}(Y) = P(Y|X, Z, \theta^k). \quad (4)$$

In the M-Step, we update θ given fixed q^{k+1}

$$\theta^{k+1} = \arg\max_{\theta} F(q^{k+1}, \theta) \quad (5)$$

Similar to HMMs, the parameter estimation in (5) requires computation of $P(Y_{t-1}, Y_t|X, Z)$ and $P(Y_t|X, Z)$ for all t in the E-step. We extend the Baum-Welch algorithm used in HMMs to conduct the required computation with the current model parameters θ. Let $\alpha_t(y) = P(X_1, Z_1, \cdots, X_t, Z_t, Y_t = y|\theta)$ and $\beta_t(y) = P(X_{t+1}, Z_{t+1}, \cdots, X_T, Z_T|Y_t = y, \theta)$. The set of $\{\alpha_t(y)\}$ and $\{\beta_t(y)\}$ can be solved inductively using a forward procedure and a backward procedure respectively, which are analogous to the forward and backward procedures used for HMMs. Then the marginal probabilities can be computed as

$$P(Y_t = y|X, Z, \theta) = \frac{\alpha_t(y)\beta_t(y)}{\sum_{\widehat{y}} \alpha_T(\widehat{y})} \quad (6)$$

$$P(Y_t = y, Y_{t+1} = y\prime|X, Z, \theta) = \tag{7}$$
$$\frac{\alpha_t(y)\beta_{t+1}(y\prime)}{\sum_{\widehat{y}} \alpha_T(\widehat{y})} P(Y_{t+1} = y\prime|Y_t = y) P(Z_{t+1}|Z_t, Z_{t+2}, X_{t+1}, Y_{t+1} = y\prime)$$

The major difference from HMMs is that the computation of (7) requires the additional local probabilities, $P(Z_t|Z_{t-1}, Z_{t+1}, X_t, Y_t)$, in the bi-directional Z sequence. The typical conditional probability table (CPT) parameters for $P(Z_t|Z_{t-1}, Z_{t+1}, X_t, Y_t)$ requires a storage space in the size of $(L-1) \times L^2 \times V \times S$, where L is the number of discrete label values for Z, V is the number of discrete word features for X, and S is the number of discrete states for Y. To reduce the computational cost and memory size for storing such a large CPT and increase the scalability of the proposed model, we exploit a multi-class logistic regression model to model this conditional probability distribution and store the model parameters of the logistic regression model instead.

The logistic regression classifier is trained in the M-step with data collected at each location t, over four types of features $Z_{t-1}, Z_{t+1}, X_t, Y_t$. Given the model parameters θ, the hidden state values of sequence Y are computed using the Viterbi inference algorithm used in HMMs. Thus the trained logistic regression model only requires a model parameter matrix W in the size of $L \times (2L + V + S + 1)$ to calculate the probability $P(Z_t|Z_{t-1}, Z_{t+1}, X_t, Y_t, W)$ for any inputs. The space required to store the W matrix is much smaller than the space required for the original conditional probability table. To avoid overfitting, we trained a $L2$-norm regularized logistic regression model using a second-order Newton method.

With the computed marginal probabilities and induced hidden states, the model parameters θ of the DDN can be re-estimated in a similar way as in HMMs in addition to the retraining of the logistic regression classifier.

4.2 Semi-supervised Representation Learning

We have introduced above how to train DDNs with labeled sentences and conduct inference to induce the hidden states. For cross-domain semi-supervised representation learning, we have a small amount of labeled sentences $\{(X^l, Z^l)\}$ in the source domain and a large amount of unlabeled sentences $\{X^u\}$ in both the source and target domains. This requires the DDN model to handle unlabeled sentences as well. Note in the DDN model we introduced, dropping the label layer Z does not affect either the structure nor the parameter of the other two layers, but simplify a DDN model into a HMM. Thus we can use DDNs as HMMs on unlabeled sentences by sharing common model parameters across labeled and unlabeled sentences. With this semi-supervised representation learning, we expect to inference latent features that are not only generalizable in different domains, but also more informative or discriminative about the target task labels.

Our overall system follows a similar procedure of (Huang and Yates, 2009). First we train a DDN model over both the labeled sentences in the source domain and the unlabeled sentences in both domains, as we described above. Then we use the trained DDN model to produce latent features (i.e, hidden state values Y) for the training and test sentences using Viterbi inference algorithm. Finally we train a classification model, e.g., CRFs, over the training sentences for the target task, e.g. POS tagging, using the latent features as augmented inputs, and then perform classification on the test sentences. We expect semi-supervised representation learning to help improve out-of-domain prediction performance with more discriminative latent features.

5 Domain Adaptation for Part-of-Speech Tagging

In this section, we report our empirical study on how semi-supervised representation learning can improve out-of-domain part-of-speech tagging accuracy.

5.1 Datasets

We used the same datasets as (Blitzer et al., 2006; Huang and Yates, 2009, 2010). The source domain contains articles from Wall Street Journal (WSJ), with 39,832 manually tagged sentences from sections 02-21 and 100,000 unlabeled sentences from a 1988 subset. The target domain contains bio-medical articles from MEDLINE, with 561 labeled sentences[1] and 100,000 unlabeled sentences. The task is to assign words with one of the POS tags from the Penn Treebank POS tags (Marcus et al., 1993) and two more tags from MEDLINE dataset. Among the tags, two tags cannot be seen in the newswire articles, *HYPH* (hyphens) and *AFX* (common post-modifiers for biomedical entities such as genes). These two tags were introduced because of the importance of hyphenated entities in biomedical text, which are about 1.8% of the words in the 561 labeled sentences.

5.2 Representation Learning

We explored both unsupervised representation learning using HMMs and semi-supervised representation learning using the proposed DDNs. We built a vocabulary with all sentences from the source and target domains. In order to reduce the vocabulary size, we further applied the preprocessing steps used in (Huang and Yates, 2009, 2010): we mapped lower frequency (0-2) words to a single unique identifier and sole-digit words into a single unique identifier in our vocabulary. With these preprocessed sentences, we applied representation learning models (DDNs and HMMs) to derive hidden states as additional features for supervised POS taggers.

We used HMMs to perform unsupervised representation learning on 139,832 newswire sentences and 100,000 unlabeled biomedical sentences following the work (Huang and Yates, 2009). Then we decoded the hidden states for 39,832 newswire sentences (the labeled sentences in the source domain) as well as 561 biomedical sentences (the test sentences in the target domain) as additional features for supervised POS tagging. In the unsupervised representation training, one hyperparameter, the number of hidden states, has to be set. A large number of hidden states would make the model more capable to derive latent features, however, it also needs more memory storage and high computation cost. We used 80 states in our experiments, following (Huang and Yates, 2009), to produce fair comparisons.

We used the proposed DDN model for semi-supervised representation learning on 39,832 labeled and 100,000 unlabeled newswire sentences as well as 100,000 unlabeled biomedical sentences. The labels we used in semi-supervised representation learning are the same labels we will use later to train POS taggers. Thus comparing to unsupervised representation learning, the semi-supervised representation learning does not require additional annotation effort, but makes use of the existing labels in the source domain. For our semi-supervised representation learning, we need to choose two hyperparameters, the number of hidden states and the L2 regularization parameter. We set the former as 80, same as in unsupervised representation learning. Our model is not sensitive to the L2 regularization parameter and we set it as 0.5.

[1] Sentences are manually annotated as part of the Penn BioIE project.

5.3 Part-of-Speech Tagging Accuracy

For supervised POS tagging, the training data contains 39,832 labeled newswire sentences and the test data contains 561 biomedical sentences. The 561 biomedical sentences contain 14,554 tokens, of which 23% are OOV (Out-Of-Vocabulary) tokens. We tested our semi-supervised representation learning using supervised Conditional Random Field (CRF) POS taggers and used a fast-training CRF package developed by Okazaki (2007). The feature set used for the CRF POS tagger is presented in Table 1. Specifically, we extracted unigram features. We also added orthographical features such as suffix (-ing, -ogy, -ed, -ly, -s, -ion, -tion, -ity), as well as capitalization. Orthographical features contribute to improving tagging accuracy for out-of-vocabulary words as is demonstrated by (Lafferty, 2001). In addition, we added the latent states as state features for each word from the learned representations. For example, a sentence like "He is the CEO ." contains 5 words: 4 regular words and a "period". A state feature is learned for each of them.

Table 1: CRF feature set used in our supervised CRF POS taggers. Z_i variables stand for labels to be predicted, W_is represent word tokens. Y_is stand for hidden state values decoded from HMM or DDN models, i.e., the new representation features.

Feature Type	Feature Description
Transition	$Z_i = t$ $Z_i = t$ and $Z_{i-1} = t'$
Word	$W_i = w$ and $Z_i = t$
Orthography	For every s $\in$ {-ing, -ogy, -ed, -s, -ly, -ion, -tion, -ity}, suffix(W_i)=s and $Z_i = t$ W_i is capitalized and $Z_i = t$ W_i has a digit and $Z_i = t$
HMM features	$Z_i = t$ and $Y_i = y$
DDN features	$Z_i = t$ and $Y_i = y$

Our experimental results in term of per-token accuracy with different representation learning methods are presented in Table 2. For all test results reported in this paper, the "*All Words*" results are average accuracies over all words in the test data, the "*OOV Words*" results are average accuracies over only OOV words in the test data that appeared less than 3 times in the training data. We reported the empirical results for the following approaches:

- **Baseline:** the baseline CRF POS-tagger trained without representation learning.
- **ASO:** the Alternating Structural Optimization technique in (Ando and Zhang, 2005).
- **SELF-CRF:** the comparison method using a self-training paradigm. We first train a CRF without representation learning on the training data and apply it on the test data, then retrain it on the training data plus the test data with predicted labels.
- **PLAIN-SEM:** the method based on the representation learning technique using contrastive estimation (Smith and Eisner, 2005). We used the modified version in (Huang and Yates, 2010).

Table 2: Per-token accuracy for out-of-domain words on MEDLINE domain trained with Wall Street Journal articles.

Approaches	All Words	OOV Words
Baseline	88.3%	67.3%
ASO	88.4%	70.9%
SELF-CRF	88.5%	70.4%
PLAIN-SEM	88.5%	69.8%
SCL	88.9%	72.0%
SEM-CRF	90.0%	71.9%
HMM	90.5%	75.2%
DDN	**91.3**%	**76.1**%

- **SCL:** the method based on the representation learning with the Structural Correspondence Learning (SCL) technique, developed by (Blitzer et al., 2006).
- **SEM-CRF:** the method based on the representation learning in (Huang and Yates, 2010).
- **HMM:** the method based on the unsupervised representation learning using HMMs in (Huang and Yates, 2009).
- **DDN:** the method based on the proposed semi-supervised representation learning.

We also investigated how our representation learning benefits supervised POS taggers by varying the number of labeled training sentences from the source domain. For comparison, we considered *Baseline*, *SCL* and *HMM*, since *SCL* and *HMM* work very well among all the other comparison methods. The per-token accuracies on test data are reported in Figure 2.

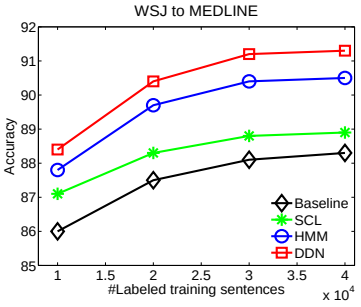


Figure 2: Per-token accuracies for out-of-domain POS tagging. WSJ is used as the source domain and MEDLINE is used as the target domain.

From Table 2 and Figure 2, we can see that with semi-supervised representation learning, DDN consistently outperforms other comparison methods for out-of-domain POS tagging. From Figure 2, we can see that by increasing the number of labeled training data, *DDN* can gain

Table 3: Statistical Significance (McNemar's) tests for out-of-domain experiments with CRF POS taggers. Results are significant with $p < 0.05$.

Null Hypothesis	p-value
HMM vs. Baseline	2.3×10^{-9}
DDN vs. Baseline	3.4×10^{-10}
DDN vs. SCL	6.7×10^{-7}
DDN vs. HMM	2.9×10^{-4}

more improvements in accuracy compared with *Baseline*, *SCL* and *HMM*. Specifically, *DDN* increases accuracy by 2.4% compared with *Baseline*, by 1.3% compared with *SCL*, and by 0.6% compared with *HMM* when the labeled training data is 1,000. When the labeled training data reaches 39,832, *DDN* increases accuracy by 3.0% compared with *Baseline*, by 2.4% compared with *SCL*, and by 0.8% compared with *HMM*. Those results suggest that *DDN* can produce more effective task-specific features by incorporating existing labels from the source domain, and further assist out-of-domain POS tagging.

We also present results for corresponding significance tests over comparisons between *Baseline*, *SCL*, *HMM* and *DDN* in Table 3. We followed the experiments in (Blitzer et al., 2006), and used a McNemar paired test for labeling disagreements (Gillick and Cox, 1989) with $p < 0.05$ being significant on all test words. We report the p values in Table 3. We can see that *DDN* significantly improves out-of-domain tagging accuracy over *Baseline*, *SCL* and *HMM*.

6 Domain Adaptation for Syntactic Chunking

In this section, we empirically study how our proposed semi-supervised representation learning can improve out-of-domain performance on syntactic chunking.

6.1 Datasets

We used the datasets from the CoNLL 2005 shared task (Carreras and Màrquez, 2005) for our second set of experiments on syntactic chunking. We used the standard training set, consisting of sections 02-21 of the Wall Street Journal (WSJ) portion of the Penn Treebank, and conducted tests on the Brown corpus (Kucera and Francis, 1967). The test data contains 3 sections (ck01-ck03) of propbanked Brown corpus data, which consists of 426 sentences containing 7,159 tokens. Besides these labeled data, we also incorporated unlabeled data from both domains. We added 100,000 unlabeled news sentences for the source domain and 57,000 unlabeled sentences for the target domain. In this setting, while the source domain contains newswire text, the test sentences are drawn from the domain of "general fiction" and contain entirely different styles of English.

The original training data and test data from the CoNLL 2005 shared task contain POS tags as well as partial syntax, namely chunks and clauses. In order to perform syntactic chunking task, we mapped the partial syntax labels to chunking labels in IOB2 format. IOB2 format is a standard format for various sequence tasks like syntactic chunking and it is widely used in previous works including the CoNLL 2000 shared task[2]. In IOB2 format, the chunk tags

[2]http://www.clips.ua.ac.be/conll2000/chunking/.

Table 4: Average chunking performance on Brown corpus, with Wall Street Journal articles as training data.

Methods	F1
Baseline	89.93%
SELF-CRF	90.21%
SCL	90.62%
HMM	91.79%
DDN	**93.05**%
UPC Chunker	91.73%

consist of two parts. The first part represents the position of the token in this chunk and the second part stands for the name of the chunk type. For example, the chunking type of *VP* is used for verb phrase words and the chunking type of *NP* is used for noun phrase words. For words forming a chunk of type k, the first word receives the B-k tag (Begin), and the remaining words receive the tag I-k (Inside). Words outside a chunk receive the tag O. Below we give an example of a sentence labeled with chunking tags in IOB2 format from the source domain:

The/B-NP $/I-NP 1.4/I-NP billion/I-NP robot/I-NP spacecraft/I-NP faces/B-VP a/B-NP six-year/I-NP journey/I-NP to/B-VP explore/I-VP Jupiter/B-NP and/O its/B-NP 16/I-NP known/I-NP moons/I-NP ./O

6.2 Representation Learning

We built a vocabulary with all sentences from the source and target domains. In order to reduce the vocabulary size, we used the same preprocessing steps as in POS tagging experiments, mapping lower frequency (0-2) words to a single unique identifier in our vocabulary and sole-digit words into a single unique identifier. On the preprocessed sentences, we then applied representation learning models (DDNs or HMMs) to derive hidden states of the sentence words, which can be used as additional features for supervised syntactic chunking systems.

We used HMMs to perform unsupervised representation learning on 139,832 newswire source sentences and 57,000 unlabeled "general fiction" sentences from Brown corpus. Then we decoded the hidden states for 39,832 labeled newswire sentences and 426 "general fiction" test sentences as additional features for supervised syntactic chunking, using the trained HMM. In the unsupervised representation training, one hyperparameter, the number of hidden states, has to be set. We used 80 states in our experiments in consideration of the model capability, memory storage and computation cost.

We used the proposed DDN model for semi-supervised representation learning on the same data as for HMMs, i.e., 139,832 newswire sentences from the source domain and 57,000 unlabeled "general fiction" sentences from the target domain. But different from unsupervised representation learning, our proposed semi-supervised representation learning makes use of the existing labels of the 39,832 sentences in the source domain. In our semi-supervised representation learning, we need to choose two hyperparameters, the number of hidden states and the L2 regularization parameter. We set the former as 80 and the latter as 0.5, which are same as in our previous experiments for POS-tagging.

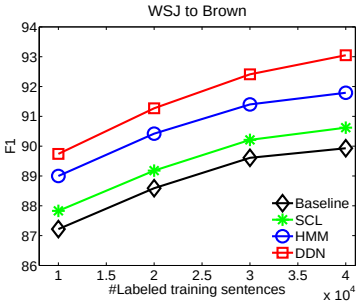


Figure 3: Results in term of F1 measure for out-of-domain syntactic chunking. WSJ is used as the source domain and Brown corpus is used as the target domain.

6.3 Syntactic Chunking Results

For supervised syntactic chunking, the training data contains 39,832 labeled newswire sentences and the test data contains 426 "general fiction" sentences. The 426 "general fiction" sentences contain 7,159 tokens. We tested our semi-supervised representation learning with supervised Conditional Random Field (CRF) syntactic chunking. We used the same fast-training CRF package developed by Okazaki (2007). For syntactic chunking, in addition to the CRF feature set in Table1 which we used in POS tagging experiments, we also extracted POS tag features. All features are represented with boolean values.

Our experimental results with different representation learning methods are presented in Table 4. The results are in term of F1 measure, since F1 measure is widely used in syntactic chunking tasks (Huang and Yates, 2009; Carreras and Màrquez, 2005). We reported the empirical results of the following approaches for comparison:

- **UPC Chunker:** a chunking system based on Voted Perceptrons (Carreras and Màrquez, 2003). Carreras and Màrquez (2005) trained such a chunker on WSJ sections 02-21 and tested it on three sections of the Brown corpus (ck01-03). The reported results serve as the current *state-of-the-art* performance on this experimental setting.
- **Baseline:** the baseline CRF chunker without representation learning.
- **SELF-CRF:** the CRF chunker with a self-training paradigm. We first train a CRF without representation learning on the training data and apply it to the test data, then retrain it on the training data plus the test data with predicted labels.
- **SCL:** the method based on the representation learning produced using the Structural Correspondence Learning (SCL) technique (Blitzer et al., 2006).
- **HMM:** the method based on unsupervised representation learning using Hidden Markov Models (Huang and Yates, 2009).
- **DDN:** the method based on the proposed semi-supervised representation learning.

We also investigated the performance of the proposed DDN-based chunker by varying the

number of labeled training sentences from the source domain. Its comparison results with *Baseline*, *SCL* and *HMM* are presented in Figure 3.

From Table 4 and Figure 3, we can see that with semi-supervised representation learning, the DDN based chunker consistently outperforms other methods for out-of-domain syntactic chunking. According to Figure 3, by increasing the number of labeled training data, *DDN* can gain more improvements in term of F1 measure comparing to *Baseline*, *SCL* and *HMM*. Specifically, *DDN* increases the F1 by 2.52% comparing with *Baseline*, by 1.91% comparing with *SCL*, and by 0.74% comparing with *HMM* when the number of labeled training sentences is 1,000. When the number of labeled training sentences reaches 39,832, *DDN* outperforms *Baseline* by 3.08%, outperforms *SCL* by 2.43%, and outperforms *HMM* by 1.26%, in term of F1-measure. These results again suggest that the semi-supervised representation learning method, *DDN*, can produce more effective task-specific features by incorporating existing labels from the source domain.

We also produced the results of corresponding significance tests, reported in Table 5. We used a McNemar paired test for labeling disagreements (Gillick and Cox, 1989) with $p < 0.05$ being significant on all test words. We reported the p values in Table 5, from which we can see that *DDN* significantly improves out-of-domain chunking performance over *Baseline*, *SCL* and *HMM*.

Table 5: Statistical Significance (McNemar's) tests for out-of-domain experiments with CRF syntactic chunkers. Results are statistical significant with $p < 0.05$.

Null Hypothesis	p-value
HMM vs. Baseline	5.6×10^{-8}
DDN vs. Baseline	2.9×10^{-10}
DDN vs. SCL	7.1×10^{-8}
DDN vs. HMM	4.7×10^{-4}

Conclusion

In this paper, we proposed a Dynamic Dependency Network model for semi-supervised representation learning. In addition to the large amount of unlabeled data from two domains, it incorporates the task-specific labels from the source training data into representation learning. We then used the induced generalizable state features to augment source training sentences and target test sentences for two cross domain NLP tasks: part-of-speech tagging and syntactic chunking. Our empirical studies show that the proposed semi-supervised representation learning outperforms unsupervised representation learning based on HMMs on out-of-domain test data for both POS tagging system and syntactic chunking system. With the proposed semi-supervised representation learning, the POS taggers and the syntactic chunkers resulted also outperform a set of other POS tagging methods and syntactic chunking methods for out-of-domain predictions. All results suggest the proposed semi-supervised representation learning can better bridge the domain gap between training sentences and test sentences by exploiting task-specific label information in the representation learning process.

Acknowledgments

This research was supported in part by NSF grant IIS-1065397.

References

Ando, R. and Zhang, T. (2005). A framework for learning predictive structures from multiple tasks and unlabeled data. *Journal of Machine Learning Research (JMLR)*, 6:1817–1853.

Baum, L., Petrie, T., Soules, G., and Weiss, N. (1970). A maximization technique occurring in the statistical analysis of probabilistic functions of markov chains. *The Annals of Mathematical Statistics*.

Ben-David, S., Blitzer, J., Crammer, K., Kulesza, A., Pereira, F., and Vaughan, J. (2010). A theory of learning from different domains. *Machine Learning*, 79:151–175.

Ben-David, S., Blitzer, J., Crammer, K., and Pereira, F. (2007). Analysis of representations for domain adaptation. In *Advances in Neural Information Processing Systems (NIPS)*.

Blitzer, J., Foster, D., and Kakade, S. (2011). Domain adaptation with coupled subspaces. In *Proceedings of the International Conference on Artificial Intelligence and Statistics (AISTATS)*.

Blitzer, J., McDonald, R., and Pereira, F. (2006). Domain adaptation with structural correspondence learning. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Carreras, X. and Màrquez, L. (2003). Phrase recognition by filtering and ranking with perceptrons. In *Proceedings of the Recent Advances in Natural Language Processing (RANLP)*.

Carreras, X. and Màrquez, L. (2005). Introduction to the conll-2005 shared task: semantic role labeling. In *Proceedings of the Conference on Computational Natural Language Learning (CoNLL)*.

Collobert, R. and Weston, J. (2008). A unified architecture for natural language processing: deep neural networks with multitask learning. In *Proceedings of the International Conference on Machine Learning (ICML)*.

Daumé III, H. (2007). Frustratingly easy domain adaptation. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*.

Daumé III, H. and Marcu, D. (2006). Domain adaptation for statistical classifiers. *Journal of Artificial Intelligence Research*, 26:101–126.

Daumé III, H., Kumar, A., and Saha, A. (2010). Co-regularization based semi-supervised domain adaptation. In *Advances in Neural Information Processing Systems (NIPS)*.

Dempster, A., Laird, N., and Rubin, D. (1977). Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society, Series B*, 39(1):1–38.

Dredze, M., Kulesza, A., and Crammer, K. (2010). Multi-domain learning by confidence-weighted parameter combination. *Machine Learning*, 79(1-2):123–149.

Gillick, L. and Cox, S. (1989). Some statistical issues in the comparison of speech recognition algorithms. In *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*.

Heckerman, D., Chickering, D., Meek, C., Rounthwaite, R., and Kadie, C. (2000). Dependency networks for inference, collaborative filtering and data visualization. *Journal of Machine Learning Research (JMLR)*, 1:49–75.

Huang, F. and Yates, A. (2009). Distributional representations for handling sparsity in supervised sequence labeling. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*.

Huang, F. and Yates, A. (2010). Exploring representation-learning approaches to domain adaptation. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*.

Jiang, J. and Zhai, C. (2007). Instance weighting for domain adaptation in nlp. In *Proceedings of the Annual Meeting of the Association of Computational Linguistics (ACL)*.

Kucera, H. and Francis, W. (1967). *Computational analysis of present-day American English*. Brown University Press.

Lafferty, J. (2001). Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the International Conference on Machine Learning (ICML)*.

Marcus, M., Santorini, B., and Marcinkiewicz, M. (1993). Building a large annotated corpus of english: The penn treebank. In *Proceedings of the Annual Meeting of Association of Computational Linguistics (ACL)*.

McClosky, D., Charniak, E., and Johnson, M. (2010). Automatic domain adaptation for parsing. In *Proceedings of Human Language Technologies: The Annual Conference of the North American Chapter of the Association for Computational Linguistics (ACL)*.

Mnih, A. and Hinton, G. (2009). A scalable hierarchical distributed language model. In *Advances in Neural Information Processing Systems (NIPS)*.

Murphy, K. (2002). *Dynamic Bayesian Networks: Representation, Inference and Learning*. PhD thesis.

Okazaki, N. (2007). CRFsuite: a fast implementation of conditional random fields (CRFs). http://www.chokkan.org/software/crfsuite/.

PennBioIE (2005). Mining the bibliome project. http://bioie.ldc.upenn.edu.

Sekine, S. (1997). The domain dependence of parsing. In *Proceedings of the Conference on Applied natural language processing*.

Smith, N. and Eisner, J. (2005). Contrastive estimation: Training log-linear models on unlabeled data. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*.

Turian, J., Ratinov, L., and Bengio, Y. (2010). Word representations: a simple and general method for semi-supervised learning. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*.

Vapnik, V. (1995). *The nature of statistical learning theory*. Springer-Verlag New York, Inc.

Unsupervised Discriminative Induction of Synchronous Grammar for Machine Translation

Xinyan Xiao[†‡] Deyi Xiong[*] Yang Liu[⋆] Qun Liu[†] Shouxun Lin[†]
[†]Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China
[‡]Youdao Inc., Beijing, China
[*]Human Language Technology, Institute for Infocomm Research, Singapore
[⋆]Department of Computer Sci. and Tech., Tsinghua University, Beijing, China
xiaoxy@rd.netease.com, dyxiong@i2r.a-star.edu.sg,
liuyang2011@tsinghua.edu.cn, liuqun@ict.ac.cn, sxlin@ict.ac.cn

ABSTRACT
We present a global log-linear model for synchronous grammar induction, which is capable of incorporating arbitrary features. The parameters in the model are trained in an unsupervised fashion from parallel sentences without word alignments. To make parameter training tractable, we also propose a novel and efficient cube pruning based synchronous parsing algorithm. Using learned synchronous grammar rules with millions of features that contain rule level, word level and translation boundary information, we significantly outperform a competitive hierarchical phrased-based baseline system by +1.4 BLEU on average on three NIST test sets.

KEYWORDS: synchronous grammar induction, discriminative model, unsupervised learning, machine translation.

Proceedings of COLING 2012: Technical Papers, pages 2883–2898,
COLING 2012, Mumbai, December 2012.

1 Introduction

In the last decade, statistical machine translation (SMT) has been advanced by expanding the basic unit of translation from word to phrase (Koehn et al., 2003) and grammar (Galley et al., 2006; Liu et al., 2006; Chiang, 2007). Most systems induce synchronous grammars (including phrases) from parallel corpora using a heuristic two-step pipeline. This pipeline first aligns a parallel corpus at the word level with heuristic word alignment combination strategies (e.g., grow-diag-final-and) (Koehn et al., 2003), and then extracts translation rules from word-aligned sentence pairs. It is working well in practice and therefore widely adopted. However, such a pipeline artificially brings an undesirable disconnection between translation model and word alignment model (Blunsom et al., 2009; DeNero and Klein, 2010).

Recently, researchers have resorted to principled probabilistic formulations. Various generative models are proposed to learn translation rules directly from sentence pairs without word alignments (Marcu and Wong, 2002; Cherry and Lin, 2007; Zhang et al., 2008; DeNero et al., 2008; Blunsom et al., 2009; Cohn and Blunsom, 2009; Neubig et al., 2011; Levenberg et al., 2012). Due to the independency assumptions in such generative models, it is hard to extend them to incorporate arbitrary features, especially word alignment information as used in the traditional two-step pipeline. As a result, despite theoretical advantages of these models over the two-step pipeline, in practice they can only produce comparable translation quality with the two-step pipeline in some scenarios, and usually even worse results.

Yet another alternative for synchronous grammar induction is unsupervised discriminative model. Unsupervised discriminative model can directly learn synchronous grammars in a theoretically justified manner just like generative model. However, the advantage over generative model is that it is able to easily incorporate word alignment information which has been proved useful in the two-step pipeline.

In this paper, we propose a global log-linear model (Sec. 2) for the induction of synchronous context free grammar (SCFG) (Chiang, 2007). The log-linear model is able to incorporate arbitrary features. Furthermore, it is trained from sentence pairs without word alignments in an unsupervised fashion. In particular:

- We approximate the exact conditional log-likelihood objective inspired by contrastive estimation (Smith and Eisner, 2005) as the optimization of the exact objective is very expensive. The key idea is to estimate parameters via *synchronous hypergraphs* of sentence pairs and *neighbor source hypergraphs* of source sentences (Sec. 3).
- Synchronous parsing is often impractical in large-scale learning applications due to its high complexity $O(n^6)$. We address this challenge by proposing a novel and efficient $O(n^3)$ cube pruning based synchronous parsing algorithm (Sec. 4).
- Aiming to enhance the ability to predict whether a translation derivation is good or not, we incorporate a variety of fine-grained features into our model, including rule level features, word level features and phrase boundary features (Xiong et al., 2010) (Sec. 5).

We evaluate our approach on the NIST Chinese-English translation task. According to the analysis of grammar (Sec. 6.2), our induced grammar is more reusable, and is able to generate better (+8.3 BLEU points) oracle translations than the grammar of the baseline. Meanwhile, in the end-to-end machine translation experiments, our approach outperforms the two-step pipeline by +1.4 BLEU points (Sec. 6.3).

2 Global Log-linear Model

We propose a log-linear model to induce SCFG rules for hierarchical phrase-based translation (Chiang, 2007) which transforms a source sentence $\mathbf{s}$ into a target sentence $\mathbf{t}$ by a sequence of SCFG rules. Such a sequence of rules $\{r\}$ is called a **derivation d**.

As the training data only contains sentence pairs, we model the derivation as a latent variable. The conditional probability $p(\mathbf{t}|\mathbf{s})$ of a target sentence given a source sentence is defined as the sum over all possible derivations $\mathbf{d}$:

$$p(\mathbf{t}|\mathbf{s}) = \sum_{\mathbf{d} \in \triangle(\mathbf{t},\mathbf{s})} p(\mathbf{d},\mathbf{t}|\mathbf{s}) \tag{1}$$

where $\triangle(\mathbf{t},\mathbf{s})$ is the set of all possible derivations that translate $\mathbf{s}$ into $\mathbf{t}$, and $\mathbf{d}$ is one such derivation. Given a source sentence $\mathbf{s}$, the conditional probability of a derivation $\mathbf{d}$ and the corresponding translation $\mathbf{t}$ is:

$$p(\mathbf{d},\mathbf{t}|\mathbf{s}) = \frac{\exp \sum_i \lambda_i H_i(\mathbf{d},\mathbf{t},\mathbf{s})}{Z(\mathbf{s})} \tag{2}$$

where $H_i(\mathbf{d},\mathbf{t},\mathbf{s}) = \sum_{r \in \mathbf{d}} h_i(r,\mathbf{s})$ is feature function. We assume H_i decomposes with derivation $\mathbf{d}$ in terms of local feature function h_i, which is related to a rule r and a source sentence $\mathbf{s}$. λ_i is the correspondent feature weight. $Z(\mathbf{s})$ is the partition function:

$$Z(\mathbf{s}) = \sum_{\mathbf{t}} \sum_{\mathbf{d} \in \triangle(\mathbf{t},\mathbf{s})} \exp \sum_i \lambda_i H_i(\mathbf{d},\mathbf{t},\mathbf{s}) \tag{3}$$

Such a discriminative latent variable model is not new to SMT (Blunsom et al., 2008; Kääriäinen, 2009; Xiao et al., 2011). However, we are distinguished from previous work by applying this model to synchronous grammar induction. The purpose of the latent variable model in such previous work is to do max-translation decoding and training (Blunsom et al., 2008), or to eliminate the gap between heuristic extraction and decoding (Kääriäinen, 2009), instead of grammar induction as synchronous rules are still extracted by the heuristic two-step pipeline. In contrast, our interest lies in using latent variable model to learn synchronous grammar directly from sentence pairs. Overall, to the best of our knowledge, this is the first attempt to apply this log-linear model for synchronous grammar induction.

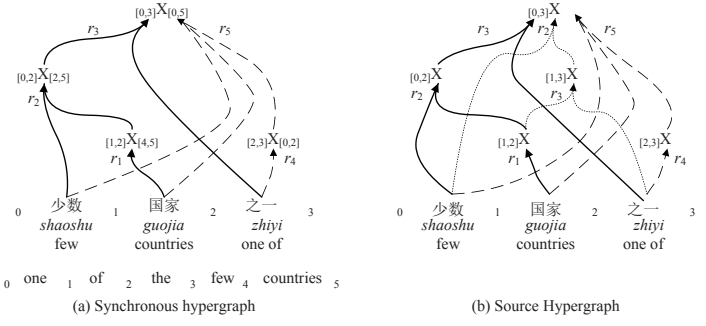
3 Training

We use maximum a posteriori estimator with stochastic gradient descent algorithm for optimization. We maximize the log-likelihood $\mathbb{L}$ of the bilingual corpus $\mathbf{T} = \{(\mathbf{s}^n, \mathbf{t}^n)\}_{n=1}^N$, penalized by a gaussian prior with mean 0 and variance σ (L_2 norm):

$$\mathbb{L} = \sum_{(\mathbf{s}^n,\mathbf{t}^n) \in \mathbf{T}} \log p(\mathbf{t}|\mathbf{s}) + \sum_i \log p_0(\lambda_i) \tag{4}$$

The gradient of the above objective is as follows:

$$\frac{\partial \mathbb{L}}{\partial \lambda_i} = E_{p(\mathbf{d}|\mathbf{t},\mathbf{s})}[H_i] - E_{p(\mathbf{d},\mathbf{t}|\mathbf{s})}[H_i] - \frac{\lambda_i}{\sigma^2} \tag{5}$$



r_1: $\langle$ *guojia* $\Rightarrow$ countries $\rangle$ r_2: $\langle$ *shaoshu* X $\Rightarrow$ the few X $\rangle$
r_3: $\langle$ X *zhiyi* $\Rightarrow$ one of X $\rangle$ r_4: $\langle$ *zhiyi* $\Rightarrow$ one of $\rangle$
r_5: $\langle$ *shaoshu guojia* X $\Rightarrow$ X the few countries $\rangle$

Figure 1: Synchronous hypergraph and source hypergraph of a sentence pair from Chinese segment "*shaoshu guojia zhiyi*" aligned with an English string "one of the few countries". These two hypergraphs are constructed in order to calculate the expectation in Eq. (5). Here, a hyperedge corresponds to an SCFG rule. A node is a bispan in synchronous hypergraph and is a source span in source hypergraph. Notably, the dotted derivation in Figure (b), which contains dotted hyperedges and goes through the source-span [1,3], is a "wrong" derivation, because it results in a translation inconsistent with the target string. Intuitively, we achieve the unsupervised learning by moving the probability of such "wrong" derivations in source hypergraph into the derivations in synchronous hypergraph.

Eq. (5) clearly shows that there are two expectations need to be calculated. The first one is the expectation $E_{p(\mathbf{d}|\mathbf{t},\mathbf{s})}[H_i]$ of a parameter given a sentence pair, and the second one $E_{p(\mathbf{d},\mathbf{t}|\mathbf{s})}[H_i]$ is the expectation of a parameter given the source sentence.

In the following sections, we first introduce how to use hypergraph to compute these expectations by synchronous hypergraph and source hypergraph respectively (Sec. 3.1). Then, we discuss the intractability of the exact training, and achieve tractable training by approximation (Sec. 3.2). Finally, we describe the training algorithm in detail to explain how the rules is induced (Sec. 3.3).

3.1 Inference with Hypergraph

We use hypergraph (Klein and Manning, 2001) to compactly represent the space of derivations. Based on hypergraphs, it's straightforward to calculate the two expectations in Eq. (5). The first expectation $E_{p(\mathbf{d}|\mathbf{t},\mathbf{s})}[H_i]$ is the expected value when observing both source sentence $\mathbf{s}$ and target sentence $\mathbf{t}$. The second expectation $E_{p(\mathbf{d},\mathbf{t}|\mathbf{s})}[H_i]$ is a similar function, but only the source sentence is observed. Thus, in order to calculate the first expectation, we construct a synchronous hypergraph to represent all derivations of a sentence pair. Similarly, for the second expectation, we use a source hypergraph to represent all derivations of a source sentence.

Figure 1 shows a synchronous hypergraph (a) and a source hypergraph (b). Each hyperedge is associated with an SCFG rule. In a synchronous hypergraph, a node is denoted by a nonterminal with a bispan. In contrast, a node in the source hypergraph is a nonterminal that spans

a continuous sequence of words of source sentence.

More formally, a **hypergraph** is a pair $\langle V, E \rangle$, where V is the set of **nodes**, E is the set of **hyperedges**. Each hyperedge $e \in E$ connects a set of antecedent nodes to a single consequent node. And each hyperedge corresponds to an SCFG rule r. In a **synchronous hypergraph**, a node $v \in V$ is in the form $X_{i,j,k,l}$, which denotes the nonterminal X spanning from i to j (that is $s_{i+1}...s_j$) in the source sentence, and from k to l in the target sentence. In a **source hypergraph**, each node $v \in V$ is in the form $X_{i,j}$, which spans from i to j in the source sentence.

Based on these hypergraphs, we compute the two expectations by applying the inside-outside algorithm as described in Li et al. (2009). The computation complexity is linear to the size of hypergraph $O(|E|)$. More exactly, $O(|E|)$ denotes $O(|\mathbf{s}|^3|\mathbf{t}|^3)$ for synchronous hypergraph, and $O(|\mathbf{G}||\mathbf{s}|^3)$ for source hypergraph. Here, $\mathbf{G}$ denotes all potential synchronous grammars.

3.2 Tractable Estimation by Approximation

However, the size of potential SCFGs $\mathbf{G}$ is extremely large given a vocabulary Ω, resulting in a large number of hyperedges in source hypergraph. See the rule r_2 in Figure 1. In reality, there are many potential translation rules that share the same source side "*shaoshu X*" as r_2, but with different target side. Suppose n is the maximum number of terminals in the target side, the number of such rules is up to $O(|\Omega|^n)$. All these rules can connect the source word "*shaoshu*" and the node $X_{1,2}$ to the node $X_{0,2}$, which produces a large number of incoming hyperedges for node $X_{0,2}$. Therefore, due to the large number of potential SCFGs, the number of hyperedges in a source hypergraph is very large. The computation on such a huge source hypergraph makes the exact inference intractable.

To make inference tractable and efficient, we shrink the size of source hypergraph by defining a smaller neighborhood with the synchronous hypergraph. We parse the source hypergraph using the neighbor grammar $\mathbf{NG}$ of synchronous hypergraph $\mathbf{H}$:

$$\mathbf{NG} = \text{Neighbor}(\mathbf{G}', \mathbf{H}) = \{r | r \in \mathbf{G}' \wedge src(r) \in \{src(r') | r' \in G(\mathbf{H})\}\} \quad (6)$$

Here, $\mathbf{G}'$ is the set of translation rules discovered in the training corpus by our algorithm. $src()$ denotes the source side of a rule. $G()$ is the set of rules in a hypergraph. The neighbor grammar contains those rules that are discovered during training (rather than all potential rules), and whose source sides occur in the synchronous hypergraph. Sine the size of $\mathbf{NG}$ is typical fairly small, the parsing of our source hypergraph becomes tractable in practice.

The definition of neighbor source hypergraph is inspired by contrastive estimation (Smith and Eisner, 2005). Similar shrinkage of discriminative neighborhood is also used in Dyer et al. (2011a). Notably, our approximation is consistent with the purpose of synchronous grammar induction for SMT. In SMT, the goal of grammar induction is for translation rather than synchronous parsing. Our source hypergraph corresponds to the potential translation space during SMT decoding. Thus, we expect such approximation to be suitable for SMT.

3.3 Training Algorithm

Based on the synchronous hypergraph and source hypergraph introduced above, we optimize $\mathbb{L}$ in an online style as shown in Algorithm 1. For each sentence pair, we first use cube-pruning based biparsing (Sec. 4) to construct a synchronous hypergraph $\mathbf{H}_1$ (line 4). Rules are discovered in the construction of hypergraph in $\mathbf{H}_1$. We then collect these learnt rules $G(\mathbf{H}_1)$ in $\mathbf{H}_1$,

Algorithm 1: Training($\{(\mathbf{s}^n, \mathbf{t}^n)\}_{n=1}^N$)

1 $\mathbf{G}' \leftarrow \varnothing$, $\lambda \leftarrow \mathbf{0}$
2 **for** $t = 0$ **to** T **do**
3 **for** $n = 0$ *to* N **do**
4 $\mathbf{H}_1 \leftarrow \textsc{BiParse}(\mathbf{s}, \mathbf{t})$ ▷ generate synchronous hypergraph
5 $\mathbf{G}' \leftarrow \mathbf{G}' + G(\mathbf{H}_1)$ ▷ collect grammars in $\mathbf{H}_1$
6 $\mathbf{NG} \leftarrow \textsc{Neighbor}(\mathbf{G}', \mathbf{H}_1)$
7 $\mathbf{H}_2 \leftarrow \textsc{ExhaustiveParse}(\mathbf{NG}, \mathbf{s})$ ▷ generate neighbor source hypergraph
8 $\lambda \leftarrow \lambda + \eta \times \frac{\partial \mathbb{L}}{\partial \lambda}(\mathbf{H}_1, \mathbf{H}_2)$ ▷ η is learning rate
9 **return** $\mathbf{G}'$, λ

and store them in $\mathbf{G}'$. After that, we create the neighbor source hypergraph $\mathbf{H}_2$ by exhaustive bottom-up chart parsing using the neighbor grammar **NG** (line 6). Finally, we calculate the gradient by these two hypergraphs and update the feature weights (line 8). When the algorithm is complete, we learn a grammar $\mathbf{G}'$ and also the feature weights λ of the model.

We implement an stochastic gradient descent (SGD) recommended by Bottou.[1] We schedule the learning rate η by an exponential decay (Tsuruoka et al., 2009). We set the regularization strength, initial learning rate and the base of exponential decay as 1.0, 0.2, 0.9 respectively. We choose these values by maximizing the translation performance measured by BLEU on the NIST 2002 development set with a subset of our training data including 20k sentence pairs.

4 Cube Pruning-based Synchronous Parsing

The approximation makes the training algorithm tractable. However, there is still one problem: how to efficiently construct the synchronous hypergraph? Exhaustive synchronous parsing requires $O(|\mathbf{s}|^3|\mathbf{t}|^3)$ time.[2] To overcome this challenge, we propose a novel and efficient cube-pruning based synchronous parsing algorithm. Given a sentence pair, this algorithm constructs the synchronous hypergraph and discovers rules in the sentence pair.

Instead of enumerating all possible hyperedges, we only create k-best hyperedges for each source span. This is achieved by exploiting the substructure in a hyperedge. For example, the hyperedge in Figure 2(b) contains two substructures: alignment of source sub-span $X_{0,2}$ and alignment of the third source word s_3. We maintain sorted candidate lists for every substructure, and create cubes that represent the potential combinations of these candidates (see Fig. 2(a)). In this way, we are able to create k-best hyperedges by cube pruning.

More specifically, we maintain charts for source words and source spans respectively:

- **s-chart** for each source word. The cell $chart[s, i]$ in s-chart stores a list of candidate alignments for the i-th source word. A candidate alignment is represented by a list of target index. The vertical dimension in Figure 2(a) is an instance of $chart[s, 3]$.

[1] http://leon.bottou.org/projects/sgd

[2] Dyer (2010) has shown that two monolingual parses can be more efficient than one synchronous parse, due to the sparsity of pre-fixed translation rules. Such rules are extracted by the heuristic two-step pipeline. In contrast, there are no pre-fixed translation rules in our case.

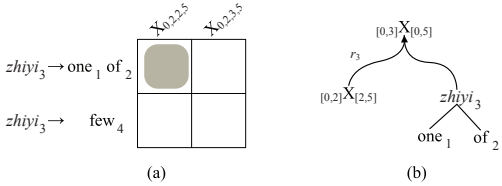


Figure 2: Construct hyperedge from a cube. (a) Cube $X_{0,2}$*zhiyi*$_3$ for source span (0,3). The vertical direction represents the candidate list of *zhiyi*$_3$, while the horizontal direction is a list of nodes that share the same source span (0,2). (b) the hyperedge corresponds to the gray grid in cube (a).

- **X-chart** for each source span. The cell $chart[X,i,j]$ in X-chart stores a list of nodes of the synchronous hypergraph. The nodes in $chart[X,i,j]$ share the same source span (i,j). The horizontal direction in Figure 2(a) is an instance of $chart[X,0,2]$.

These two charts are created by cube-pruning from the bottom up. When processing a source span, we infer all potential source parses and create cubes as shown in Figure 2(a) for every source parses. A cube represents the space of hyperedges for a source parse, where each dimension of a cube denotes the candidate list of a substructure for a hyperedge. Thus, the point in a cube corresponds to a hyperedge (see Fig. 2(b)). Based on these cubes, we create k-best hyperedges, store them into the chart, and then proceed to larger source spans.

4.1 Cube

Formally, a **cube** is a tuple of lists $\mathbf{L} = \langle L_1, ..., L_d, ..., L_{|L|} \rangle$, where $|L|$ is the dimension of the cube and L_d is a list for the d-th dimension. Although $|L|$ can be larger than 3 (a hypercube is defined in that case), we still call it cube for consistency. Given a point $\mathbf{p}$ in the cube, multiplication operator $\otimes$ constructs a hypothesis by $L_1[p_1] \otimes ... \otimes L_{|L|}[p_{|L|}]$. Particularly, there are two types of cubes: *word cube* and *span cube*.

Word Cube A word cube represents the alignment space for a source word s_i, and is used to create the items for $chart[s,i]$. A word cube has $|\mathbf{t}|$ dimensions. The d-th dimension list L_d contains two elements: ε (null translation) and t_d (the d-th target word). The hypothesis of a point $\mathbf{p}$ is a set of target words that aligns to the source word.

Span Cube A span cube represents the space of hyperedges for a source parse of source span (i,j), and is used for creating $chart[X,i,j]$. As a span cube denotes a deduction of the source span, it is represented by a source sequence of symbol γ, which is a mixture of words and nonterminals. For example, the cube in Figure 2(a) is represented by $X_{0,2}$*zhiyi*$_3$. According to γ, the d-th dimension L_d of a span cube $\mathbf{L}$ is defined according to γ_d:

$$L_d = \begin{cases} chart[s,i_1] & \text{if} \quad \gamma_d = s_{i_1} \\ chart[X,i_1,j_1] & \text{if} \quad \gamma_d = X_{i_1,j_1} \end{cases} \tag{7}$$

The hypothesis of a point $\mathbf{p}$ in a span cube corresponds to a hyperedge. The tail nodes of such a hyperedge is the set of nodes $\{v | v \in \{L_d[p_d]\}\}$, where $\{L_d[p_d]\}$ is the set of elements for

Algorithm 2: BiParse($\mathbf{s}, \mathbf{t}$)

```
1  for i ← 1,.., |s| do
2      for j ← 1,.., |t| do                          ▷ create a |t|-dimension word cube
3          L_j ← {ε, t_j}
4      𝒮 ← ⟨L_1, ..., L_|t|⟩
5      chart[s, i] ← MergeProducts(𝒮, ⊗)              ▷ create k-best alignments
6  for h ← 1,.., |s| do                              ▷ h is the size of span
7      forall the i, j s.t. j − i = h do
8          𝒮 ← ∅
9          for γ inferable from chart do             ▷ create all inferable span cubes
10             𝒮 ← 𝒮 + ⟨chart[γ_1], ..., chart[γ_|γ|]⟩
11         chart[X, i, j] ← MergeProducts(𝒮, ⊗)       ▷ create k-best hyperedges
```

the point $\mathbf{p}$ in the cube. The head node of the hyperedge $X_{i,j,k,l}$ has a target span that is the maximum target span covered by $\{L_d[p_d]\}$. See the Figure 2(b), for point (0,0), since the word "*zhiyi*$_3$" aligns to "one$_1$" and "of$_2$", and the tail node is $X_{0,2,2,5}$, the max target coverage is $(0,5)$. Therefore the head node is $X_{0,3,0,5}$.

However, for a point $\mathbf{p}$ in source cube, the elements in each dimension $\{L_d[p_d]\}$ may conflict with each other. See Figure 2(a). Point (0,1) in the cube denotes two elements: an alignment for "*zhiyi*$_3$" to "*few*$_4$" and a node $X_{0,2,2,5}$. The target alignment of "*zhiyi*$_3$" overlaps with the target span of node $X_{0,2,2,5}$. We say this is a *conflict*. Such a conflict denotes an inconsistent alignment. Therefore, we skip those conflicting points in a cube when constructing hyperedge.

Notably, a point in a span cube clearly expresses the inside word alignments of a hyperedge, allowing us to encode word level features for hyperedges.

4.2 Biparsing Algorithm

Algorithm 2 shows the main process of our cube-pruning based biparsing algorithm.

The algorithm begins with the construction of s-chart (lines 1-5). We first construct a $|\mathbf{t}|$-dimension word cube for every source word (lines 2-3). Secondly, by MergeProducts function as described in Chiang (2007), we create k-best alignments for every source word, and add them to the $chart[s,i]$ (line 5).

After that, we create each cell $chart[X,i,j]$ for X-chart (lines 6-11) from the bottom up. For a source span (i,j), all span cubes inferable from s-chart and X-chart are established. Every span cube is a tuple of $|\gamma|$-lists according to Eq. (7), and is append to $\mathscr{L}$ (lines 9,10). After creating all cubes, we construct k-best hyperedges and store them into $chart[X,i,j]$ (line 11).

Obviously, hyperedges and nodes have been created during the construction of the two charts, which means Algorithm 2 produces a synchronous hypergraph. As each hyperedge corresponds to an SCFG rule, Algorithm 2 is also a procedure to discover SCFG rules. On the other hand, the complexity of our biparsing algorithm is $O(|\mathbf{s}||\mathbf{t}| + |\mathbf{s}|k\log k + |\mathbf{s}|^3 + |\mathbf{s}|^2 k\log k)$. Considering k is prefixed and $|\mathbf{t}|$ is linear to $|\mathbf{s}|$ in reality, the complexity can be simplified into $O(|\mathbf{s}|^3)$.

4.3 Scoring Function and Future Cost

The hypothesis, that is created by the operator $\otimes$, corresponds to a hyperedge e and an SCFG rule r. Suppose the head node of a hyperedge e is $X_{i,j,k,l}$, we calculate the score of e by:

$$S(e) = \sum_i \lambda_i h_i(r, \mathbf{t}, \mathbf{s}) + F(i,j,k,l) \tag{8}$$

The first part is the feature score, while the second part $F()$ denotes the future cost. This future cost is an approximation of the outside score of bispan $[i,j,k,l]$, which denotes the cost to generate strings outside the head node $X[i,j,k,l]$. We estimate the future cost by using word level features as follows:

$$F(i,j,k,l) = \sum_{g \notin (i,j)} \operatorname*{argmax}_{h \notin (k,l)} S(s_g, t_h) + \sum_{h \notin (k,l)} \operatorname*{argmax}_{g \notin (i,j)} S(s_g, t_h) \tag{9}$$

$S(s_g, t_h)$ calculates the score of features for a word pair. The future cost can be calculated before inference.

Notably, a rule is not factorized into hypothesis in each dimension of a cube. Because rule features are used in our model, the multiplication operator $\otimes$ is only approximately monotonic under this scoring function. Thus, the MERGEPRODUCTS only produces approximate k-best hypotheses.

4.4 Implementation

Here, we discuss some engineering details of our algorithm.

For efficiency, we only construct 30-best hyperedges for each cube. These hyperedges are recombined, and the top-10 nodes are saved to the chart.

We prune the SCFG rules (corresponding to hyperedge) that are already discovered from synchronous hypergraphs using two constraints. Firstly, we prune a hypergraph by viterbi pruning with $p = 1$ (Huang, 2008). Secondly, we maintain a rule cache with size limit of 20M. When the cache is full, we drop those rules containing nonterminals that are discovered in only one sentence pair. For rules sharing the same source side, we retain the 50 most frequent ones.

Sometimes, a hyperedge produced by the cube corresponds to an illegal SCFG rule as defined in Chiang (2007). For example, rules without source terminals, target terminals and ITG rules (Dekai, 1997). We allow such edges to be added to the hypergraph, in order to make sure that most sentences are reachable. By this setting, the failure of biparsing only occurs in less than 1% of the whole training corpus. However, due to the ambiguity of these rules, we do not use them during end-to-end translation.

Overall, by these implementations, our synchronous algorithm (Alg. 2) runs in a speed of 0.41 second/sentece-pair.

5 Features

The log-linear model allows us to incorporate arbitrary features. In addition to the rules that we are interested in, we also incorporate word-level information and translation boundary information, both of which have proved useful for SMT. Through these features, we can enhance the ability of our model to predict whether a translation derivation is good or not. Notably, previous work typically use such information in locally normalization, while we learn weights of these features by globally normalization on the entire sentence structure.

Rule features We associate each rule with a single feature. Each rule feature counts the number of times that a rule appears in a derivation. In this way, we are able to learn a weight for every rule by global normalization. This is quite different from traditional pipeline that gives each rule with relative frequency that is locally normalized by rules with the same source side or target side.

Word association features As shown in section 4, our inference algorithm also induces inside word alignments for each derivation. Thus, it is straightforward to introduce features that contain word level information. Here, we associate each word pair with a fine-grained boolean feature. We also include the two lexical weights as described by Koehn et al. (2003), estimated by word translation probabilities output by GIZA++. We set the initial weight of these two lexical weights as 1.0. The lexical weights enable our system to score and rank the hyperedges at the beginning. Although word alignment features are used, our system is neither constrained by prefixed word alignment nor requires heuristic alignment combination strategy.

Phrase Boundary features As shown in Figure 1(b), although the rule sequence of the solid derivation and dotted derivation are exactly the same, only solid derivation produces correct translation. They can be distinguished by taking into account boundary information (Xiong et al., 2010; Dyer et al., 2011b). We design two feature templates here: $BE{:}s_i + 1, s_j$ denotes the bigrams of beginning and ending source words. $PS{:}s_i, s_{j+1}$ means the preceding and succeeding source words. We do not use target word information, since this is non-local for hypergraph, and will largely increase the size of hypergraph.

Length Feature Finally, we also integrate the length feature of target side that is used in traditional SMT system.

6 Experiments

In this section, we present our experiments on Chinese-to-English translation tasks. The experiments are aimed at measuring the quality and effectiveness of grammar induced by our method. We present the performance of our unsupervised discriminate synchronous grammar induction (UDSGI) using two groups of features during decoding. We test the translation performance of UDSGI and the baseline on the same decoder.

Baseline The baseline system is an in-house implementation of hierarchical phrase-based translation system(Chiang, 2007). The grammar is extracted from word-aligned corpus by traditional two-step pipeline. Symmetric word alignments were created by first running GIZA++ (Och and Ney, 2003) in both directions and then applying refinement rule "grow-diag-final-and" (Koehn et al., 2003). The system uses 8 dense features including: forward and backward translation probabilities; forward and backward lexical weights; language model; 3 penalties for word count, extracted rule count, and glue rule count.

UDSGI Dense This configuration uses the same feature set as the baseline. Our log-linear model for grammar induction does not contain the forward and backward translation probabilities. However, we still compute the forward and backward translation probabilities for UDSGI by normalizing the expectation $E_{p(\mathbf{d}|\mathbf{t},\mathbf{s})}[H_i]$ of a rule, when algorithm 1 is complete. The normalization is similar to the traditional estimation of these two probabilities.

	|Source Words|		|Target Words|		Number of rules	
Rule Type	Baseline	UDSGI	Baseline	UDSGI	Baseline	UDSGI
Phrase	3.16	2.39	3.97	2.59	2.0M	6.6M
One-NT	3.20	2.31	3.91	2.51	6.4M	2.9M
Two-NT-Mono	2.62	2.52	3.47	2.81	4.2M	2.7M
Two-NT-Swap	2.55	2.46	3.65	2.87	0.5M	0.8M
All Rule	2.89	2.40	3.77	2.63	13.2M	13.1M

Table 1: Comparison of grammars induced by Baseline and UDSGI. Phrase represents the phrase rule. One-NT and two-NT denotes rules contain one or two nonterminals. Mono and Swap means that the two nonterminals are monotone or swapping in the target side. |Source Words| and |Target Words| denotes the average count of source and target terminals in a rule.

UDSGI Dense+Sparse Our global log-linear model encodes millions of sparse features including rule, word pair and phrase boundary features. During decoding, we also enhance our model by these sparse features. In order to optimize these sparse features with the dense features by minimum error rate training (MERT), we group features of the same type into one coarse "summary feature", and get three such features including: rule, word-pair and phrase-boundary features. In this way, we rescale the weights of the three "summary features" with the 8 dense features by MERT. Such approach is similar to Dyer et al. (2011b).

6.1 Data

We used the NIST evaluation set of 2002 (MT02) as our development set for MERT, and test the translation performance on the NIST 2003-2005 (MT03-05) evaluation sets. Case-insensitive NIST BLEU-4 (Papineni et al., 2002) is used to measure translation performance.

We used a bilingual corpus that contains 200K sentence pairs of up to length 40 from the LDC data. [3] There are 8.8M words in the 200K data. We trained a 5-grams language model by the SRILM toolkit (Stolcke, 2002). The monolingual data for language model training includes the Xinhua portion of the GIGAWORD corpus and the English side of the entire LDC data, which contains 432 million words.

We used MERT (Och, 2003) to optimize the feature weights for decoding by maximizing BLEU. Since the instability of MERT has a substantial impact on results, we follow Clark et al. (2011) to report the average scores of 3 independent runs.

6.2 Grammar Analysis and Oracle Results

The synchronous grammar generated by UDSGI has 13.1 millions rules. The number of these rules is comparable with that of grammar extracted by the traditional pipeline, which has 13.2 millions rules. However, the two grammars are quite different as shown in Table 1. Our grammar is more reusable than the baseline's, because it contains less source words and less target words. Interestingly, even we discard the rules which occur only once (See Sec. 4.4), we still learn 60% more swapping rules with two nonterminals (Two-NT-Swap) than the baseline.

[3]The 200K data comes from the following corpus: LDC2002E18, LDC2003E07, LDC2003E14, LDC2004T07, LDC2005T06 and Hansards portion of LDC2004T08 (part of).

System	MT03	MT04	MT05	Avg.
Baseline	64.96	69.48	65.45	66.63
UDSGI	73.37	77.13	74.29	74.93

Table 2: BLEU-4 scores of oracle translations generated by grammars of Baseline and UDSGI. *Avg.* reports the average score on the three test set.

System	$\|\mathbf{G}'\|$	MT03	MT04	MT05	Avg.
Moses-Chart	45.0M	32.93	34.73	31.24	32.97
Baseline	13.2M	32.36	34.51	31.86	32.91
Baseline-expand	46.2M	33.04	35.13	32.08	33.42
UDSGI Dense	13.1M	33.46	35.43	32.74	33.87
UDSGI Dense+Sparse	13.1M	33.58	36.27	33.05	34.30

Table 3: Evaluation of translation quality in terms of BLEU. *Moses-Chart* is the running of hierarchical phrased-model in Moses. *Baseline-expand* uses a similar extraction constraint to Moses-Chart and the same decoder as Baseline. The improvement of UDSGI over Baseline is statistically significant (Koehn, 2004) ($p < 0.01$).

We can not directly evaluate the quality of grammar, since there is not a golden grammar. However, as grammar is used to generate target translations, it's reasonable to decide the quality of a grammar by testing what best translations it can produce. Therefore, we compare the oracle translation result of the grammar in baseline and the grammar in UDSGI, with four reference translations given.[4] As shown in Table 2, our grammar achieves a much higher oracle BLEU (+8.3 BLEU points) than the baseline grammar. This suggests that the grammar induced by our method is able to generate better translations than the grammar extracted by the traditional two-step pipeline.

6.3 Translation Results

Table 3 compares the translation performance of our approach and the baseline on the test sets. When extracting rules as Chiang (2007), the baseline produces an average BLEU of 32.91. We also run Moses-Chart, the implementation of hierarchical phrased-model in Moses (Koehn et al., 2007), on our data with its default settings. As shown in the table, our Baseline is comparable with Moses-Chart. However, Moses-Chart extracts much more rules, because it uses a different extraction constraint from Chiang (2007). The differences mainly include edges of initial phrases can be unaligned and minimum size of source part of sub-phrases is 2.

We also relax the Baseline by applying these two options, and call such setting as *Baseline-expand*. Baseline-expand outperforms Baseline by +0.51 BLEU with 3.5 times of rules. As our UDSGI method learns a similar size of rules with the Baseline's, we only compare our method with the Baseline in the following sections.

Using the same 8 dense features as used in the baseline system, we obtain an average improve-

[4]We calculate the oracle translation by the traditional decoder using sentence BLEU score as feature function. We use add 0.1 smoothing for the ngram precision, and set the reference length for a source span (i,j) as $\lceil \frac{L_{min} \times (j-i)}{|\mathbf{s}|} \rceil$. Here, L_{min} is the shortest length of references for a sentence.

System	$\|G'\|$	MT03	MT04	MT05	Avg.
Baseline	17.6M	33.83	35.94	33.23	34.33
UDSGI Dense+Sparse	13.5M	35.28	37.43	34.25	35.65

Table 4: Experiment results by integrating the entire LDC corpus.

ment of +0.96 BLEU score on the three test sets over the Baseline. As the difference between Baseline and UDSGI Dense only lies in the grammar, the improvement indicates that the grammar induced by UDSGI does outperform that extracted by the traditional two-step pipeline. When we incorporate the sparse features learnt by our training algorithm, we achieve a further improvement of +0.43 BLEU point. Therefore, our training algorithm is able to learn the useful information encoded by the sparse features for translation.

6.4 Exploiting the Entire LDC Data

Since the previous experiment only runs on a medium-scale corpus, we want to know whether the improvement only comes from poor word alignment performance in the baseline system. Therefore, we want to verify the improvement on large-scale data set. Unfortunately, our approach needs to store all rules in memory, which is impractical for large scale data. Instead of directly training on LDC data (357M words), which is 40-times larger than the medium-scale data, we try to explore these data in a different way.

The baseline still extracts rules from the 200K data, while our approach also learns grammar on 200K data. However, we run GIZA++ on the entire LDC data. The word alignment performance of the baseline is therefore enhanced by using all entire data. We used the word translation probabilities of this large-scale trained GIZA++ to calculate lexical weights in our model during training.

Table 4 shows the results. Not surprisingly, the performance of baseline significantly increases by 1.4 points, due to the improvement of word alignment quality and the larger number of extracted rules. Interestingly, our method also achieves similar improvements with enhanced GIZA++. Still, our approach outperforms the baseline by +1.3 points, which is quite close to the improvement in medium-scale corpus experiments. Furthermore, the grammar in UDSGI is smaller than the grammar of the baseline (76.7%). Therefore, we believe that our improvement comes from the theoretical advantage of our approach, and that such advantage does exist even on large-scale data.

7 Related Work

Because of the great importance of synchronous grammars to SMT systems, researchers have proposed various methods in order to improve the quality of grammars. In addition to the generative model introduced in Section 1, researchers also have made efforts on word alignment and grammar rescoring.

The first effort is to improve word alignment by considering phrase/syntax information (May and Knight, 2007; DeNero and Klein, 2010; Pauls et al., 2010; Burkett et al., 2010; Riesa et al., 2011). Such approaches also use discriminative framework to combine word alignment and syntactic alignment information. In this way, they prefer word alignments that are consistent with syntactic structure alignments. However, labeled word alignment data are required in order to learn the discriminative model.

Researchers also try to rescore the weights of translation rules. They rescore the weights of rules extracted from the two-step pipeline, by using the similar latent log-linear model as us (Blunsom et al., 2008; Kääriäinen, 2009), or incorporating various features using labeled data (Huang and Xiang, 2010). Such methods still need to run the heuristic two-step pipeline to extract the grammar, while our method can directly learn the grammar and correspondent weights.

Our work also has a connection to the research direction that exploits resources-rich languages to construct similar tools for resource-poor languages. This can be done with parallel data (Pauls et al., 2010) or without parallel data (Cohen et al., 2011).

Saers et al. (2009) also propose a cubic biparsing algorithm based on beam pruning. They apply this algorithm for generative model-based ITG grammar induction.

Conclusion and Future Work

We have presented a global log-linear model for synchronous grammar induction, and have also proposed efficient training and inference algorithms. In addition to the theoretical advantage, we also achieve significant improvements over the traditional heuristic two-stage pipeline on both medium-scale and large-scale training data.

In the future, we hope to find efficient algorithms that are capable of incorporating contextual features, especially language model and context-based translation model, and optimizing parameters by maximizing BLEU. Because our proposed model is quite general, we are also interested in applying this method to linguistically syntax-based SMT.

Acknowledgments

The authors were supported by High-Technology R&D Program (863) Project No. 2011AA01A207 and No. 2012AA011102, and NSFC Project No. 60903138 and No. 61202216. We would like to thank Yi Lin and the anonymous reviewers for their insightful comments.

References

Blunsom, P., Cohn, T., Dyer, C., and Osborne, M. (2009). A gibbs sampler for phrasal synchronous grammar induction. In *In Proc. ACL 2009*.

Blunsom, P., Cohn, T., and Osborne, M. (2008). A discriminative latent variable model for statistical machine translation. In *Proc of ACL-08*.

Burkett, D., Blitzer, J., and Klein, D. (2010). Joint parsing and alignment with weakly synchronized grammars. In *Proc. NAACL 2010*.

Cherry, C. and Lin, D. (2007). Inversion transduction grammar for joint phrasal translation modeling. In *Proc. SSST 2007, NAACL-HLT Workshop on Syntax and Structure in Statistical Translation*.

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

Clark, J. H., Dyer, C., Lavie, A., and Smith, N. A. (2011). Better hypothesis testing for statistical machine translation: Controlling for optimizer instability. In *Proc. 2011*, pages 176–181.

Cohen, S. B., Das, D., and Smith, N. A. (2011). Unsupervised structure prediction with non-parallel multilingual guidance. In *Proc. EMNLP 2011*.

Cohn, T. and Blunsom, P. (2009). A Bayesian model of syntax-directed tree to string grammar induction. In *Proc. EMNLP 2009*.

Dekai, W. (1997). Stochastic inversion transduction grammars and bilingual parsing of parallel corpora. *Computational Linguistics*, 23(3):377–404.

DeNero, J., Bouchard-Côté, A., and Klein, D. (2008). Sampling alignment structure under a Bayesian translation model. In *Proc. EMNLP 2008*.

DeNero, J. and Klein, D. (2010). Discriminative modeling of extraction sets for machine translation. In *Proc. ACL210*.

Dyer, C. (2010). Two monolingual parses are better than one (synchronous parse). In *Proc. NAACL 2011*.

Dyer, C., Clark, J. H., Lavie, A., and Smith, N. A. (2011a). Unsupervised word alignment with arbitrary features. In *Proc. ACL2011*.

Dyer, C., Gimpel, K., Clark, J. H., and Smith, N. A. (2011b). The cmu-ark german-english translation system. In *Proc. WMT2011*.

Galley, M., Graehl, J., Knight, K., Marcu, D., DeNeefe, S., Wang, W., and Thayer, I. (2006). Scalable inference and training of context-rich syntactic translation models. In *Proc. ACL 2006*.

Huang, F. and Xiang, B. (2010). Feature-rich discriminative phrase rescoring for smt. In *Proc. Coling 2010*.

Huang, L. (2008). Forest reranking: Discriminative parsing with non-local features. In *Proceedings of ACL-08*.

Kääriäinen, M. (2009). Sinuhe – statistical machine translation using a globally trained conditional exponential family translation model. In *Proc. EMNLP 2009*.

Klein, D. and Manning, C. D. (2001). Parsing and hypergraphs. In *Proc. IWPT 2001*.

Koehn, P. (2004). Statistical significance tests for machine translation evaluation. In *Proc. EMNLP 2004*.

Koehn, P., Hoang, H., Birch, A., Callison-Burch, C., Federico, M., Bertoldi, N., Cowan, B., Shen, W., Moran, C., Zens, R., Dyer, C., Bojar, O., Constantin, A., and Herbst, E. (2007). Moses: Open source toolkit for statistical machine translation. In *Proc. ACL 2007 (demonstration session)*.

Koehn, P., Och, F. J., and Marcu, D. (2003). Statistical phrase-based translation. In *Proc. HLT-NAACL 2003*.

Levenberg, A., Dyer, C., and Blunsom, P. (2012). A bayesian model for learning scfgs with discontiguous rules. In *Proc. EMNLP 2012*. Association for Computational Linguistics.

Li, Z., Eisner, J., and Khudanpur, S. (2009). Variational decoding for statistical machine translation. In *Proc. ACL 2009*.

Liu, Y., Liu, Q., and Lin, S. (2006). Tree-to-string alignment template for statistical machine translation. In *Proc. ACL 2006*.

Marcu, D. and Wong, W. (2002). A phrase-based, joint probability model for statistical machine translation. In *Proc. EMNLP 2002*.

May, J. and Knight, K. (2007). Syntactic re-alignment models for machine translation. In *Proc. EMNLP 2007*.

Neubig, G., Watanabe, T., Sumita, E., Mori, S., and Kawahara, T. (2011). An unsupervised model for joint phrase alignment and extraction. In *Proc. ACL 2011*.

Och, F. J. (2003). Minimum error rate training in statistical machine translation. In *Proc. ACL 2003*.

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

Papineni, K., Roukos, S., Ward, T., and Zhu, W.-J. (2002). Bleu: a method for automatic evaluation of machine translation. In *Proc. ACL 2002*.

Pauls, A., Klein, D., Chiang, D., and Knight, K. (2010). Unsupervised syntactic alignment with inversion transduction grammars. In *Proc. NAACL 2010*.

Riesa, J., Irvine, A., and Marcu, D. (2011). Feature-rich language-independent syntax-based alignment for statistical machine translation. In *Proc. EMNLP 2011*.

Saers, M., Nivre, J., and Wu, D. (2009). Learning stochastic bracketing inversion transduction grammars with a cubic time biparsing algorithm. In *Proc. IWPT 2009*.

Smith, N. A. and Eisner, J. (2005). Contrastive estimation: Training log-linear models on unlabeled data. In *Proc. ACL 2005*.

Stolcke, A. (2002). Srilm – an extensible language modeling toolkit.

Tsuruoka, Y., Tsujii, J., and Ananiadou, S. (2009). Stochastic gradient descent training for l1-regularized log-linear models with cumulative penalty. In *Proc. ACL 2009*.

Xiao, X., Liu, Y., Liu, Q., and Lin, S. (2011). Fast generation of translation forest for large-scale smt discriminative training. In *Proc. EMNLP 2011*. Association for Computational Linguistics.

Xiong, D., Zhang, M., and Li, H. (2010). Learning translation boundaries for phrase-based decoding. In *Proc. NAACL2010*.

Zhang, H., Quirk, C., Moore, R. C., and Gildea, D. (2008). Bayesian learning of non-compositional phrases with synchronous parsing. In *Proc. ACL 2008*.

Paraphrasing for Style

Wei Xu[1] *Alan Ritter*[2] *William B. Dolan*[3] *Ralph Grishman*[1] *Colin Cherry*[4]
(1) New York University
(2) University of Washington
(3) Microsoft Research
(4) National Research Council, Canada
xuwei@cims.nyu.edu, aritter@cs.washington.edu, billdol@microsoft.com,
grishman@cs.nyu.edu, Colin.Cherry@nrc-cnrc.gc.ca

ABSTRACT
We present initial investigation into the task of paraphrasing language while targeting a particular writing style. The plays of William Shakespeare and their modern translations are used as a testbed for evaluating paraphrase systems targeting a specific style of writing. We show that even with a relatively small amount of parallel training data, it is possible to learn paraphrase models which capture stylistic phenomena, and these models outperform baselines based on dictionaries and out-of-domain parallel text. In addition we present an initial investigation into automatic evaluation metrics for paraphrasing writing style. To the best of our knowledge this is the first work to investigate the task of paraphrasing text with the goal of targeting a specific style of writing.

KEYWORDS: Paraphrase, Writing Style.

Proceedings of COLING 2012: Technical Papers, pages 2899–2914,
COLING 2012, Mumbai, December 2012.

1 Introduction

The same meaning can be expressed or *paraphrased* in many different ways; automatically detecting or generating different expressions with the same meaning is fundamental to many natural language understanding tasks(Giampiccolo et al., 2007), so much previous work has investigated methods for automatic paraphrasing(Madnani and Dorr, 2010).

Paraphrases can differ along many dimensions, including utterance length, diction level, and speech register. There is a significant literature in sentence compression aimed at modeling the first of these, length: producing meaning-preserving alternations that reduce the length of the input string (Chandrasekar et al., 1996; Vanderwende et al., 2007; Clarke and Lapata, 2008; Cohn and Lapata, 2009; Yatskar et al., 2010). However, we know of no previous work aimed at modeling meaning-preserving transformations that systematically transform the register or style of an input string. Can we learn to reliably map from one form of language to another, transforming formal prose into a more colloquial form, or a casual email into a more formal equivalent?

Systems capable of paraphrasing text targeting a specific writing style could be useful for a variety of applications. For example, they could:

1. Help authors of technical documents to adhere to appropriate stylistic guidelines.

2. Enable non-experts to better consume technical information, for example by translating legalese or medical jargon into nontechnical English.

3. Benefit educational applications, allowing students to:

   (a) Access *modern English* versions of works by authors they are studying.

   (b) Experiment with writing in the style of an author they are studying.

In this paper, we investigate the task of automatic paraphrasing while targeting a particular writing style, focusing specifically on the style of Early Modern English employed by William Shakespeare. We explored several different methods, all of which rely on techniques from phrase-based MT, but which were trained on different types of parallel monolingual data. The first system was trained on the text of Shakespeare's plays, along with parallel modern English "translations" that were written to help students better understand Shakespeare's work. We also developed several baselines which do not make use of this parallel text and instead rely on manually compiled dictionaries of expressions commonly found in Shakespearean English, or existing corpora of out-of-domain parallel monolingual text.

We evaluate these models both through human judgments and standard evaluation metrics from the Machine Translation (MT) and Paraphrase literature, however no previous work has investigated the ability of automatic evaluation metrics to capture the notion of writing style. We show that previously proposed metrics do not provide the complete picture of a system's performance when the task is to generate paraphrases targeting a specific style of writing. We therefore propose three new metrics for evaluating paraphrases targeting a specific style, and show that these metrics correlate well with human judgments.

corpus	initial size	aligned size	No-Change BLEU
http://nfs.sparknotes.com	31,718	21,079	24.67
http://enotes.com	13,640	10,365	52.30

Table 1: Parallel corpora generated form modern translations of Shakespeare's plays

2 Shakespearean Paraphrasing

We use Shakespeare's plays as a testbed for the task of paraphrasing while targeting a specific writing style. Because these plays are some of the most highly-regarded examples of English literature and are written in a style that is now 400 years out of date, many linguistic resources are available to help modern readers pick their way through these Elizabethan texts. Among these are "translations" of the plays into colloquial English, as well as dictionaries that provide modern equivalents for archaic words and phrases.

We compare 3 different stylistic paraphrase systems targeting Shakespearean English which rely on different types of linguistic resources. One leverages parallel "translations", another exploits dictionary resources, and a third relies on modern, out-of-domain monolingual parallel data and an in-domain language model.

2.1 Modern Translations

Access to parallel text in the target style allows us to train statistical models that generate paraphrases, and also perform automatic evaluation of semantic adequacy using BLEU, which requires availability of reference translations. For this purpose we scraped modern translations of 17 Shakespeare plays from `http://nfs.sparknotes.com`, and additional translations of 8 of these plays from `http://enotes.com`.

After tokenizing and lowercasing, the plays were sentence aligned (Moore, 2002), producing 21,079 alignments from the 31,718 sentence pairs in the Sparknotes data, and 10,365 sentence pairs from the 13,640 original pairs in the Enotes data. The modern translations from the two sources are qualitatively quite different. The Sparknotes paraphrases tend to differ significantly from the original text, whereas the Enotes translations are much more conservative, making fewer changes. To illustrate these differences empirically and provide an initial paraphrase baseline, we computed BLEU scores of the modern translations against Shakespeare's original text; the Sparknotes paraphrases yield a BLEU score of 24.67, whereas the Enotes paraphrases produce a much higher BLEU of 52.30 reflecting their strong similarity to the original texts. These results are summarized in Table 1.

To generate paraphrases, we applied a typical phrase-based statistical MT pipeline, performing word alignment on the data described in table 1 using GIZA++ (Och and Ney, 2003), then extracting phrase pairs and performing decoding using Moses (Koehn et al., 2007).

For evaluation purposes, the parallel text of one play, Romeo and Juliet, was held out of the training corpus for this system and the baseline systems described in the following section.

2.2 Baselines

Phrase-based translation has been demonstrated as an effective approach to paraphrasing (Quirk et al., 2004; Chen and Dolan, 2011). However, this approach does require the existence

target	source	target	source
ABATE	shorten	AYE	always
CAUTEL	deceit	GLASS	mirror
SUP	have supper	VOICE	vote

Table 2: Example dictionary entries

Smoothed Probability Estimate	target	source
0.0000790755	PERCHANCE	maybe
0.00003691883	PERADVENTURE	maybe
0.00007524298	HAPLY	maybe
0.00007141065	HAPPILY	maybe
total 0.00026264791		

Table 3: Example ngram probabilities in target language

of parallel corpora of aligned phrases and sentences, resources which may not be available for many writing styles that we might wish to target. For this reason we were motivated to investigate alternative approaches in order to help quantify how critical this type of parallel data is for the task of stylistic paraphrasing.

2.2.1 Dictionary Based Paraphrase

Several dictionaries of stylistically representative words of Shakespearean English and their modern equivalents are available on the web. These dictionaries can be used to define a translation model which can be used in combination with a language model as in standard phrase-based MT.

To build a phrase table, we scraped a set of 68,709 phrase/word pairs from `http://www.shakespeareswords.com/`; example dictionary entries are presented in table 2. As described in (Koehn and Knight, 2000), we estimate phrase translation probabilities based on the frequencies of the translation words/phrases in the target language (Shakespearean English). For instance, if we look at the modern English word *maybe*, our dictionary lists 4 possible Shakespearean translations. We obtained the probabilities for each translation according to the n-gram back-off model built from 36 of Shakespeare's plays using the SRILM toolkit (Stolcke, 2002), normalizing the probabilities for each source phrase, for example $p(\text{PERCHANGE}|\text{maybe}) = \frac{0.0000790755}{0.00026264791} = 0.30107035689$. An example is presented in Table 3. This method allows us to estimate reasonable translation probabilities for use in a phrase table, which is used in combination with a language model built from the 36 plays, which are then fed into the Moses decoder (Koehn et al., 2007).

2.2.2 Out of Domain Monolingual Parallel Data

As a final baseline we consider a paraphrase system which is trained on out-of-domain data gathered by asking users of Amazon's Mechanical Turk Service (Snow et al., 2008) to caption the action in short video segments (Chen and Dolan, 2011). We combined a phrase table extracted from this modern, out of domain parallel text, with an in-domain language model consisting of Shakespeare's 36 plays, applying the Moses decoder (Koehn et al., 2007) to find the best

paraphrases. Although this monolingual parallel data does not include text in the target writing style, the in-domain language model does bias the system's output towards Shakespeare's style of writing. We found that performing Minimum Error Rate Training (Och, 2003) using a small set of held out parallel text from Romeo and Juliet was necessary in order to tune the video corpus baseline to generate reasonable paraphrases.

2.3 Comparison Using Existing Automatic Evaluation Metrics

Figure 1 compares a variety of systems targeting Shakespearean English using the previously proposed BLEU (Papineni et al., 2002) and PINC (Chen and Dolan, 2011) automatic evaluation metrics which have been demonstrated to correlate with human judgments on semantic adequacy and lexical dissimilarity with the input. A description of each of the systems compared in this experiment is presented in Table 4. As mentioned in §2.1, the Enotes paraphrases diverge little from the original text, resulting in a BLEU score of 52.3 when compared directly to the original lines from Shakespeare's plays. Because our goal is to produce paraphrases which make more dramatic stylistic changes to the input, in the remainder of this paper, we focus on the Sparknotes data for evaluation.

2.3.1 Discussion

Two main trends are evident in Figure 1. First, notice that all of the systems trained using parallel text achieve higher BLEU scores than the unmodified modern translations. While the dictionary baseline achieves a competitive PINC score, indicating it is making a significant number of changes to the input, its BLEU is lower than that of the modern translations. Secondly, it seems apparent that the systems whose parameters are tuned using Minimum Error Rate Training tend to be more conservative, making fewer changes to the input and thus achieving lower PINC scores, while not improving BLEU on the test data. Finally we note that using the larger target language model seems to yield a slight improvement in BLEU score.

2.4 Examples

Example paraphrases of lines from *Romeo and Juliet* and several Hollywood movies, generated by the top performing system according to BLEU and PINC, are presented in table 5.

3 Human Evaluation

Figure 1 provides some insight into the performance of the various systems, but it is initially unclear how well the BLEU and PINC automatic evaluation metrics perform when applied to paraphrases that target a specific style of writing. BLEU and PINC have previously been shown to have high correlation with human judgments of semantic adequacy and lexical dissimilarity of paraphrase candidates, but the implications of this for the more specialized task of stylistic paraphrasing are unclear.

While BLEU is typically used to measure semantic adequacy, it seems reasonable to assume that it could also be useful for measuring stylistic alternations, since utterances are more likely to contain overlapping ngrams if they are both semantically and stylistically similar. What BLEU cannot tell us, however is what portion of its improvements are due to stylistic similarity or semantic equivalence. For this reason, we were motivated to perform an evaluation based on human judgments of semantic adequacy, lexical dissimilarity and stylistic similarity.

For this purpose, we randomly sampled 100 lines from *Romeo and Juliet*, then two of the authors

System	Description
16and7plays_36LM	Phrase table extracted from all 16 Sparknotes plays and 7 Enotes plays (holding out R&J) and language model built from all 36 of Shakespeare's plays, again excluding R&J. Uses default Moses parameters.
16and7plays_36LM_MERT	Same as 16and7plays_36LM except parameters are tuned using Minimum Error Rate Training (Och, 2003).
16and7plays_16LM	Phrase table is built from both Sparknotes and Enotes data, and Language model is built from the 16 plays with modern translations
16and7plays_16LM_MERT	Same as 16and7plays_16LM except parameters are tuned using MERT.
16plays_36LM	Only Sparknotes modern translations are used. All 36 plays are used to train Shakespearean language model.
16plays_36LM_MERT	Same as 16plays_36LM except parameters are tuned using MERT.
video_corpus_baseline	Paraphrase system combining out of domain parallel text (Chen and Dolan, 2011) with an in-domain language model. Described in detail in §2.2.2.
modern (no change)	No changes are made to the input, modern translations are left unchanged.
Dictionary	Dictionary baseline described in §2.2.1

Table 4: Descriptions of various systems for Shakespearean paraphrase. *Romeo and Juliet* is held out for testing.

annotated each sentence and its Shakespearean translation to indicate semantic adequacy, lexical dissimilarity, stylistic similarity, and overall quality. The aggregate results of the human evaluation are displayed in Figure 2. Agreement between annotators measured using Pearson's ρ is displayed in Table 6.

Based on the human evaluation, it appears that the baseline combining paraphrases collected from Mechanical Turk (Chen and Dolan, 2011) with a Shakespearean language model has the highest semantic adequacy, yet this approach is also fairly conservative in that it makes few changes to the input.

The dictionary baseline, and the paraphrase system trained on parallel modern translations are roughly comparable in terms of the number of changes made to the input, but the system trained on modern translations achieves higher semantic adequacy, while also being rated higher on style and overall.

These results are roughly in line with the automatic metrics presented in Figure 1. However we also see several important trends which are not apparent from the automatic evaluation. Although the video baseline achieves the highest semantic adequacy in the human evaluation,

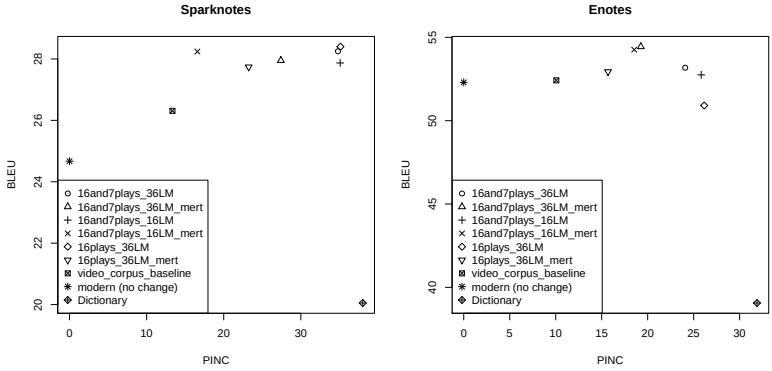


Figure 1: Various Shakespearean paraphrase systems compared using BLEU and PINC. A brief description of each system is presented in table 4.

its BLEU score is significantly lower than 16plays_36LM on the Sparknotes data.[1] It would appear that in this case BLEU is conflating semantic adequacy with writing style. Although the paraphrases produced by the video baseline have high semantic adequacy, their style tends to differ substantially from the reference translations resulting in fewer ngram matches, and thus a lower BLEU score.

4 Automatic Metrics Evaluating Writing Style

While PINC and BLEU do seem useful for automatically evaluating stylistic paraphrases, BLEU tends to conflate the notions of semantic adequacy with writing style. When comparing various systems using automatic metrics, it would seem useful to separate the effects caused by these two distinct criteria. We would like our automatic evaluation metrics to distinguish between a system which generates perfect paraphrases which do not match the target style of writing versus a system which generates sentences in the correct style, but which convey different meaning.

To help address this issue we propose three new automatic evaluation metrics whose goal is to measure the degree to which automatic paraphrases match the target style. These metrics assume existence of large corpora in both the source and target style, but do not require access to any parallel text, or human judgments.

We present a preliminary evaluation of the proposed metrics by measuring their correlation with human judgments, but it should be emphasized that we are only evaluating these metrics with respect to one specific style of writing. We are optimistic that these results will generalize across writing styles, however, since they are based entirely on ngram statistics.

[1] Note that the BLEU score of 16plays_36LM is significantly lower when evaluated on the Enotes data. This makes sense, because the 16 plays come from Sparknotes. This system is not trained on the 7 Enotes plays which, whose modern translations tend to be slightly different in style.

Source	Speaker	Input	Output
Romeo & Juliet	Benvolio	He killed your relative, brave Mercutio, and then young Romeo killed him.	he slew thy kinsman , brave mercutio , and then young romeo kill him .
Romeo & Juliet	Romeo	I can read my own fortune in my misery.	i can read mine own fortune in my woes .
Star Wars	Palpatine	If you will not be turned, you will be destroyed!	if you will not be turn 'd , you will be undone !
Star Wars	Luke	Father, please! Help me!	father , i pray you , help me !
The Matrix	Agent Smith	Good bye, Mr. Anderson.	fare you well , good master anderson .
The Matrix	Morpheus	I'm trying to free your mind, Neo. But I can only show you the door. You're the one that has to walk through it.	i 'll to free your mind , neo. but i can but show you the door. you 're the one that hath to tread it .
Raiders of the Lost Ark	Belloq	Good afternoon, Dr. Jones.	well met , dr. jones .
Raiders of the Lost Ark	Jones	I ought to kill you right now.	i should kill thee straight .

Table 5: Example Shakespearean paraphrases generated by the best overall system.

Semantic Adequacy	Lexical Dissimilarity	Style	Overall
0.73	0.82	0.64	0.62

Table 6: Agreement between annotators measured using Pearson's ρ.

4.1 Cosine Similarity Style Metric

As a first approach to automatic evaluation of writing style, we present a vector-space model of similarity between the system output and a large corpus of text in both the source and target style. The intuition behind this metric is that a large ngram overlap between the system's output and a corpus of text in the target style should indicate that the output is likely to be stylistically appropriate.

More concretely, we extract ngrams from both the source and target corpus which are represented as binary vectors $\vec{s}$, and $\vec{t}$; similarly the output sentence is represented using a vector of ngrams $\vec{o}$. The proposed metric is the normalized cosine similarity between the source and target corpora:

$$S_{\text{Cosine}}(\vec{o}) = \frac{\frac{\vec{o}\cdot\vec{t}}{\|\vec{o}\|\times\|\vec{t}\|}}{\frac{\vec{o}\cdot\vec{t}}{\|\vec{o}\|\times\|\vec{t}\|} + \frac{\vec{o}\cdot\vec{s}}{\|\vec{o}\|\times\|\vec{s}\|}}$$

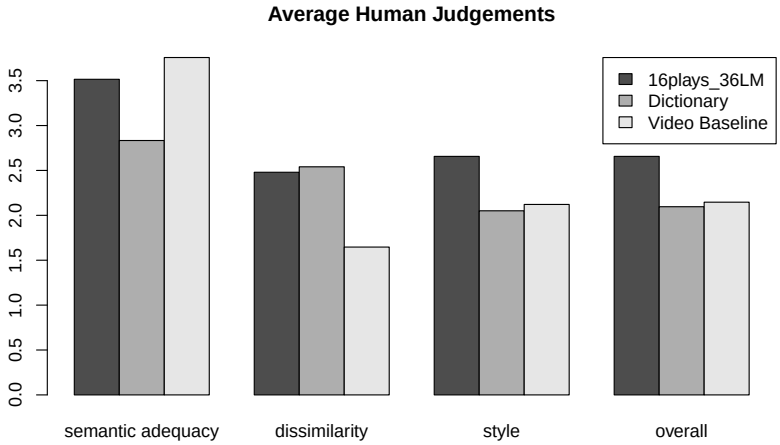


Figure 2: Average human judgments evaluating semantic adequacy, lexical dissimilarity, stylistic similarity, and overall quality of Shakespearean paraphrase systems

4.2 Language Model Style Metric

Another approach is to build a language model from a corpus of text in the target style and a background language model from text outside the style, then apply Bayes' rule to estimate the posterior probability that a sentence was generated from the target language model[2]:

$$\begin{aligned} P(\text{style} = \text{target}|\text{sentence}) &= \frac{P_{\text{LM}}(\text{sentence}|\text{target})P(\text{target})}{P(\text{sentence})} \\ &= \frac{P_{\text{LM}}(\text{sentence}|\text{target}) \times 0.5}{P_{\text{LM}}(\text{sentence}|\text{target}) \times 0.5 + P_{\text{LM}}(\text{sentence}|\text{source}) \times 0.5} \\ &= \frac{P_{\text{LM}}(\text{sentence}|\text{target})}{P_{\text{LM}}(\text{sentence}|\text{target}) + P_{\text{LM}}(\text{sentence}|\text{source})} \end{aligned}$$

4.3 Logistic Regression Style Metric

We also consider an approach to measuring style which is based on logistic regression. Here the idea is to estimate the probability that each sentence belongs to the target style based on the ngrams it contains, using large corpora of *in domain* and *out-of domain* sentences to learn parameters of a logistic regression model.

The probability that a sentence belongs to the target style is estimated as follows:

$$P(\text{style} = \text{target}|\text{sentence}) = \frac{1}{1 + e^{-\left(\vec{\theta} \cdot f(\vec{\text{sentence}})\right)}}$$

[2] Here we assume an uninformative prior, that is $P(\text{source}) = P(\text{target}) = 0.5$.

		ρ (Annotator 1)	ρ (Annotator 2)
semantic adequacy	BLEU	0.35	0.31
dissimilarity	PINC	0.78	0.82
style	BLEU	0.07	0.06
style	PINC	0.20	0.45
style	Cosine	0.37	0.41
style	LM	0.46	0.51
style	Logistic regression	0.47	0.47

Table 7: Correlation between various human judgments and automatic evaluation metrics. Pearson's correlation coefficient is displayed between the automatic metrics and human judgments from each annotator.

Where $f(\vec{\text{sentence}})$ is a vector of ngrams contained by the sentence, and $\vec{\theta}$ is a vector of weights corresponding to each possible ngram.

The parameters, $\vec{\theta}$, are optimized to maximize conditional likelihood on the source and target corpus, where the assumption is that the target corpus is in the target style, whereas the source corpus is not.[3]

4.4 Evaluation

We trained the logistic regression, language model and cosine similarity evaluation metrics using the original Shakespeare plays and modern translations as the source and target corpus respectively, then measured Pearson's Correlation Coefficient between the automatic evaluation metrics and human judgments described in §3. These results are reported in table 7.

As can be seen in table 7, the correlation between semantic adequacy and BLEU appears smaller than that reported in previous work (Chen and Dolan, 2011). Presumably this is due to the conflation of stylistic differences and semantic adequacy discussed in §3. However it also appears that the correlation between BLEU and human style judgments is too low to be of practical use for evaluating style.

PINC, on the other hand has high correlation with judgments on dissimilarity, and is also correlated with human style judgments. We believe PINC has correlation with writing style, because the systems we are evaluating all target Shakespearean English, so whenever changes are made to the input, they are likely to make it similar to the target style. Although PINC has relatively high correlation with human judgments, it is likely not a very useful measure of writing style in practice. For example, consider a paraphrase system which makes many changes to the input and thus gets a high PINC score, but targets a completely different writing style.

Both the language model and logistic regression style metrics achieve the highest overall correlation with human writing style judgments, achieving comparable performance.

We note that overall the automatic metrics tend to agree with human judgments as displayed in Figure 3.[4]

[3] Parameters were optimized using MEGAM `http://www.cs.utah.edu/~hal/megam/`.

[4] Although the automatic style metrics rate the dictionary system higher than the video corpus baseline, both systems have very comparable style scores in the automatic and human evaluations.

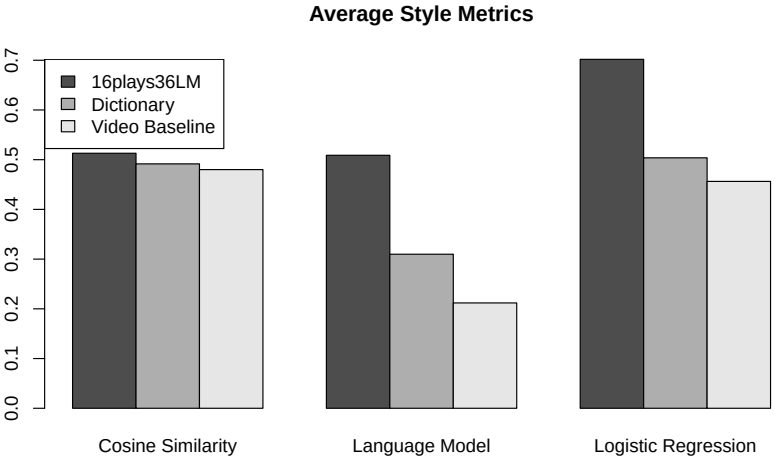


Figure 3: Results comparing the 3 systems using the automatic style metrics.

		ρ (Annotator 1)
semantic adequacy	BLEU	0.27
dissimilarity	PINC	0.79
style	BLEU	0.12
style	PINC	0.41
style	Cosine	0.37
style	LM	0.45
style	Logistic regression	0.46

Table 8: Correlation between human judgments and automatic evaluation metrics when paraphrasing Shakespeare's plays into modern prose.

5 Translating Shakespeare's Plays to Modern English

Finally we perform an evaluation on the task of automatically translating Shakespeare's plays into modern English.

For the purposes of this evaluation, we make use of the same paraphrase systems previously described, but swap the source and target languages. Additionally, each system makes use of a language model constructed from the 16 modern translations, with *Romeo and Juliet* held out for testing. 100 lines from *Romeo and Juliet* were automatically translated into modern English using each system, and the aligned modern translations were used as a reference when computing BLEU. The results of evaluating each of the automatic evaluation metrics on this data are presented in Figure 5, correlation of the automatic metrics with with human judgments are presented in Table 8 and average human judgments are presented in Figure 4.

These results suggest that in comparison to the dictionary and video corpus baselines, our

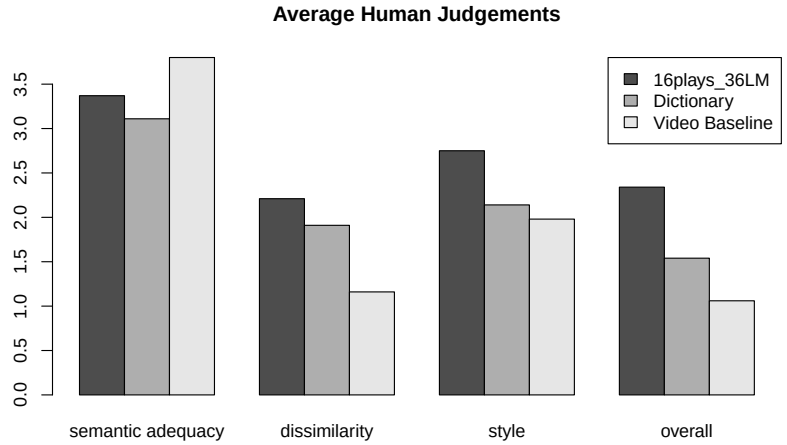


Figure 4: Average human judgments translating Shakespeare's plays into modern English.

system trained on modern translations generates a large number of paraphrases which match the target style. Note that the paraphrase system based on the out-of-domain video corpus makes very few changes to the input, and thus achieves a very low PINC score. This is due to the many out of vocabulary words in Shakespeare's plays which result in very few matching source phrases in the video baseline's phrase table. Several automatic paraphrases into modern English are presented in Table 9.

6 Related Work

Much previous work has addressed the task of automatically generating paraphrases (Barzilay and Lee, 2003; Dolan et al., 2004; Shinyama and Sekine, 2003; Das and Smith, 2009; Bannard and Callison-Burch, 2005; Callison-Burch, 2008; Kok and Brockett, 2010). In addition several authors have previously proposed automatic metrics specifically for evaluating paraphrases (Chen and Dolan, 2011; Callison-Burch et al., 2008; Liu et al., 2010). We are not aware, however, of any work that has addressed the task of generating or evaluating paraphrases targeting a specific style of writing.

Perhaps most relevant, however, is recent work on automatic generation of rhythmic poetry (Greene et al., 2010). This work focuses on automatically generating and translating poetry in an appropriate meter (e.g. iambic pentameter) using finite-state transducers, but does not investigate the task of paraphrase. Their generation system is trained on Shakespeare's sonnets, and they investigate the task of automatically translating Dante's Divine Comedy from Italian to English. While our work does not address the issue of meter, it should be possible to combine our translation models with their weighted finite state transducers to produce Shakespearean paraphrase models which produce output in an appropriate meter.

Finally we highlight related work on authorship classification which can be seen as detecting a specific style of writing (Gamon, 2004; Raghavan et al., 2010). This work has not specifically

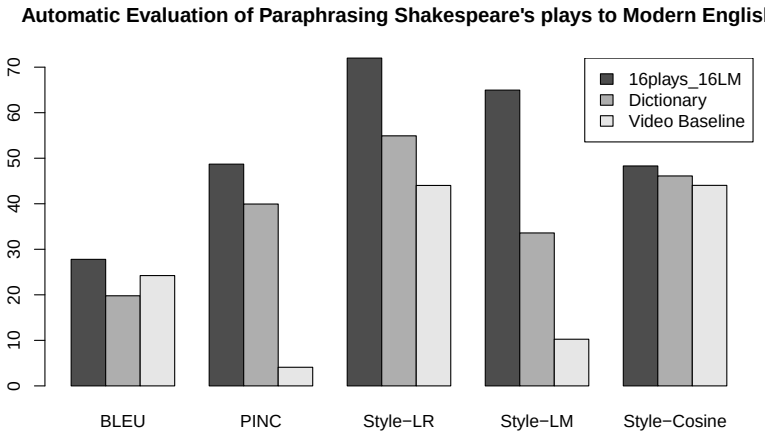


Figure 5: Automatic evaluation of paraphrasing Shakespeare's plays into modern English comparing a system based on parallel text (16plays_16LM), a Dictionary baseline, and a system trained on out of domain parallel monolingual text. Note that the video corpus baseline achieves low overall PINC score, as few phrases in the input match phrases found in its phrase table, resulting in a small number of changes to the input.

Speaker	Input	Output
MERCUTIO	i will bite thee by the ear for that jest .	i ' ll bite you by the ear for that joke .
MONTAGUE	what further woe conspires against mine age ?	what ' s true despair conspires against my old age ?
ROMEO	how doth my lady ?	how is my lady ?
FRIAR LAURENCE	hast thou slain tybalt ?	have you killed tybalt ?
NURSE	an i might live to see thee married once , i have my wish .	if i could live to see you married , i ' ve my wish .
PRINCE	benvolio , who began this bloody fray ?	benvolio , who started this bloody fight itself ?
JULIET	what is your will ?	what do you want ?
LADY CAPULET	call her forth to me .	bring her out to me .

Table 9: Example modern paraphrases of lines from Romeo and Juliet generated using our system.

addressed the task of automatically generating or evaluating paraphrases in a specific style, however.

7 Conclusions

We have presented the first investigation into the task of automatic paraphrasing while targeting a specific writing style. Using Shakespeare's plays and their modern translations as a testbed for this task, we developed a series of paraphrase systems targeting Shakespearean English. We showed that while existing evaluation metrics are useful for evaluating paraphrases in this context, BLEU tends to conflate semantic equivalence with writing style and thus gives an incomplete picture of system performance on these different dimensions.

To address this problem, we introduced three new metrics for evaluating writing style, one based on cosine similarity one based on language models, and the third based on logistic regression. We measured correlation between automatic metrics and human judgments, and showed that our new metrics have better correlation with human judgments than existing metrics in the context of our task. While this evaluation is limited to one specific style of writing, we are optimistic that these or similar metrics will also perform well when evaluating paraphrase systems targeting other writing styles.

We have shown that access to even a small amount of parallel text produces paraphrase systems capable of generating a large number of stylistically appropriate paraphrases while preserving the meaning of the input text. Our paraphrase systems targeting Shakespearean English could be beneficial for educational applications, for example helping to make Shakespeare's work accessible to a broader audience. Future work could investigate stylistic paraphrasing in other domains, such as paraphrasing emails into formal documents, or translating legal documents into nontechnical English.

References

Bannard, C. and Callison-Burch, C. (2005). Paraphrasing with bilingual parallel corpora. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*.

Barzilay, R. and Lee, L. (2003). Learning to paraphrase: an unsupervised approach using multiple-sequence alignment. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology - Volume 1*, NAACL '03.

Callison-Burch, C. (2008). Syntactic constraints on paraphrases extracted from parallel corpora. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*.

Callison-Burch, C., Cohn, T., and Lapata, M. (2008). Parametric: an automatic evaluation metric for paraphrasing. In *Proceedings of the 22nd International Conference on Computational Linguistics - Volume 1*.

Chandrasekar, R., Doran, C., and Srinivas, B. (1996). Motivations and methods for text simplification. In *Proceedings of the 16th conference on Computational linguistics - Volume 2*.

Chen, D. L. and Dolan, W. B. (2011). Collecting highly parallel data for paraphrase evaluation. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics (ACL-2011)*, Portland, OR.

Clarke, J. and Lapata, M. (2008). Global inference for sentence compression: An integer linear programming approach. *Journal of Artificial Intelligence Research*, 31(1):399–429.

Cohn, T. and Lapata, M. (2009). Sentence compression as tree transduction. *Journal of Artificial Intelligence Research*, 34:637–674.

Das, D. and Smith, N. A. (2009). Paraphrase identification as probabilistic quasi-synchronous recognition. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 1 - Volume 1*, ACL '09.

Dolan, B., Quirk, C., and Brockett, C. (2004). Unsupervised construction of large paraphrase corpora: Exploiting massively parallel news sources. In *Proceedings of Coling 2004*.

Gamon, M. (2004). Linguistic correlates of style: authorship classification with deep linguistic analysis features. In *Proceedings of the 20th international conference on Computational Linguistics*.

Giampiccolo, D., Magnini, B., Dagan, I., and Dolan, B. (2007). The third pascal recognizing textual entailment challenge. In *Proceedings of the ACL-PASCAL Workshop on Textual Entailment and Paraphrasing*, RTE '07.

Greene, E., Bodrumlu, T., and Knight, K. (2010). Automatic analysis of rhythmic poetry with applications to generation and translation. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*.

Koehn, P., Hoang, H., Birch, A., Callison-Burch, C., Federico, M., Bertoldi, N., Cowan, B., Shen, W., Moran, C., Zens, R., Dyer, C., Bojar, O., Constantin, A., and Herbst, E. (2007). Moses: open source toolkit for statistical machine translation. In *Proceedings of the 45th Annual Meeting of the ACL on Interactive Poster and Demonstration Sessions*.

Koehn, P. and Knight, K. (2000). Estimating word translation probabilities from unrelated monolingual corpora using the em algorithm. In *Proceedings of the Seventeenth National Conference on Artificial Intelligence and Twelfth Conference on Innovative Applications of Artificial Intelligence*.

Kok, S. and Brockett, C. (2010). Hitting the right paraphrases in good time. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*.

Liu, C., Dahlmeier, D., and Ng, H. T. (2010). PEM: A paraphrase evaluation metric exploiting parallel texts. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*.

Madnani, N. and Dorr, B. (2010). Generating phrasal and sentential paraphrases: A survey of data-driven methods. *Computational Linguistics*, 36(3):341–387.

Moore, R. C. (2002). Fast and accurate sentence alignment of bilingual corpora. In *Proceedings of the 5th Conference of the Association for Machine Translation in the Americas on Machine Translation: From Research to Real Users*, AMTA '02.

Och, F. J. (2003). Minimum error rate training in statistical machine translation. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics - Volume 1*.

Och, F. J. and Ney, H. (2003). A systematic comparison of various statistical alignment models. *Comput. Linguist.*

Papineni, K., Roukos, S., Ward, T., and Zhu, W.-J. (2002). Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*.

Quirk, C., Brockett, C., and Dolan, W. (2004). Monolingual machine translation for paraphrase generation. In *Proceedings of EMNLP 2004*.

Raghavan, S., Kovashka, A., and Mooney, R. (2010). Authorship attribution using probabilistic context-free grammars. In *Proceedings of the ACL 2010 Conference Short Papers*.

Shinyama, Y. and Sekine, S. (2003). Paraphrase acquisition for information extraction. In *Proceedings of the second international workshop on Paraphrasing - Volume 16*, PARAPHRASE '03.

Snow, R., O'Connor, B., Jurafsky, D., and Ng, A. Y. (2008). Cheap and fast—but is it good?: evaluating non-expert annotations for natural language tasks. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '08.

Stolcke, A. (2002). Srilm - an extensible language modeling toolkit. In *INTERSPEECH*.

Vanderwende, L., Suzuki, H., Brockett, C., and Nenkova, A. (2007). Beyond sumbasic: Task-focused summarization with sentence simplification and lexical expansion. *Inf. Process. Manage.*

Yatskar, M., Pang, B., Danescu-Niculescu-Mizil, C., and Lee, L. (2010). For the sake of simplicity: unsupervised extraction of lexical simplifications from wikipedia. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*.

Modeling ESL Word Choice Similarities By Representing Word Intensions and Extensions

Huichao Xue, Rebecca Hwa
Department of Computer Science,
University of Pittsburgh,
210 S Bouquet St, Pittsburgh, PA 15260 USA
hux10@cs.pitt.edu, hwa@cs.pitt.edu

ABSTRACT
Automatic error correction systems for English as a Second Language(ESL) speakers often rely on the use of a *confusion set* to limit the choices of possible correction candidates. Typically, the confusion sets are either manually constructed or extracted from a corpus of manually corrected ESL writings. Both options require the involvement of English teachers. This paper proposes a method to automatically construct confusion sets for commonly used prepositions from non-ESL corpus without manual intervention. The proposed method simulates how ESL learners learn both the intensions and extensions of English words from standard English text. Our experimental results suggest that the automatically constructed confusion sets based on the similarities between the learned words' intensions is competitive with those directly learned from an ESL corpus containing about 150K preposition usages.

TITLE AND ABSTRACT IN ANOTHER LANGUAGE, L_2 (OPTIONAL, AND ON SAME PAGE)

通过分析单词的内涵和外延来对用词混淆建模

针对把英语作为第二语言的人群的自动语法纠错系统，通常会需要使用“混淆集”来限制系统纠错的种类。一般来说，这些混淆集或者是由专家总结经验得出，或者是从被专家纠错过的文字当中提取的。这两种方法都需要英语专家的介入。在这篇论文当中，我们提出了一种无需专家介入，自动建立常用介词混淆集的方法。在此方法中，我们对英语单词的内涵和外延建模，并且模拟了英语学习者们学习单词内涵和外延的过程。实验表明，使用单词内涵之间的相似度来创建的混淆集，与从含15万介词的标注语料当中提取的混淆集质量是相当的。

KEYWORDS: Confusion Sets, Lexical Semantics, Grammatical Error Correction, Distance Metric Learning, English as Second Language, Second Language Acquisition.

KEYWORDS IN L_2: 选词混淆、混淆集、词汇语义学、语法纠错、距离度量学习、英语作为第二语言、第二语言学习.

Proceedings of COLING 2012: Technical Papers, pages 2915–2930,
COLING 2012, Mumbai, December 2012.

1 Introduction

A large portion of the English text (e.g., on the web) is written by people whose native language is not English. Many *English as a Second Language* (ESL) writers, even those with a high level of proficiency, make common grammatical mistakes. Researchers working on Grammar Error Correction (GEC) try to analyze the patterns of these mistakes in order to understand the underlying reasons for their occurrence and to build tools that help ESL writers to correct their errors (Leacock et al., 2010).

Many recently developed GEC systems (Chodorow et al., 2007; J. R. Tetreault & Chodorow, 2008; Gamon et al., 2009; Liu et al., 2010; Rozovskaya & Roth, 2011; Dahlmeier & Ng, 2011a) share a similar infrastructure: first, they isolate some specific types of errors (e.g., preposition errors, article errors, or word choice errors); then, they propose a correction for each instance by treating it as a classification problem. To cast the correction problem as a classification problem, the system has to know, a priori, what are the set of possible corrections for an error. That is, the system needs to pre-define a *confusion set* for each error type.

Previous work has shown the importance of the role of confusion sets. However, the construction of confusion sets requires a great deal of human involvement. English teachers are involved in Liu et al. (2010) to manually filter the initial large verb confusion sets; Rozovskaya & Roth (2010a) used annotated ESL corpus to limit their confusion sets for prepositions. They have shown that even for closed word classes such as prepositions, limiting the confusion sets help simplify the classifiers' tasks and finally lead to both a better precision and recall.

In this paper, we propose a method to automatically construct confusion sets without manual intervention or an annotated ESL corpus. Our approach is to model and simulate how ESL learners might learn words from reading English text. In the process of mastering the language, the learners are often confused about how to choose between similar words. Our goal in this work is to build a model that analyzes which words might appear similar to each other to an ESL learner and then builds up confusion sets with those words. The work presented in this paper addresses learning frequently used *prepositions*, but the idea may be generalized to open word classes.

Our simulation focuses on two main aspects of learning new words: learning their intensions and extensions. The intension of a word is often implied by its definition and its relations to other words; the extension of a word is often characterized by its usages[1]. Ultimately, ESL learners need to achieve a compatible understanding of both the word's intensions and extensions; but before that happens, they may confuse words that have either similar intensions or extensions. Our proposed model applies an algorithm called Relevance Component Analysis (Bar-Hillel et al., 2006) to describe how an ESL learner might organize the extensional representations of words onto an intensional space. We then build up confusion sets with words that have similar intensions.

We compare our model against two models that simulate how learners obtain words' intensions and extensions separately. Under the intensions-only model, word choice confusions are directly measured by the semantic similarity between words. Under the extensions-only model, word choice confusions are attributed to the learner not having completely mastered a word's usages; it can be seen as a faulty language model. In our experiments, we found that, by considering the

[1]We use the terms *intension* and *extension* following the definitions from from Linguistics literature(see, for example, Chalmers (2002)).

interaction of word intensions and extensions, our proposed model produces better confusion sets than those which consider them separately; moreover, the resulting confusion sets are competitive with those directly learned from an error-annotated ESL corpus containing 150K preposition usages.

2 Background

The mistakes made by ESL writers are not random. In their studies, Rozovskaya & Roth (2011) find that those who share the same native language tend to make similar types of mistakes. The natural question that arises is: what are the underlying causes for the mistakes? In the frame of computational linguistics research, the question might be rephrased as: Can we build a mathematical model that simulates ESL writing mistakes?

A model that builds a table of **confusion sets** whose distributions correlate well with the mistakes made by ESL writers is an important component in simulating ESL writing. For instance, Brockett et al. (2006) simulates an ESL corpus according to a set of manually constructed rules, which would not be available until confusion sets are established.

In addition to aiding our understanding of the underlying causes of ESL writing mistakes, confusion sets also have useful practical applications. Generally speaking, reducing the confusion set helps lead the classifiers in the GEC system to a better performance by prohibiting them from considering the outcomes that are both *unlikely* and *misleading*. For example, although ESL learners normally would not confuse *within* with *in*, classifiers may have difficulties telling them apart. Therefore, eliminating *within* from *in*'s confusion set may help the classifier. Generally speaking, by reducing the confusion set's size to rule out these outcomes, although the systems will be disabled from correcting certain types of mistakes, they will often increase the accuracies on more prevalent error types and finally lead to a better *overall* performance. In the past, Rozovskaya & Roth (2010a) showed that by limiting the size of the confusion set for prepositions, their GEC system's performance improved.

One challenge in building a model of confusion sets is that automatic methods typically generate huge lists of words, given the many possible factors that contribute to confound ESL writers. For instance, Dahlmeier & Ng (2011a) observed that ESL collocation errors may be due to similarities of the words' spellings, pronunciations, synonyms, and paraphrases in the writer's native language (L1). However, by including all words that are similar according to any of these factors, one would end up with a large confusion set which introduces difficulties for the classification tasks down the GEC pipeline.

A possible solution is to ask human experts using their knowledge about ESL mistakes to restrict the confusion set. This is the approach taken by Liu et al. (2010) for their GEC system for verb selection. Another alternative is to make use of an ESL corpus in which the mistakes have been corrected by an English teacher; in this case, the confusion sets can be tabulated from the annotations (Rozovskaya & Roth, 2010a; Dahlmeier & Ng, 2011a). A benefit of the corpus-driven approach is that the resulting confusion sets provide a reliable estimation of the distributions of the underlying error patterns. However, this type of annotated corpora take time and effort to develop. Moreover, even when an ESL student makes many mistakes, the proportion of the writing that contains no error is still much greater. For example, in the NUS Corpus of Learner English (NUCLE) corpus (Dahlmeier & Ng, 2011b), there are a total of 3,302 preposition mistakes out of a total of 147,087 prepositions. Therefore, to build confusion sets for open class words such as verbs, one would need a very large annotated corpus.

To address the challenge without relying on extensive human involvement, this paper proposes methods to construct confusion sets directly from standard English corpora (Section 3). We conjecture that standard English corpora contain enough information for us to infer ESL learners confusions. This is because learners' confusions are mainly caused by learners' understandings of word similarities, which is developed while studying standard English texts.

What knowledge do ESL learners learn about words? There are mainly two views. One view is that learning words mean understanding the words' meanings and their relations to one another. Another view is that learning words mainly means understanding which word to choose under which conditions.

In *lexical semantics*, people hold the first view. In this area, researchers try to find how and what words mean, denote, and their relations/similarities. This view tends to explain the cause of confusions to be the similarities between words. Dahlmeier & Ng (2011a); Liu et al. (2010) take this view in confusion set construction. They build confusion sets containing the words that are similar in semantic meanings.

In *language modeling*, people hold the second view. People consider the ability of choosing the appropriate word under each context to imply the mastery of the *language*, which include the understandings of the *words* in the language. This view tends to explain the cause of confusions to be the learners' incapability to completely manage how to use words.

3 Automatic Confusion Sets Construction

ESL writers are more likely to confuse words that they find to be similar during their language learning. In this section we present three models that simulates how ESL learners might learn words. In the first two subsections, we describe models of separately learning words' intensions and extensions, respectively. In the last subsection, we introduce a model that is optimized for learning the intensions and extensions of words all together. Within each subsection, we also develop the reason of ESL writers' confusions, and propose the corresponding way to automatically construct confusion sets.

3.1 Learning Words' Intensions – Distributional Models

Under an *intension based* perspective, a learner's primary goal is to understand word meanings, and it is the similarities between words' intensions that cause word choice confusions. However, this is not to say that learners ignore word usages. Indeed, although dictionary entries contain direct definitions of words, researches in the past showed that learners do not learn by memorizing dictionary entries; instead, they infer words' meaning/function from the context, and then connecting the new words to the words they are already feel familiar(Fischer, 1990). Under this perspective, learning the extensions of words is not explicit, it is a means to achieve the primary goal of understanding word meanings.

To simulates an intension based learner, we build a model of word similarity metrics from processing standard English text. Specifically, we build *distributional models* in which the similarities of words are calculated from a comparison of the contexts they appear in (Pereira et al., 1993; Lin, 1998; Lee, 1999). Then, to fill in a word's confusion set, we pick the words that are most similar according to the metric. Pantel & Lin (2002) showed this method is able to yield similarities that correlate well with the similarities of words' intensions.

In our work, we calculate the words' intension similarity by using a distributional model (Pereira et al., 1993; Lee, 1999), in which each preposition is represented as a distributional vector of its

context features. Examples of usage contexts that have been shown to be relevant for the task of preposition selection in previous work (De Felice, 2008; J. Tetreault et al., 2010; Dahlmeier & Ng, 2011a) include:

Gov: the syntactic dependency governors of the preposition

Obj: the dependency objects of the preposition

GovTag, ObjTag: the part-of-speech tags of the dependency governors and objects

L1-Trans: L1 translations of the preposition

We employ **Gov, Obj, GovTag, ObjTag** features to capture the grammatical context of the preposition selection. We also employ **L1-Trans** to capture both the intended semantic meaning of the preposition and the **L1** background information which was shown to be relevant to confusions(Rozovskaya & Roth, 2010a; Dahlmeier & Ng, 2011a).

The distribution of each preposition's usage context can be estimated from a standard English corpus. Then the similarity between any pair of preposition vectors can be computed using common distance metrics such as: KL-Divergence, Euclidean distance, and cosine similarity.

This approach, however, may not be appropriate for our problem for the following two reasons:

Firstly, under a distributional model, two prepositions are considered similar only if the distribution of all their usages are similar. This is a strong restriction in the sense that two prepositions might only be similar under certain specific usage contexts but are not generally similar. For example, the prepositions *of* and *for* typically have fairly distinctive usages; however, ESL writers often confuse the two if the previous word was *need*.

Secondly, even if two words have similar usages under certain usage context, i.e. have similar probabilities of being used(e.g. both with 0.2 probability), people still may not be likely to confuse them with each other – instead, they are more likely to confuse them with a third word which have higher probabilities(e.g. 0.5). This is because the learner is more likely to pick the word that seems most plausible in the context, if without further information.

3.2 Learning Words' Extensions – Preposition Selector

Under an *extension-only* model, it is assumed that the learners' main goal is to understand how to choose words in a given context, and that they learn about such knowledge from standard English text. Because classifiers can also be trained to choose words, we simulate ESL learners' learning process as training a classifier for the word selections task(J. R. Tetreault & Chodorow, 2008; J. Tetreault et al., 2010) on standard English text. The trained classifier can be seen as a type of language model: given a context, it predicts the most likely word in that context.

Under this model, it is expected that the word choice confusions are mainly caused by the learners' incapability to completely master the word usages. Therefore, to see what confusions an ESL learner may have, we then rerun the trained classifier on the training data to collect the mistakes it makes.

3.3 Learning Both Intensions and Extensions – RCA

We believe the knowledge of word intensions and extensions build on top of each other while learners learn English words. Therefore we propose a model that reflects the interactions between the understandings of words' intensions and extensions; it works toward making the intensions and extensions compatible with each other. Similar with the model in section 3.1, in the end, we build words' confusion sets by filling them in with words that are most similar in their intensions.

Our new model describes a two step process when a learner makes word selection choices: by examining the context, he/she will first think about an intension to convey; then he/she chooses a word that conveys as similar an intension as possible. We will refer to the first step as making *intension decisions*, and the second step as making *word choice decisions*. The goal of their learning is to become more comfortable about the word choices in standard English texts.

We formalize the learning process described above mathematically, to facilitate further analysis:

Intension Space We firstly assume that all possible intensions may be embedded in an Euclidean space S. Two intensions are similar when their locations in S are close to each other. The n prepositions $w_1, \ldots, w_n$ have corresponding intensions $\vec{v}_1, \ldots, \vec{v}_n \in S$. Because we mainly focus on these n prepositions, we assume that all intensions during learners' learning process can be described by a linear interpolation of the n prepositions' intension vectors $v_1, \ldots, v_n$. That is, the subspace containing all intensions learners consider has at most n dimensions. We therefore may assume $S = \mathbb{R}^n$, without loss of generality. Further, we denote $V = (\vec{v}_1, \ldots, \vec{v}_n)$, $I_i = (\underbrace{0, \ldots, 0}_{i-1}, 1, \underbrace{0, \ldots, 0}_{n-i})^T$, so that $\vec{v}_i = VI_i$. Because the n prepositions cannot be too similar to each other, their intensions $\vec{v}_1, \ldots, \vec{v}_n$ cannot clutter. We ensure this by forcing $|\det(V)| \geq 1$[2].

Intension Decisions We model the learners' ability to make *intension decisions* as a function $\vec{f}$ which maps a context C to a point in the intension space $\vec{f}(C) \in S$, which points to the intension the learner would like to choose under context C. C is in the format of a set of relevant contextual features for the preposition choice decisions. Following the discussion in section 3.1, we consider the relevant contextual features **Gov, Obj, GovTag, ObjTag, L1-Trans**.

The Uncomfortness of Word Choice Decisions For one word usage sample (C, w_i), where C is the context, and w_i is the actual preposition choice, we define the "uncomfortness" of the ESL learner by $||\vec{f}(C) - \vec{v}_i||^2$. This means: the more difference between the learner's expected word choice $\vec{f}(C)$ and the actual word choice $\vec{v}_i$, the more "uncomfortable" the learners are.

Learning Goal We assume that the learners learn about the word usages from some standard English corpus[3] D, containing word usage samples in the format (C, w_i). The learners' learning goal is to find V and $\vec{f}$ which get them most "comfortable" with the word usages in D. Therefore, mathematically, the learners' objective is to minimize the overall uncomfortness on the English text D: $\min_{\vec{f},V} \sum_{(C,w_i)\in D} ||\vec{f}(C) - v_i||^2$.

[2]Because $|\det(V)|$ is the area circled by the word vectors in V, forcing it to be higher than 1 can be interpreted as assuming the learners know beforehand that the prepositions cannot be too similar to the others.

[3]Although there may be other sources where the learners may obtain English knowledge from, such as dictionaries, the learners would learn word usages better from texts(Fischer, 1990).

Following the discussion above, we formalize the learning process of ESL learners as finding the best *uncluttered* word vectors V and word usage patterns $\vec{f}$ which together minimize the "uncomfortness" function over some standard English corpus D:

$$\min_{\vec{f},V} \sum_{(C,w_i)\in D} ||\vec{f}(C) - VI_i||^2 \quad s.t. |\det(V)| \geq 1 \tag{1}$$

We calculate the optimal set of word vectors V in the optimization problem above by firstly reducing the problem into a Minimization of Within Class Distances problem, as shown in Appendix A, and then solving it using the Relevance Component Analysis(RCA) algorithm(Bar-Hillel et al., 2006).

In the end, we will be able to obtain the word vectors for prepositions $VI_1, \ldots, VI_n$, and therefore also their similarities by calculating the distance between them (the distance between prepositions w_i, w_j is $||VI_i - VI_j||$). According to our model's assumption, after ESL learners' learning, the similarities of intensions of two words' will highly correlate to this distance. We can therefore fill in the confusion set for every preposition with the prepositions that have the least distances to it.

This approach is similar to the approach described in Section 3.2 in that it also focuses on the similarities of preposition usages under specific contexts. The two approaches differ, however, in their treatments of the degrees to which words are considered to be similar. For example, consider a corpus where under some certain context C, prepositions p_a, p_b and p_c occur 101, 100, 100 times, respectively. Using RCA, the system would consider all three to be mutually confusable because they appear almost equally frequently in the same context. On the other hand, while the preposition selector considers p_b and p_c to be confusable with p_a, it does not conclude that p_b and p_c are also mutually confusable under context C.

Thus, if most usage contexts contain only one or two preposition types, the preposition selector and RCA may produce similar confusion sets; but if the data also include usage contexts that contain three or more preposition types, RCA may offer confusion sets based on a more globally optimized similarity metric.

4 Experimental Setup

We conduct experiments to compare different methods for constructing confusion sets. To evaluate the confusion sets' qualities, we examine how they impact the performance of an end-to-end grammar error correction (GEC) system. In particular, we train a separate classifier for each preposition using only training examples that are covered by the confusion set, a setup similar to the **NegL1** system as described in (Rozovskaya & Roth, 2010a). Additionally, we also compare the confusion sets with an intrinsic evaluation; we measure how well each method's confusion sets match real ESL mistakes by calculating their *coverage* on an annotated ESL corpus.

4.1 Data

As the ground-truth for our experiments, we use the NUS Corpus of Learner English(NUCLE) (Dahlmeier & Ng, 2011b). This is an error-annotated ESL corpus; that is, the writers' mistakes have been identified and corrected by an English teacher. In this collection, many writers' native (L1) language is Chinese. Following the methodologies established in other studies on the

preposition selection problem, we focus on the 36 most frequent prepositions[4]. We used 80% of the full corpus for training, 10% for development and 10% for testing.

We use the NUCLE corpus in several ways. First, it is used to establish upper-bound confusion sets. We constructed these "gold" confusion sets by tabulating the observed preposition errors in the corpus. Second, it is used as a source of training data for the end-to-end GEC system[5]. For each confusion set construction method, we extract from the training portion of NUCLE those instances that are consistent with the proposed confusion sets to train the GEC system. The trained systems are then tested on the unfiltered test set. Third, it is used as the ground truth for computing the coverage metric.

The non-ESL corpus used for constructing confusion sets is the Foreign Broadcast Information Service (FBIS) corpus, which is a Chinese-English bilingual corpus. For most experiments, only the English portion is used. For experiments that make use of L1 translations, we extracted the Chinese translations for English prepositions using the GIZA++ (Och & Ney, 2004) implementation of the IBM word alignment model (Brown et al., 1993). Of the FBIS corpus, we used its first 32,000 sentences, which contain 151,767 prepositions.

4.2 Metrics

4.2.1 Extrinsic Evaluation

We use F_1-measure to evaluate the confusion sets' effects on the GEC system

$$F_1 = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

where **precision** is the number of suggested corrections that agree with the human annotator divided by the total number of proposed corrections by the system, and **recall** is the number of suggested corrections that agree with the human annotator divided by the total number of errors annotated by the human annotator.

A challenge faced by automatic GEC system is that ESL writers do not make mistakes on most of the usual cases. In NUCLE, 1.3% of the preposition instances contain an error. To reduce the class imbalance for the underlying classifiers during training, we follow the methodology used by Dahlmeier & Ng (2011b) to keep all instances that contain an error and retain a random sample of q percent of the correct instances in the training data. In our experiments, the value of q ($20\% \leq q \leq 40\%$) is tuned on development data. We keep the test data as it is. That is, the filtering we discussed above is only applied on the training data.

4.2.2 Intrinsic Evaluation: Coverage

When an ESL student mistakenly uses some preposition instead of the correct one, the wrong preposition is not necessarily in the proposed confusion set list. We refer to the proportion of

[4]These preposition words include *about, along, among, around, as, at, beside, besides, between, by, down, during, except, for, from, in, inside, into, of, off, on, onto, outside, over, through, to, toward, towards, under, underneath, until, up, upon, with, within, without*

[5]While using NUCLE to train the GEC system seems in contradiction with our overall aim of reducing our reliance on error-annotated corpus, we argue that the usage is appropriate here because we need to compare different approaches of constructing confusion sets without interference from other factors. We do not pursue alternatives such as injecting noise into standard English as training data (Rozovskaya & Roth, 2010a,b) to avoid unintended interactions between the confusion sets and the error generation methods.

ESL students' mistakes in a corpus that fall into the proposed confusion set list as the *coverage* of the confusion set list on that corpus.

The coverage metric can be seen as measuring *recall*: how well does the proposed confusion set table cover the mistakes in some ESL corpus? If each confusion set includes all the prepositions, then the coverage would be 100%. As discussed earlier, in order for the confusion sets to be useful, they cannot be too large. A high quality confusion set table is one whose confusion sets are small in their sizes but cover the majority of the mistakes seen in the ESL corpus.

4.3 Confusion Set Construction Methods

Our experiments compare the following confusion set construction methods:

The Trivial Confusion Sets(all preps) To show the confusion sets' effect in general from comparison, we establish a baseline by using the trivial confusion sets, in which all prepositions are considered to be confusable to each other.

Construction from NUCLE(gold) We establish the upper-bound of the confusion set table by tabulating the preposition mistakes in NUCLE. This confusion set table contains the most prepositions, and therefore is the one with the highest coverage of ESL mistakes.

Construction by Distributional Similarity Metrics As described in Section 3.1, this model represents a preposition as a feature vector and directly computes the distance between pairs of prepositions to construct confusion sets. The values of the feature vectors are computed from the FBIS corpus. Three standard distance/similarity measures are used: KL-Divergence(kl div), Euclidean Distance(euc dist) and Cosine Similarity(cos sim).

Construction from Preposition Selector Errors(selector) Section 3.2 proposes generating confusion sets from classification errors. Here, we train a Maximum Entropy classifier[6] for the preposition selection task on the FBIS corpus, and rerun the classifier on the same data to collect the mistakes it still makes.

Construction by Word Usage Similarity Modeling(RCA) In Section 3.3 we proposed to simulate ESL learners' learning of both words' intensions and extensions. We formalize their learning as an optimization problem and then calculate words' intensions and extensions using the RCA algorithm(Bar-Hillel et al., 2006). The final confusion sets contain words which have similar intensions.

4.3.1 Fixing Sizes of Confusion Sets

Our evaluation fixes the size of the confusion sets in the final confusion set tables to be N, where $3 \leq N \leq 7$. This is mainly because confusion sets tables with sizes greater than 7 are able to cover over 90% of the ESL mistakes, and increasing confusion sets' sizes from there start to hurt the GEC systems' performance. On the other hand, when the sizes are too small, the confusion set lists prevents the GEC system from making reasonable corrections.

5 Experiments

We compare the proposed methods of constructing confusion sets by using the resulting confusion sets in an end-to-end GEC system as described in Section 4. The experiments aim

[6]We used the package downloaded from `http://homepages.inf.ed.ac.uk/lzhang10/maxent_toolkit.html`

to address the following questions: (1) How does the proposed method for automatically constructing confusion sets from non-ESL corpus compare against those developed from error-annotated ESL corpus? (2) Regarding the models for ESL learners' word learning, does considering the interactions between their learning of words' intensions and extensions help to capture the learners' confusions? (3) How are these models affected by the choices of different context feature groups? (4) How would the quality be affected by the choice of confusion set sizes?

Figure 1 shows a summary of the results. Each plot shows the GEC system's *performance* versus the *size* of the confusion sets for each confusion sets construction method under a different set of context feature choices [7]. In the baseline all preps, because we always fix the confusion sets' size to be a constant number 36 to contain all prepositions, the resulting curves are displayed as horizontal lines in the figures.

We make four observations:

First, regarding the use of non-ESL corpus, the experimental results suggest that confusion sets that are automatically constructed from non-ESL corpus is competitive with those constructed from an error-corrected ESL corpus. When picking the best feature sets **Gov,Obj,L1-Trans** in RCA, the GEC system can perform as well as if it were using the gold confusion sets constructed from a corpus containing 150K preposition usages.

Second, regarding the models for ESL learners' word learning, our experiments suggest that the learners' confusions are better captured when we model their learning of both words' intensions and extensions altogether. In our experimental results, confusion sets constructed by RCA model, which considers the interaction of words' intensions and extensions, consistently outperforms the other automatic methods selector, kl div, euc dist, cos sim, which only consider the learning of either words' intensions or their extensions.

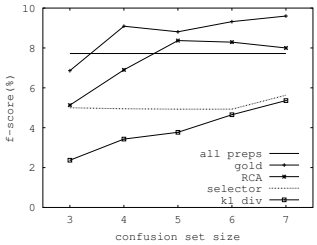
Third, regarding the feature sets used in constructing confusion sets, we find that in general all the models tend to perform better when they use more features. For example, by using **Gov,Obj** in addition to **L1-Trans**, selector raises the GEC system's F-score from 5.00% to 8.81%. RCA, however, is more stable with respect to the feature set changes. We separately show, for these two models, a comparison of the features' effects on them in Figure 2.

Fourth, our evaluation confirms that, in general, using confusion sets helps improving the GEC system's performance. This is because by limiting the confusion set's sizes, one can greatly reduce the underlying classifiers' mis-classification errors, at the cost of reducing their coverage a little. These two factors together lead to positive changes overall. To further demonstrate this effect, we show in Table 1 statistics of the decomposition of GEC systems' errors on the testing dataset. Also worth noticing is that our proposed approach (RCA), although having a slightly less coverage compared to the (gold), reduces mis-classification errors even further.

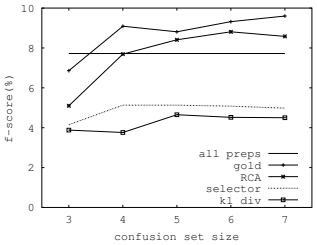
5.1 Discussions

The experiments above demonstrated RCA's strength over other methods. In this section we provide more in-depth analysis on the differences between RCA and other methods, by comparing those methods' effects on the GEC system's precision and recall separately.

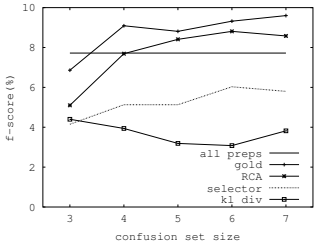
[7] Note that among the standard similarity metrics, we only plot kl div's F_1-scores because it performs better or similar to the other two methods in most of the cases. In later experiments, we will also only demonstrate the best of the three when all of them are performing similarly.



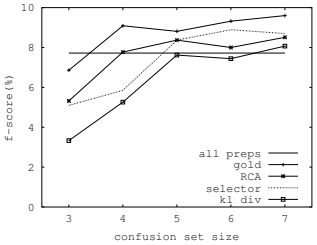
(a) Using **L1-Trans**



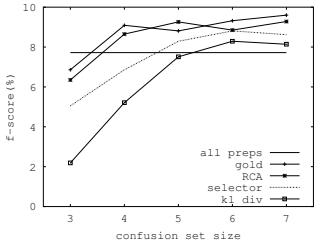
(b) Using **GovTag,ObjTag**



(c) Using **GovTag,ObjTag,L1-Trans**

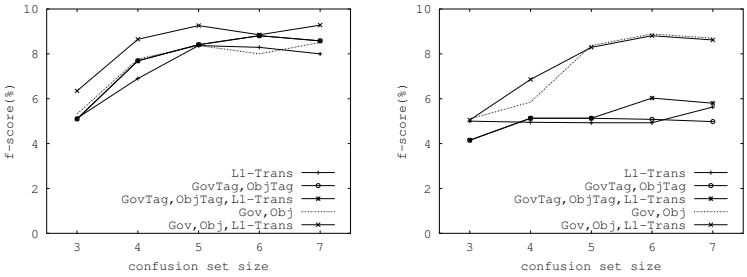


(d) Using **Gov,Obj**



(e) Using **Gov,Obj,L1-Trans**

Figure 1: F_1-Scores of different confusion set construction methods. For each of the five feature combinations, a plot demonstrates the performance of different methods using that feature combination. We display every method's performance as a curve in which each point represents the GEC system's F_1-Score when using that method to construct confusion set list of a particular size for the 36 prepositions.



(a) RCA methods' F_1 scores

(b) Selector methods' F_1 scores

Figure 2: F_1-scores of models using different feature sets to build confusion sets for all 36 prepositions.

	all	size=3		size=4		size=5		size=6		size=7	
	preps	gold	RCA	gold	RCA	gold	RCA	gold	RCA	gold	RCA
Out of Coverage	0	54	58	37	42	33	37	24	29	22	19
Mis-classification	284	135	119	162	147	173	158	189	176	203	195

Table 1: Confusion sets help reducing mis-classification errors. Here we categorize the GEC system's mistakes by whether they are caused by the confusion sets. *Out of Coverage* represents the cases where confusion sets precluded the right correction to be made, while *Mis-classification* includes all the other cases where the underlying classifiers are responsible for the prediction mistakes. The RCA we demonstrate here uses **Gov,Obj,L1-Trans** features.

We fix the feature set that all methods use to be **Gov,Obj,L1-Trans** in the discussion, because it allows all models to perform their best.

5.1.1 Precision

In Figure 3, comparing with the all prep baseline, we see that by limiting the classifiers' choices, confusion sets are indeed able to raise up GEC systems' precision. The confusion sets computed by RCA and euc dist are more helpful in raising the GEC system's precision, in contrast with selector. The difference is more significant when the confusion sets are small.

5.1.2 Confusion Set Coverage

Furthermore, we would like to provide an analysis of the GEC system's recalls, which is, in our setup, mainly *affected* by the number of ESL mistakes that are precluded from classifiers' consideration by the confusion sets. We measure this by calculating the proportion of ESL mistakes they cover using the metrics developed in 4.2.2. The coverage also reflects one confusion set's match to ESL students' real mistakes.

Shown in Figure 4 are the coverage of confusion sets constructed by different models, of different sizes. RCA and the selector greatly outperform other automatic approaches.

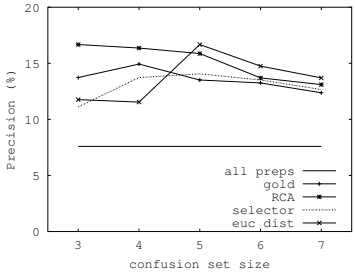


Figure 3: Precision of different methods on NUCLE

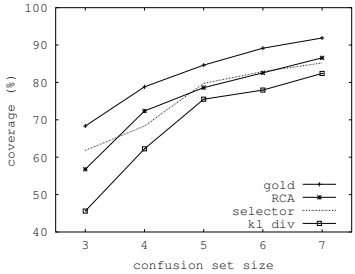


Figure 4: Coverage of confusion sets in different models using features **Gov,Obj,L1-Trans**

6 Conclusions

We proposed a method to automatically construct confusion sets for preposition errors without relying on any annotated ESL corpus or human post-processing. Based on the notion that ESL word selection errors are mainly because ESL learners are not able to choose between similar words, we build a model that analyzes which words might appear similar to each other to an ESL learner. Our model applies an algorithm called Relevance Component Analysis (Bar-Hillel et al., 2006) to describe how an ESL learner might learn both words' intensions and extensions from reading English text. The resulting confusion sets have been shown to both improve GEC system's performance, and correlate well with real ESL mistakes. Also, by modeling the interaction between the intensional and extensional knowledge in ESL learners' learning, our model ends up with better confusion sets than the models considering the development of *only* intensional or extensional knowledge. One key strength of our proposed technique is that because it only relies on standard English corpora, it is more scalable. Although this paper focuses on prepositions, the proposed approach may be applicable to other word classes.

Acknowledgments

This work is supported by U.S. National Science Foundation Grant IIS-0745914. We thank the anonymous reviewers for their suggestions; we also thank Janyce Wiebe, Joel Tetreault, Yafei Wei and Lingjia Deng for helpful discussions.

A Solving the Optimization Problem for Word Usage Similarity

To solve the minimization problem in formula 1, we will first cast it into a Minimization of Within Class Distances problem.

Firstly, suppose there are N unique contexts $C_1, \ldots, C_N$ in the corpus, note that by grouping the samples with same contexts together, we may rewrite formula 1 as:

$$\min_{f,V} \sum_{1\le k\le N} \sum_{(C_k,w_i)\in D_k} ||\vec{f}(C_k) - VI_i||^2 \quad s.t. |\det(V)| \ge 1$$

where $D_k = \{(C, w_i) \in D \mid C = C_k\}$.

Secondly, for a certain V, the optimal function $\vec{f}$ which minimizes the cost function should satisfy: $\vec{f}(C_k) = \frac{\sum_{(C_k,w_i)\in D_k} VI_i}{|D_k|} = V\vec{m}_k$, where $\vec{m}_k = \frac{\sum_{(C_k,w_i)\in D_k} I_i}{|D_k|}$. That is, $\vec{f}$ should map context C_k to the centroid of the word choice vectors in group D_k. We may therefore rewrite the formula above as:

$$\min_{V} \sum_{1\le k\le N} \sum_{(C_k,w_i)\in D_k} ||V\vec{m}_k - VI_i||^2 \quad s.t. |\det(V)| \ge 1$$

$$\Leftrightarrow \quad \min_{V} \sum_{1\le k\le N} \sum_{(C_k,w_i)\in D_k} ||\vec{m}_k - I_i||^2_{V^TV} \quad s.t. \det(V^TV) \ge 1$$

where the notation $||\vec{t}||_B$ is the Mahalanobis distance: $||\vec{t}||_B = \sqrt{\vec{t}^T B \vec{t}}$. Together, this gives us the exact equation for the minimization of within class distances problem that the RCA algorithm may solve(Bar-Hillel et al., 2006, p. 945). We therefore directly apply the RCA algorithm to calculate the optimal V: $V = T\hat{R}^{-\frac{1}{2}}$ where T is a constant number and

$$\hat{R} = \sum_{1\le k\le N} \sum_{(C_k,w_i)\in D_k} (I_i - \vec{m}_k)(I_i - \vec{m}_k)^T$$

References

Bar-Hillel, A., Hertz, T., Shental, N., & Weinshall, D. (2006). Learning a mahalanobis metric from equivalence constraints. *Journal of Machine Learning Research*, *6*(1), 937.

Brockett, C., Dolan, W. B., & Gamon, M. (2006). Correcting esl errors using phrasal smt techniques. In *Proceedings of the 21st international conference on computational linguistics and the 44th annual meeting of the association for computational linguistics* (pp. 249–256). Sydney, Australia: Association for Computational Linguistics.

Brown, P., Pietra, V., Pietra, S., & Mercer, R. (1993). The mathematics of statistical machine translation: Parameter estimation. *Computational linguistics*, *19*(2), 263–311.

Chalmers, D. J. (2002). On sense and intension. *Noûs*, *36*, 135–182.

Chodorow, M., Tetreault, J. R., & Han, N.-R. (2007). Detection of grammatical errors involving prepositions. In *Proceedings of the fourth acl-sigsem workshop on prepositions* (pp. 25–30). Prague, Czech Republic: Association for Computational Linguistics.

Dahlmeier, D., & Ng, H. (2011a, July). Correcting semantic collocation errors with l1-induced paraphrases. In *Proceedings of the 2011 conference on empirical methods in natural language processing* (pp. 107–117). Edinburgh, Scotland, UK: Association for Computational Linguistics.

Dahlmeier, D., & Ng, H. T. (2011b). Grammatical error correction with alternating structure optimization. In *Proceedings of the 49th annual meeting of the association for computational linguistics: Human language technologies - volume 1* (pp. 915–923). Portland, Oregon, USA: Association for Computational Linguistics.

De Felice, R. (2008). *Automatic error detection in non-native english*. Unpublished doctoral dissertation, University of Oxford.

Fischer, U. (1990). *How students learn words from a dictionary and in context*. Unpublished doctoral dissertation, Princeton University.

Gamon, M., Gao, J., Brockett, C., Klementiev, A., Dolan, W., Belenko, D., & Vanderwende, L. (2009). Using contextual speller techniques and language modeling for esl error correction. *Urbana*, *51*, 61801.

Leacock, C., Chodorow, M., Gamon, M., & Tetreault, J. (2010). Automated grammatical error detection for language learners. *Synthesis lectures on human language technologies*, *3*(1), 1–134.

Lee, L. (1999). Measures of distributional similarity. In *Proceedings of the 37th annual meeting of the association for computational linguistics on computational linguistics* (pp. 25–32). College Park, Maryland: Association for Computational Linguistics.

Lin, D. (1998). An information-theoretic definition of similarity. In *Proceedings of the fifteenth international conference on machine learning* (pp. 296–304). San Francisco, CA, USA: Morgan Kaufmann Publishers Inc.

Liu, X., Han, B., Li, K., Stiller, S. H., & Zhou, M. (2010). Srl-based verb selection for esl. In *Proceedings of the 2010 conference on empirical methods in natural language processing* (pp. 1068–1076). Cambridge, Massachusetts: Association for Computational Linguistics.

Och, F., & Ney, H. (2004). The alignment template approach to statistical machine translation. *Computational Linguistics*, *30*(4), 417–449.

Pantel, P., & Lin, D. (2002). Discovering word senses from text. In *Proceedings of the eighth acm sigkdd international conference on knowledge discovery and data mining* (pp. 613–619). Edmonton, Alberta, Canada: ACM.

Pereira, F., Tishby, N., & Lee, L. (1993). Distributional clustering of english words. In *Proceedings of the 31st annual meeting on association for computational linguistics* (pp. 183–190). Columbus, Ohio: Association for Computational Linguistics.

Rozovskaya, A., & Roth, D. (2010a). Generating confusion sets for context-sensitive error correction. In *Proceedings of the 2010 conference on empirical methods in natural language processing* (pp. 961–970). Cambridge, Massachusetts: Association for Computational Linguistics.

Rozovskaya, A., & Roth, D. (2010b). Training paradigms for correcting errors in grammar and usage. In *Human language technologies: The 2010 annual conference of the north american chapter of the association for computational linguistics* (pp. 154–162).

Rozovskaya, A., & Roth, D. (2011, June). Algorithm selection and model adaptation for esl correction tasks. In *Proceedings of the 49th annual meeting of the association for computational linguistics: Human language technologies* (pp. 924–933). Portland, Oregon, USA: Association for Computational Linguistics.

Tetreault, J., Foster, J., & Chodorow, M. (2010). Using parse features for preposition selection and error detection. In *Proceedings of the acl 2010 conference short papers* (pp. 353–358). Uppsala, Sweden: Association for Computational Linguistics.

Tetreault, J. R., & Chodorow, M. (2008). The ups and downs of preposition error detection in esl writing. In *Proceedings of the 22nd international conference on computational linguistics - volume 1* (pp. 865–872). Manchester, United Kingdom: Association for Computational Linguistics.

ISO-TimeML Event Extraction in Persian Text

Yadollah Yaghoobzadeh[1] *Gholamreza Ghassem-sani*[1] *Seyed Abolghasem Mirroshandel*[2] *Mahbaneh Eshaghzadeh*[1]
(1) Department of Computer Engineering, Sharif University of Technology, Tehran, Iran
(2) Faculty of Engineering, University of Guilan, Rasht, Iran
yaghoobzadeh@ce.sharif.edu, sani@sharif.edu,
mirroshandel@guilan.ac.ir, *eshaghzadeh*@ce.sharif.edu

ABSTRACT

Recognizing TimeML events and identifying their attributes, are important tasks in natural language processing (NLP). Several NLP applications like question answering, information retrieval, summarization, and temporal information extraction need to have some knowledge about events of the input documents. Existing methods developed for this task are restricted to limited number of languages, and for many other languages including Persian, there has not been any effort yet. In this paper, we introduce two different approaches for automatic event recognition and classification in Persian. For this purpose, a corpus of events has been built based on a specific version of ISO-TimeML for Persian. We present the specification of this corpus together with the results of applying mentioned approaches to the corpus. Considering these methods are the first effort towards Persian event extraction, the results are comparable to that of successful methods in English.

TITLE AND ABSTRACT IN PERSIAN

استخراج رویدادها از متون فارسی بنا بر تعریف ISO-TimeML

یافتن رویدادها و ویژگی‌های آنها بر اساس TimeML یکی از مسائل مهم در حوزه‌ی پردازش زبان های طبیعی است. بسیاری از کاربردهای پردازش زبان‌های طبیعی مانند سامانه‌های پرسش و پاسخ، استخراج اطلاعات، خلاصه سازی و استخراج اطلاعات زمانی نیاز دارند تا دانشی درباره رویدادهای موجود در متون ورودی داشته باشند. روش‌هایی که تاکنون در مورد این مسئله ایجاد شده، محدود به چند زبان خاص است و در بسیاری از زبان‌ها از جمله زبان فارسی، تاکنون کاری در این رابطه انجام نشده است. در این مقاله، ما دو روش مختلف برای استخراج رویدادها در زبان فارسی ارائه می‌دهیم. برای این کار، پیکره‌ای مطابق با ISO-TimeML، البته نسخه‌ی خاص فارسی آن، ساخته شد. ما مشخصات این پیکره و نتایج حاصل بر روی آن، را نشان می‌دهیم. نتایج روش‌های ارائه‌شده در این مقاله، به عنوان اولین روش‌های پیاده‌سازی شده بر روی زبان فارسی، با روش های موفق در زبان انگلیسی قابل مقایسه است.

KEYWORDS : Event Mention, Temporal Information Extraction, Classification, Annotation Scheme, TimeBank, ISO-TimeML, Persian Language.

KEYWORDS IN PERSIAN : ذکر رویداد، استخراج اطلاعات زمانی، رده بندی، شمای برچسبگذاری، تایم بانک، استاندارد تایم ام ال، زبان فارسی

Proceedings of COLING 2012: Technical Papers, pages 2931–2944,
COLING 2012, Mumbai, December 2012.

1 Introduction

Event extraction is a demanding task in natural language processing (NLP). Several applications like question answering (QA), information retrieval (IR), summarization, and temporal information extraction need to have some knowledge about events for better operation. This task remains a challenging task. Recognizing the various forms in which an event may be expressed (verbs, nouns, adjectives and prepositions), distinguishing events of different classes, and finding the features of an event are all difficult task (Verhagen et al., 2010).

In this paper, the events are defined based on TimeML view. In TimeML, events are "*situations* that *occur* or *happen*, or predicates that describe *states* or *circumstances* in which something obtains or holds the truth" (Pustejovsky et al., 2003). The task of event extraction includes two major stages that are introduced by TimeML: 1) detection and annotation of a text span (i.e., verbs, nouns, predicative constructions, prepositional phrases, and adjectival phrases) that is an event, and 2) determining the semantic class of events (i.e., *Reporting*, *Perception*, *I_Action*, *I_State*, *State*, *Occurrence*, and *Aspectual*).

Performing these two tasks (event mention detection and classification) in any language requires a corpus of annotated events, at least for measuring the accuracy of the algorithm. Currently, there are no such corpora for many languages including Persian, which is the native natural language of Iran, Afghanistan and Tajikistan. We have developed a corpus of annotated events in order to extract events from Persian texts. This corpus contains 4237 events. The annotation process has been based on an adapted version of ISO-TimeML guidelines. We have applied some changes to event attributes, the value of these attributes, annotation rules, and event extents.

In this paper, we also propose a system for automatic event recognition. In the system, various morphological, syntactic, and semantic features have been used. The syntactic features are in the form of dependency parse trees. Semantic features are taken from a Persian version of WordNet. For identification of event mentions and classifying them, the system uses these features in two different methods: a rule-based and a learning-based method. In the rule-based method, we proposed several rules for different types of events. In the learning-based method, a classification technique has been used for identification of events. We have used different models for different forms of events (i.e., verb, noun, and adjective). Our experiments show that the proposed methods, which are the first attempt in Persian event extraction, are quite effective.

The remainder of this paper is organized as follows: section 2 is about ISO-TimeML adaptation for Persian. Section 3 explains previous work in event extraction. Our event extraction system is proposed in section 4. The experimental results of the system are presented in section 5. Finally, the last section of the paper includes conclusion and some possible future work.

2 Adapting ISO-TimeML Event Guidelines for Persian

To apply ISO-TimeML as an annotation scheme to a new language, the language specific issues should be considered carefully. Accordingly, some aspects of scheme must be modified and some others must be restated for target language. The adapted schema may go through various changes regarding event attributes, event attribute values, event annotation rules and event extent rules according to the target language structure.

A number of languages including Korean (Im et al., 2009), Italian (Caselli et al., 2011), French (Bittar et al., 2011) have already adapted TimeML and ISO-TimeML guidelines to their needs. In the following subsections, we present the adapted version of ISO-TimeML for Persian (let us call it PersTimeML) in three main categories: event annotation, event extent and event attributes.

2.1 Event Annotation

Event annotation in PersTimeML is mostly based on ISO-TimeML. Generally, for simplicity it is assumed in current version of PersTimeML that generic events must be annotated. Also there have been special cases that have been tailored particularly for the specific properties of Persian. Here, we only discuss these cases disregarding common situations with ISO-TimeML.

2.1.1 Nouns

In Persian, gerund phrases, known as "esm-e masdar", must always be annotated as events, even when they represent generic events. These are built by affixing a particular Persian letter, i.e. "nôn", to the verb stem. There are also some categories of nouns that function like gerund phrases but do not have any lexical mark. These nouns were named *predicative nouns* and defined as nouns that inherit some verb and some noun characteristics (Karimi-Doostan, 2011). In Persian sentences, these nouns are usually the starting point of an NP or a PP. We always annotate these nouns as events, too. Following examples are instances of these cases:

a. **Barresê-e** (Review) maqâle-hâ (papers) chand (a few) rôz (day) tôl mikeshad (takes).
   Translation: **Reviewing** of the papers takes a few days.
b. Ostâd (Instructor) bâ (with) **taavêq-e** (postpone) emtehân (exam) mokhâlefat kard (disagreed).
   Translation: The professor disagreed with **postponing** the exam.
c. Alê (Ali) be (to) **jostojô** (search) dar miân-e (through) sâyt (site) edâme dâd (continued).
   Translation: Ali continued to **search** in the site.

In examples (a) and (b), "barresêy-e" (review) and "taavêq-e" (postpone) are *predicative nouns*, when have an "e" mark in their end and are linked to their subsequent nouns. In example (c), "jostojô" (search) is followed by "be" (in) as a preposition.

2.1.2 Adjectives

In addition to TimeML guidelines for annotating adjectives, we must also consider *objective deverbal adjectives* in PersTimeML. These are adjectives that derived from passive modes of verbs (Lesani, 2003). Two examples are "neveshte shode" (written) and "gerefte shode" (taken). *Objective deverbal adjectives* translate to the past participle form of the verbs in English. These adjectives always must be annotated as events because they are implying verbal events that have occurred in the past. For example, in the following sentence the "neveshte shode" (written) is an *objective deverbal adjective* that must be tagged as an event.

a. Ô (He) dastân-e (story) **neveshte shode** (written) dar (in) ân (that) rôzname (newspaper) râ bâvar nakard (didn't believe).
   Translation: He didn't believe the **written** story in that newspaper.

2.2 Event Extents

According to the new paradigm in ISO-TimeML about the stand-off annotation instead of in-line annotation (Pustejovsky et al., 2010), we can annotate multiple tokens in an event tag even though the tokens are not located consecutively in the sentence. This new approach simplifies the handling of compound words (e.g. compound verbs in Persian) by tagging all the associated tokens as just one event. The majority of the Persian verbs are compound. Persian compound verbs consist of a light verb and a number of non-verbal elements. For example, "barkhord kardan" (to hit) is a compound verb including the light verb of "kardan" (do) and the non-verbal element of "barkhord" (hit) (Rasooli et al., 2011).

Since there is not a fixed list of compound verbs, recognizing them in the sentences is difficult. Besides, detecting all parts of compound verbs can be challenging for an annotator because they may be located separately with long distances. The following examples show how we annotate compound verbs with event tags.

a. Bârân (Rain) be (to) mantaq-e (area) **sadame-e** (damage) zyâdê (large) **khâhad zad** (will do).
   Translation: The rain will largely damage the area.
   - Part of PersTimeML output will be:
     <Event xml:id="e1" target="#token3#token5" text= "sadame-e khâhad zad"… />
b. Bâ (with) oo (he) **sohbat** (talk) **kardam** (did).
   Translation: I talked with him.
   - Part of PersTimeML output will be:
     <Event xml:id="e1" target="#token2#token3" text=" sohbat kardam"… />

Segmentation of Persian verbs is another difficulty regarding annotation of verbs (Shamsfard, 2011). For tackling this problem, we need to pre-process the sentences to merge all Tense/Aspect/Mood (TMA) mark tokens with their verbal head. Therefore, in example (a) above, in the pre-processing, the whole "khâhad zad" (will do) will be merged as one token by inserting short space between the verb parts.

2.3 Event Attributes

Persian, as a natural language has its own specific aspects in event attributes. As we currently know, in Persian TMAs are not separated. However, we can map the TMA values to *tense*, *aspect* and *mood* appropriately to follow the ISO-TimeML guideline. According to this mapping, the possible values for each of these attributes will be: *tense***:** "past", "present", "future", "none", *aspect***:** "perfective", "progressive", "imperfective_perfective", "imperfective", "none" and *mood***:** "subjunctive", "imperative", "indicative", "none".

3 Automatic Event Extraction Methods

There are many ongoing researches on event extraction according to TimeML specification language. Almost all detection systems act in three following stages: 1) pre-processing; 2) event mention detection; and 3) event attributes detection. Existing methods can be divided into rule-based, statistical, and combined approaches, which are explained in more details in the next following subsections.

3.1 Rule-based Methods

One of the first event recognition systems in French used a rule-based method (Bittar, 2009). The system utilizes specific hand-made dictionaries for event detection and classification. Besides, a number of rules were used to reduce the errors. In other words, event recognition in this system includes two types of processing: lexical, using specific dictionaries and contextual, using a number of rules.

The first system on Italian also used a rule-based approach that utilized dependency parse trees of the input sentences (Robaldo et al., 2011). The rules of this system have designed based on both syntactic and lexical information using a number of keywords. Some extra resources like WordNet (Fellbaum, 1998) plus a list of specific words were used to build a comprehensive keyword list. To identify the class of events, in addition to dependency structure of sentences, a list of Italian verbs with three different semantic categories (state, process, and movement) has been also utilized.

Edinburg is another rule-based system that used a named entity recognition system for nominal event detection. Special lists, which were extracted from an annotated corpus, in conjunction with WordNet were also used for detection of nominal events (Grover et al., 2010).

3.2 Statistical Methods

Bethard and Martin proposed a statistical system for event extraction using a multi-class classification method (Bethard and Martin, 2006). The system automatically annotated each token by "Inside", "Outside", or "Begin" tags. It also determined the semantic class of each event. This system used various morphological, syntactic, and semantic features. Their reported results have shown an acceptable rate in event detection.

Another statistical method for event extraction applied a classification algorithm based on Support Vector Machines (SVM) in both sentence and word levels (March and Baldwin, 2008). First, it filters out the sentences without events. Then, in the remaining sentences, it searches for events. This method does not determine the features of events (i.e., the third stage of the event extraction task).

TipSem is one of the most successful TimeML event extraction methods, which has used Conditional Random Fields (CRF) classification technique (Llorens et al., 2010). It utilizes morphological, syntactic, and semantic features in addition to semantic roles for event extraction.

3.3 Combined Methods

Evita is the first system that has been designed for extraction of TimeML events (Sauri et al., 2005). The system has benefited from both statistical and rule-based techniques. In pre-processing stage, it extracts part of speech tags, phrase chunks, and lemmas of the sentences' tokens using some existing tools. For event detection of nouns, Evita uses WordNet, and in ambiguous cases, i.e. nouns that may or may not be an event in sentences, a Bayesian classifier is used for disambiguation. This classifier has been trained on the SemCore data. For adjectives, the cases that have been annotated in TimeBank are considered as events. The heads of the predicative complements are also regarded as events. Evita uses TimeBank information for finding the class of events. It simply chooses the majority class for the specific event in the corpus.

TRIOS is another combined method, which first utilizes the TRIPS semantic parser (UzZaman and Allen, 2010). Based on the output of the parser and applying some rules, it detects the events. Then, for improving the accuracy, a Markov Logic Network (MLN) classifier is used. This classifier is also used for extraction of event features.

4 Automatic Event Extraction from Persian Text

As mentioned before, most of the event recognition and classification methods include three phases: pre-processing, event mention detection and event attributes extraction. These phases also exist in our system for Persian texts. We called this system Persian Event Tagger (PET). In pre-processing stage of PET, we convert input documents into the CoNLL-2009 Shared Task (CoNLL09) format by using Dadegan tools[1]. It means that the Dadegan tools extract part of speech (POS) tags, dependency labels, and other necessary information from input documents. The dependency labels are according to Persian dependency Treebank, which is the first released dependency Treebank for Persian. This corpus currently contains 30,000 sentences that were manually annotated[2].

We apply our feature extractor subsystem to the converted document to obtain useful and meaningful features for recognizing and classifying of events. The outputs are data set files, which contain all features for each token in a separate line. We employ these data set files to perform event mention detection and event attribute detection stages. A rule-based and a statistical learning-based approach are implemented for both of these stages. In event attribute detection, we just find *class* because other attributes like *tense*, *mood,* and *aspect* were previously found with Dadegan tools. For other event attributes, we just choose the default values; for instance, we set *polarity* to "Positive". In section 4.1, the features are discussed. The rule-based and learning-based subsystems are explained in sections 4.2 and 4.3, respectively.

4.1 Features

The features that have been used in PET can be clustered into three types: lexical, syntactic, and semantic. In the following, each type of these features is explained in more details:

4.1.1 Lexical Features

Lexical features that we have used in the system are token's text, coarse-grained POS, fine-grained POS, word's stem, word's postfix (i.e., last three letters of word), and a Boolean feature *isModAux, which* is "true" for modal or auxiliary verbs. For recognizing these features, we rely on the raw text, coarse-grained, and fine-grained POS from the pre-processed input file.

4.1.2 Syntactic Features

In some event tagging situations, PET requires deeper knowledge about sentences. Dependency parsing is a new and effective approach for obtaining this knowledge. We can extract various syntactic features from dependency labels of the input sentences.

The extracted syntactic features, for each candidate, include: dependency label, lemma of the head token, text of the head token, POS of the head token, dependency label of the head token, governing verb text, governing verb lemma, governing noun text, and governing noun lemma.

[1] Freely abailable for download at http://dadegan.ir/en/tools
[2] Freely available for download at http://dadegan.ir/en/persiandependencytreebank

Other extracted syntactic features are *isPartOfCompoundVerb* and *isPartOfCompoundNoun*, which indicate that candidate word is part of a compound verb or noun, respectively. These features can be assigned by searching the dependency parse tree of the input sentence.

4.1.3 Semantic Features

A number of semantic features are required for recognizing nominal events. They are also needed for classification of both nominal and verbal events. These features can be obtained from resources like WordNet by searching through word senses and checking their hypernyms. In Persian, we have two resources that are similar to WordNet. One of them is FarsNet, which has been developed semi-automatically with 9,266 synsets and 13,155 words (Shamsfard et al., 2010). Another resource, which we call it PersianWN, has been developed automatically. This resource, has covered 29,716 Persian phrases with reported precision 82.6% (Montazery and Faili, 2010).

To recognize nominal events, a number of Boolean features are extracted from both FarsNet and PersianWN. We consider several synsets including: "event", "human_action", "human_activity", "act", "phenomenon", and "action" to be eventive. Thus, when hypernyms of a sense fall into these synsets, we will consider this sense to be eventive. In this way, we can extract following features for each noun according to its senses: *isAllSensesEvent*, *isAllSensesState*, *isMainSenseEvent* (i.e., "true" when more than 1/3 of senses are eventive), *isOneSenseEvent*, *isOneSenseState*.

To classify events, some other features are extracted. For each event *class*, a list of phrases is created. These lists are initially filled by sample phrases that have been mentioned in the ISO-TimeML guideline. Then, we augment these lists by adding their synonyms taken from FarsNet. After creating and enriching these lists, we set a number of features (i.e., *isReporting*, *isAspectual*, *isPerception*, *isI_Action*, *isI_State*, *isState*, and *IsOccurance*) for each input phrase. We search a phrase in each list and if it is successfully found, the corresponding feature, which is related to that list, will be assigned to "true". For instance, when verb lemma exists in the aspectual phrase list, the *isAspectual* feature will be set to "true".

It must be noted that there is a difficulty in searching compound nouns/verbs in a list, lexicon, or dictionary. In these cases, the system has to combine all parts of the noun/verb before starting to look up. These parts can be obtained and then combined using related labels. For example, in sentence (a) we must look up the whole text "bazgoo kard" (restate) in dictionaries or lexicons.

a. Ô (He) moshkelât (problems) râ (-) **bâzgô** (restate) **kard** (did)**.**
Translation: He **restated** the problems.

4.2 Rule-based Method

For event mention detection, we apply PersTimeML guidelines to the input text by utilizing the previously mentioned features. Besides, a number of lists including special phrases like aspectual and causative signal words are used for event tagging. Therefore, for each candidate word, we apply a number of if-then-else rules, which are based on one or more features. These rules are explained in the following subsections.

4.2.1 Recognizing Verbal Events

According to PersTimeML, we annotate all verbs as events except for modals, auxiliary verbs, and verb "to be". We can easily find these verbs by checking the Boolean feature *isModAux*.

Furthermore, for each verb, we search the sentence for its probable non-verbal elements. If any such elements are found, they will be combined with the verb. These searches can be performed using related syntactic features, i.e., *isPartOfCompoundVerb* and position of the head token.

4.2.2 Recognizing Nominal Events

Recognizing nominal events is more challenging than other forms of events and requires a deeper analysis of input sentences. Some PersTimeML rules can be applied to the nouns by utilizing the mentioned syntactic features. In current version of PET, we apply causative and aspectual rules. This means that a noun when appears in a specific position in an aspectual or a causative structure is tagged as an event. These aspectual and causative structures are found by searching the context of the noun for occurrence of an aspectual or a causative signal word.

For instance, in sentence (a) below, there is a causative structure and therefore according to the PersTimeML, we must annotate all of "bârân-e" (rain), "seyl" (flooding), and "môjeb-e" (cause) as separate events. With dependency labels, we can determine the subject of a causative sentence. Then, if the subject is a phenomenal noun, we will annotate it as an event. Furthermore, "seyl" (flooding) can be recognized by annotating the head of the noun phrase that immediately appear after the signal word "môjeb-e" (cause).

Sentence (b), has an aspectual structure, which is triggered when encountering the verb "âghâz shod" (has started). We must annotate the subject of this structure, i.e., "marg" (death), as an event. In other structures, when an aspectual or a causative signal word is triggered, we can apply similar rules for finding the events.

a. [Bârân-e] (Rain) zyâd (heavy) **[môjeb-e]** (cause) [seyl] (flooding) shod (become).
   Translation: The heavy rain **caused** flooding.
b. [Marg-e] (Death) khôkhây-e (pigs) **âghâz shod** (has started).
   Translation: The death of pigs **has started**.

For other nouns, we first disregard each noun that have *isPartOfCompoundVerb* feature with value of "true", because in fact, it is part of a compound verb in the sentence. Besides, if a noun is part of another noun, i.e., has value of "true" for *isPartOfCompoundNoun* feature, it will be annotated in conjunction with its governing noun as an event tag.

Finally, for remaining nouns, our rule-based module can only use the semantic features, because currently there is not any acceptable Word Sense Disambiguation (WSD) system or even a semantic tagged corpus for Persian. In the module, we only use the feature *isAllSensesEvent*. When value of this feature is "true" for a noun, the noun will be annotated as an event. Another solution for improving the result of event recognition for nouns is creating a list of all *predicative nouns*. This is discussed in the evaluation section in more detail.

4.2.3 Recognizing Adjective Events

To recognize adjective events, the system first checks to see if the adjective is a predicative complement in the sentence. It can be performed by using the both dependency label and governing verb of the adjective. The adjective will be regarded as an event, if its dependency label is "MOS" (i.e., the adjective is predicate in the sentence) and its governing verb exists in a list of special predicate verbs such as "shodan" (become) and "kardan" (do).

A second rule is employed for recognizing objective deverbal adjectives by watching the postfix feature. When the value of this feature equals to "shode", we will annotate the adjective as an event.

4.2.4 Identifying the *Class* of Events

To determine *class* of events, we rely only on event *class* features (e.g., *isAspectual, isPerception, isI_State* and so on). For instance, if the value of *isReporting* feature is equal to "true", we will assign the "Reporting" value to *class* attribute. For those events, not having any of these features with value of "true", we consider the default values (i.e., "occurrence" for nouns and verbs and "state" for adjectives)

4.3 Learning-based Method

By applying the TempEval 2010 format to the developed corpus and using the previously mentioned features, a learning model can be trained. The whole relevant processes including data loading, data pre-processing, creating and applying a model, and the evaluation can be performed using an open source software called RapidMiner[3].

By RapidMiner GUI, we can easily design our learning process and obtain various desired evaluations. Therefore, our learning-based module identifies event mentions and determines the *class* of them using RapidMiner processes. We utilize a feature selection process based on a naïve Bayesian classifier in this module. The process detects optimized subsets of features for each classification task individually. The classification cases are expressed in the evaluation section.

5 Evaluation

For evaluating both PersTimeML and PET, we built a suitable corpus with annotated events. The corpus is in fact the first and a preliminary Persian version of well-known English corpus, TimeBank. We called this corpus PTB. An iterative incremental process has been used to create PersTimeML, PTB, and PET.

The input documents have been taken from Peykareh, also known as Bijankhan corpus. Peykareh is currently the most popular Persian corpus, which contains more than seven million tokens (Bijankhan et al., 2010). We selected a number of documents from diverse topics including political, economic, sport news, stories, etc. Then, we pre-processed these selected documents with Dadegan tools for tokenizing and converting to the conll09 format. We had extracted sentence texts from Conll09 files because of the necessary tokenization pre-processing that input sentences had required. Then, we applied the rule-based PET to primarily annotate events, followed by a manual correction of the system output. The manual correction was performed using the MAE (Multi-purpose Annotation Environment) annotation tool[4]. The MAE output files then converted into the TempEval 2010 data format for simplifying the evaluations.

We annotated 43 documents from Peykareh. This contained 26,949 tokens and 4,237 events. Statistics about frequency of events for each POS tag and also, frequency of events in each event *class* are shown in TABLE 1.

[3] Freely available at: http://rapid-i.com/
[4] Freely available at: http://pages.cs.brandeis.edu/~astubbs/mae.html

POS of Event	Frequency		Event Class	Frequency
Noun	1,960		*Occurrence*	2,488
Verb	1,875		*State*	623
Adjective	312		*Reporting*	433
Preposition	51		*I_Action*	276
None	39		*I_State*	218
			Aspectual	133
			Perception	66

TABLE 1 – Some statistics about PTB

To evaluate the PET, the corpus was split into a development set, a training set, and an evaluation set. The evaluation was performed token by token even for multi token events. It means that the scorer programs took each token individually and then calculated the value of each performance property like recall, precision and f-measure. The training and evaluation sets were both used in evaluating the rule-based method. We used a five-fold cross-validation with a stratified sampling over the training and evaluation sets for evaluating the learning-based method.

The results of the PET in event mention detection for both rule-based and learning-based methods are shown in TABLE 2. In order to calculate the results for each event category, we only considered tokens in that category in evaluations. In the learning-based method, this led to an individual model for each individual event category. It should also be noted that the learning-based method assigns each individual token to either an *event* class or a *non-event* one.

	Rule-based			Learning-based		
Category	Precision	Recall	F	Precision	Recall	F
All	78.9	72.5	75.6	79.2	87.5	83.1
Verb	96.5	99.3	97.9	97.1	99.5	98.3
Noun	66.3	64.4	65.3	82.1	81.8	82.0
Adjective	88.5	55.8	68.4	78.3	76.4	77.3

TABLE 2 – Evaluation results for PET for event recognition

In the evaluation for all categories, the both modules showed high precisions, while for recall the learning-based method is 15% better than the rule-based one. This weaker recall for the rule-based method is due to the lower recall for nouns and adjectives. Therefore, it showed that although the existing rules have gained a satisfactory precision, we should add extra rules or modify the lexicons for finding more event tokens.

For nouns, the most effective features that have been used in the PET are semantic features. These features had been extracted from special dictionaries and lexicons. An experiment on six random documents of the PTB showed that 60.7% of nominal events were *predicative nouns*. As we said before, *predicative nouns* in Persian when function as gerunds, are events. Therefore, it must be noted that the quality of these resources for finding eventive nouns or having a list of all *predicative nouns* immensely affects the performance of the event tagger systems.

Some coverage tests of the lexical databases were performed for the nouns of the PTB corpus. From 11,942 nouns of PTB, 7,625 nouns were found in FarsNet (63.8%) and 7,934 nouns were found in PersianWN (66.4%). Although the PersianWN has had greater coverage, it suffers from

lower accuracy because as we mentioned in 4.1, it was developed in a fully automatic process with 82.6% accuracy. On the other hand, the FarsNet was manually validated and therefore, has had high accuracy. When we experimented with PersianWN for recognizing noun events, we gained a low recall (35.1%) but a high precision (80.0%) that is a sign for its incorrect senses or hypernyms relationship between synsets.

	Rule-based			Learning-based
Category	Correct	Incorrect	Accuracy (%)	Accuracy (%)
All	1,946	819	70.3	74.9
Verb	475	353	57.4	73.5
Noun	1,301	398	76.5	82.3
Adjective	147	79	65.0	78.8

TABLE 3 – Evaluation results for event class detection

The evaluation results represented in TABLE 3 show the accuracy for detection the *class* of events. In the learning-based method, one model is created for each POS category and one classification per event is performed to assign the proper *class* to each event. The high F-measure of the rule-based approach (70.3%) is just achieved by using event *class* lists. One reason is that the majority of events have "occurrence" value in their *class* attribute. According to TABLE 2, 2,488 out of 4,237 events (i.e., 58.7% of events) have *class* attribute with value of "occurrence". Thus, a baseline system that assign "occurrence" to all events, will achieve 58.7% accuracy. By this baseline, the accuracy of 74.9% by learning-based module is both acceptable and remarkable.

	Learning-based event class detection		
Classes	Precision	Recall	F
Occurrence	73.8	91.5	81.7
State	72.2	30.7	43.1
I_State	58.2	61.5	59.8
I_Action	67.8	35.2	46.3
Reporting	92.9	86.7	89.7
Perception	75.0	46.1	57.1
Aspectual	86.2	60.1	70.9

TABLE 4 – Evaluation results for each event class

Evaluation results for the various event classes in the learning-based method are shown in TABLE 4. The worst results are in "state", "I_Action" and "perception" classes. "Perception" is scarcely occurred in the corpus and therefore, had low result. "State" and "I_Action" classes occurred in more distinct phrases in comparison with "reporting" and "aspectual" classes and thus, had lower accuracies.

Conclusion and Perspectives

In this paper, we have addressed the problem of event recognition and classification, which has been a challenging task since early days of statistical natural language processing. More specifically, we focused on ISO-TimeML event annotation for Persian. Since there have not been any suitably tagged corpus in Persian, we have developed an annotated corpus. In the annotation process, we adapted the ISO-TimeML standard for Persian. We have also proposed two different methods for automatic identifying event mentions and their corresponding attributes as part of a

Persian event tagger system. The first was utilized a rule-based approach with different rules. The second was a statistical learning-based approach, which used a classification technique for tackling the problem. Our experimental results show an acceptable accuracy, considering our system as being the first effort for event recognition in Persian.

Currently, we are working on finding ways for further improvement of our system, PET, and at the same time annotating more documents for increasing the size of the corpus. It seems that using cross-lingual techniques can further improve the accuracy of existing methods for Persian (and languages that currently lack rich resources for NLP applications). Other possible future work includes employing richer learning models such as Conditional Random Fields (CRFs) for event recognition and classification. It is also the case that PTB should be retagged by other annotators to meet the inter-annotator agreement criterion. PTB can also be further improved by annotating other ISO-TimeML tags such as time expressions and temporal relations.

Acknowledgments

This paper has been partially funded by the Iran Telecommunication Research Center (ITRC).

References

Bethard, S. and Martin, J. (2006). Identification of event mentions and their semantic class. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing-EMNLP*, pages 146–154. ACL.

Bijankhan, M., Sheykhzadegan, J., Bahrani, M. and Ghayoomi, M. (2010). Lessons from building a Persian written corpus: Peykare. *Language Resources and Evaluation*, *45*(2): 143–164.

Bittar, A. (2009). Annotation of Events and Temporal Expressions in French Texts. In *The 19th Meeting of Computational Linguistics in the Netherlands,* pages 25–38.

Bittar, A., Amsili, P., Denis, P. and Danlos, L. (2011). French TimeBank: an ISO-TimeML annotated reference corpus. In *Proceedings of the 49th Annual Meeting of ACL,* pages 130–134.

Caselli, T., Lenzi, V. B., Sprugnoli, R., Pianta, E. and Prodanof, I. (2011). Annotating Events, Temporal Expressions and Relations in Italian : the It-TimeML Experience for the Ita-TimeBank. In *Proceedings of the Fifth Law Workshop (LAW V)*, pages 143–151. ACL.

Fellbaum, C. (1998). *WordNet: An Electronic Lexical Database*. MIT Press.

Grover, C., Tobin, R., Alex, B. and Byrne, K. (2010). Edinburgh-LTG: TempEval-2 system description. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, pages 333–336. ACL.

Im, S., You, H., Jang, H. and Nam, S. (2009). KTimeML: specification of temporal and event expressions in Korean text. In *7th Workshop on Asian Language*, pages 115–122.

Karimi-Doostan, G. (2011). Lexical categories in Persian. *Lingua*, *121*(2): 207–220.

Lesani, H. (2003). moghâyesey-e sefât-e feelêy-e fâelê va mafôlê dar zabân-e rôsê va fârsê. *pajoohesh zabanhaye khareji*, *15*: 75–84.

Llorens, H., Saquete, E. and Navarro-Colorado, B. (2010). TimeML Events Recognition and Classification: Learning CRF Models with Semantic Roles. In *Proceedings of the 23rd*

International Conference on Computational Linguistics-Coling, pages 729–733. Beijing, China.

March, O. and Baldwin, T. (2008). Automatic event reference identification. In *Australasian Language Technology Association Workshop 2008*, pages 79–87, Australia.

Montazery, M. and Faili, H. (2010). Automatic Persian WordNet Construction. In *Proceedings of the 23rd International Conference on Computational Linguistics: Posters*, pages 846–850. ACL.

Pustejovsky, J., Ingria, R., Setzer, A. and Katz, G. (2003). TimeML : Robust Specification of Event and Temporal Expressions in Text. *New Directions in Question Answering*.

Pustejovsky, J., Lee, K., Bunt, H. and Romary, L. (2010). ISO-TimeML : An International Standard for Semantic Annotation. In *Proceedings of the Seventh conference on International Language Resources and Evaluation (LREC'10)*.

Rasooli, M.S., Faili, H., Minaei-Bidgoli, B. (2011). Unsupervised Identification of Persian Compound Verbs. In *Advances in Artificial Intelligence*, pages 394–406.

Robaldo, L., Caselli, T. and Russo, I. (2011). From Italian Text to TimeML Document via Dependency Parsing. In *Proceeding of the 12th International Conference on Intelligent Text Processing and Computational Linguistics,* pages 177–187.

Sauri, R., Knippen, R., Verhagen, M. and Pustejovsky, J. (2005). Evita: A robust event recognizer for QA systems. In *LT/EMNLP*. ACL.

Shamsfard, M. (2011). Challenges and open problems in persian text processing. In *LRL*, pages 65–69.

Shamsfard, M., Hesabi, A., Fadaei, H., Mansoory, N., Famian, A., Bagherbeigi, S., Fekri, E., Monshizadeh, M. and Assi, SM. (2010). Semi Automatic Development of FarsNet; The Persian WordNet. In *Proceedings of 5th Global WordNet Conference, Mumbai, India*.

UzZaman, N. and Allen, J. F. (2010). Event and Temporal Expression extraction from raw text: first step towards a temporally aware system. *International Journal of Semantic Computing*, *4(4)*.

Verhagen, M., Sauri, R., Caselli, T., and Pustejovsky, J. (2010). SemEval-2010 Task 13 : TempEval-2. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, pages 130–134. ACL.

Measuring the Similarity between TV Programs Using Semantic Relations

Ichiro YAMADA Masaru MIYAZAKI Hideki SUMIYOSHI
Atsushi MATSUI Hironori FURUMIYA Hideki TANAKA

Japan Broadcasting Corporation, 1-10-11 Kinuta, Setagaya-ku, Tokyo, JAPAN

yamada.i-hy@nhk.or.jp, miyazaki.m-fk@nhk.or.jp,
sumiyoshi.h-di@nhk.or.jp, matsui.a-hk@nhk.or.jp,
furumiya.h-ke@nhk.or.jp,tanaka.h-ja@nhk.or.jp

ABSTRACT

This paper presents a novel method of measuring the similarity between TV programs by using summaries of the Electronic Program Guide (EPG). Most previous methods use statistics such as the TFIDF based cosine measure of word vectors, whose elements are words appearing in the summaries. However, these approaches are not effective because TV program summaries, especially short ones, do not necessarily share many words even when they have similar meanings. The proposed method generates a graph structure whose nodes are TV programs and nouns. These nouns are connected by semantic relations that are extracted from the Web automatically. The similarity between two TV programs is measured in terms of the relativeness of two TV program's nodes in the graph structure by using a random walk algorithm. Experiments showed that our method is better at measuring similarities between two TV programs compared with baseline methods.

KEYWORDS : Measuring similarity, Semantic relation, Recommendation system, Graph structure, Random walk

Proceedings of COLING 2012: Technical Papers, pages 2945–2960,
COLING 2012, Mumbai, December 2012.

1 Introduction

Japanese broadcasting stations have started services to provide viewers with previously broadcast TV programs on demand. Since these services are becoming popular, it is important for them to have an efficient way to find programs that a viewer wants to watch amidst huge program archives. Many on-demand services have the ability to present TV programs related to a selected program and rank them. This function makes it possible for viewers to find programs they would be interested in but would not have known about in advance. The TV program that the viewer selects from such a list is highly dependent on its presentation rank given by the on-demand service. FIGURE 1 shows how the number of selected programs depends on the presentation rank in a major Japanese on-demand service, NHK on demand. The higher ranked TV programs are selected more frequently than the lower ranked ones. For this reason, it is important to find out how to select related programs in huge program archives and how to rank them for better on-demand services. In order to select the related programs and rank them, NHK on-demand system measures the similarity between TV program summaries in Electronic Program Guides (EPGs) in advance. The system ranks TV programs according to their similarities to the selected program by a viewer. Since the viewer might interested in watching TV programs which are relevant, but not exactly similar in content, the technique for measuring the similarity between programs can exclude exactly similar content and select only related programs and rank them[1].

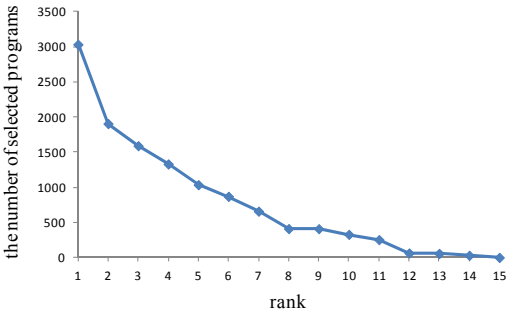


FIGURE 1 – The number of selected programs depends on the presentation rank in the NHK on demand service. (2010/9-2011/5)

NHK on-demand adopts a method of Goto et al. (2010) for measuring the similarity between TV program summaries. This method is based on word co-occurrences. Each summary is represented using words or n-grams of words as a vector, and the similarity between two summaries is calculated from their corresponding vectors. However, this method sometimes gives inappropriate similarity values for summaries (especially short ones) that have similar meanings but do not share many words. For example, the following sentences in program summaries were judged as dissimilar.

(1) This TV program conveys aspects about the treatment of diabetes.
(2) The doctor describes measures to alleviate hypertension, such as low-salt diet and drug therapy.

[1] A function for excluding exactly similar content has not been implemented on the NHK on-demand system yet.

If we knew that *low-salt diet* and *drug therapy* are *treatments* and *diabetes* and *hypertension* have the same hypernym *lifestyle-related diseases*, these sentences could be judged to have some semantic relevance.

This paper proposes a method for measuring such semantic relevance, that is, similarity between two TV program summaries by using semantic relations between nouns, such as causality and hyponymy. These semantic relations are extracted from the Web automatically and make it possible to judge the similarity of sentences appropriately even if they do not share many words. Our method generates a graph structure whose nodes are TV programs and nouns. These noun nodes are connected by the semantic relations. The similarity between two TV program summaries is calculated in terms of the relativeness of two TV program's nodes in the graph structure by using a random walk algorithm. Through experiments on ranking related TV programs provided by the NHK on demand service, we found that our method provides a proper similarity measure that shows a better correlation with human intuition than the baseline approaches we tested for comparison.

In the remainder of this paper, section 2 reviews related work on recommender systems. Section 3 introduces the methods of extracting semantic relations from Web data that we use for measuring the similarities between TV program summaries. Section 4 describes our method of measuring the similarities and ranking related TV programs. We present experimental results in section 5 before concluding the paper.

2 Related work

Many studies have been conducted on recommendation systems which offer relative products in relation to the user selected one. These systems are classified into content-based filtering and a user based approach known as collaborative filtering.

The content-based filtering is based on word co-occurrences in text relating to products, and it is commonly used in traditional information retrieval systems. Goto et al. (2010) proposed a method of TV program recommendation using content-based filtering. They used a score based on Okapi BM25 (Robertson 1999) and put weights on semantically significant words, such as named entities and compound words. Oku et al. (2007) proposed a context-aware recommendation system. Their system suggested products according to the user's context, such as time, weather, accompanying persons, and price, which change according to the situation. However, these methods rely on the assumption that more similar documents share more of the same words, even though it is true that similar pieces of content do not necessarily share the same words. Moreover, summaries in the EPG use a great variety of expressions; such summaries would not use the same word frequently to express a similar meaning.

Statistical models are useful for measuring the similarities between documents that do not share words but express similar meanings. Latent Semantic Indexing (LSI) represents terms in a latent semantic space by using the SVD of the corresponding term-document matrix (Deerwester et al. 1990). Hofmann et al. (1999) proposed PLSI which uses a latent variable model. However, these models are not good for measuring the similarities of TV program summaries, as determined in the experiment (Section 5).

Collaborative filtering is now used by several real-world recommendation systems. Amazon.com proposed item-to-item collaborative filtering (Linden et al. 2003). They produce

recommendations from customers who bought similar items to the ones in your shopping cart. Their algorithm works online and uses computation scales independent of the number of customers and number of items. Koren et al. (2009) proposed an approach based on matrix factorization; it characterizes both items and users by using vectors of factors inferred from an item rating pattern. Their system won the Netflix Prize, which was an open competition to predict user ratings for films. However, collaborative filtering suffers from the cold start problem wherein it cannot estimate the rating of new items and users, and recommendation systems for TV programs must handle new items (TV programs) frequently.

The proposed method is based on content-based filtering. We can put a high similarity value on documents that have similar meanings even if they do not share many words. These similarities between two items can be applied to the collaborative filtering approach by using not only the user's preference but similar items to the user's preference. Melville et al. (2010) proposed hybrid techniques which use both content-based and collaborative filtering. The proposed method can be used with such hybrid techniques, and it is expected to have promising results.

3 Acquisition of relations between nouns

This section describes three methods for acquiring semantic relations between nouns. The acquired relations are then used for measuring similarities between TV program summaries. We used the following four semantic relations.

- Hyponymy: A is a hypernym of B.

  Ex.) *lifestyle-related diseases* / *hypertension*

- Causality: B is caused by A.

  Ex.) *stroke* / *hypertension*

- Specialty: A is famous for B.

  Ex.) *Kyoto* / *temples*

- Material: A is made from B.

  Ex.) *beer* / *wheat*

We selected these relations because they are effective at capturing the VOD user's attention when it comes to TV program suggestions. For example, someone who is interested in *Kyoto* would probably like TV programs concerning *temples*.

We also use the entity-attribute-value relation. It can be considered that the entity word and the value word have the relation of attribute. For example, in the relation "*hypertension* / *management* / *weight loss*," the relation between *hypertension* and *weight loss* can be considered to be *management*.

In the following subsection, we describe the methods of acquiring these semantic relations.

3.1 Relation acquisition from Wikipedia

For the hyponymy relation and entity-attribute-value relation acquisition, we used an open-source software[2] based on the extraction methods of Sumida et al. (2008) and Yamada et al. (2010). Sumida et al. (2008) proposed a method of automatically acquiring hyponymy relations of nouns from Wikipedia. They focused on the hierarchical layout of articles in Wikipedia, which is made of titles, sections, sub-sections, itemizations, and so on. For example, in the article titled *lifestyle-related diseases*, there are itemizations *hypertension*, *diabetes*, and *history*. Relations such as the one between *lifestyle-related diseases* and *hypertension* and the one between *lifestyle-related diseases* and *diabetes* can be considered to be hyponymy relations, but the one between *lifestyle-related diseases* and *history* cannot be considered a hyponymy relation. Their method first extracts hyponymy relation candidates from the hierarchical structure of Wikipedia. The candidates are then classified into plausible and implausible ones by using a support vector machine (SVM) classifier.

Sumida et al. (2008) also proposed another method for hyponymy acquisition that exploits other information sources: the first sentence of Wikipedia articles, which is regarded as the article's definition, and category names. This method generates hyponymy relation candidates in which the hyponymy corresponds to the article titles and the hypernym comes from either of the information sources. The candidates are classified in the same process of analyzing a hierarchical layout.

We expanded their hyponymy acquisition method to generate entity-attribute-value relations (Yamada et al. 2010). We confirmed our assumption that if two words located in the layout structure can be regarded as a hyponymy relation, the article title, hypernym word and hyponym word can be interpreted as an entity, the attribute, and its attribute value independently. Take, for example, the hyponymy relation *management* and *weight loss* from the Wikipedia article *hypertension*; it can be interpreted that the entity *hypertension*'s *management* (attribute) is *weight loss* (value).

3.2 Relation acquisition from Web text

Causality, specialty, and material relations are extracted by using a semantic relation acquisition service provided by the ALGIN forum[3] in Japan. This service is based on a method proposed by Stijn et al. (2009) and can extract large-scale relations between nouns from 6 million Japanese Web pages by inputting a small number of seed patterns. The service learns linguistic patterns that express each relation such as "X gives rise to Y" for causality with semantic word classes of X and Y acquired by large-scale clustering (Kazama et al. 2008). For example, if we know that the pattern "X gives rise to Y" expresses causality and the phrase "*hypertension* gives rise to *stroke*" appears frequently on the Web, the relation between *hypertension* and *stroke* will be regarded as causal. Moreover, *heart attack*, which belongs to the same class as *stroke*, can be also considered to have a causality relation with *hypertension*.

However, the relations acquired by this method include some obvious errors and ambiguities. For example, the method erroneously regards the relation between *disease* and *stroke* as being causal from the pattern "The *disease* gave rise to the *stroke*". This is because the word *disease* belongs to the same class of *hypertension*. To avoid this error, we generated a stop-word list manually to

[2] http://alaginrc.nict.go.jp/hyponymy/index.html
[3] http://alaginrc.nict.go.jp

exclude these erroneous relations from the results of the semantic relation acquisition service.

3.3 Hypernym acquisition using compound nouns

A suffix of a compound noun sometimes becomes a hypernym of the original noun in a head-final language (Kuroda et al. 2009). For example, the suffix *disease* is considered to be the hypernym of *lifestyle-related diseases*. Since our target language in the experiment was Japanese, we first decomposed a compound noun into a sequence of nouns using a morphological analyzer and then checked whether the suffix sequence was a valid hypernym of the compound noun. We judged that the suffix is a valid hypernym if it is registered in a dictionary. In the experiment described in section 5, we used the Japanese WordNet (Bond et al. 2012) as the dictionary.

3.4 Acquired relations

We acquired hyponymy and entity-attribute-value relations by using the relation extraction software mentioned in section 3.1, targeting 5 years of Wikipedia dump data from 2007 to 2011. We also extracted causality, specialty, and material relations by using the semantic relation acquisition service mentioned in section 3.2, by inputting a few seed patterns, and acquired reliable patterns. We extracted hypernyms by using the suffixes of nouns appearing in the extracted relations and TV program summaries. TABLE 1 shows examples of the acquired relations.

We randomly sampled 200 of the automatically acquired relations respectively, and one of the authors checked whether the relations were valid or not. TABLE 2 shows the number of acquired

Word *X*	Relation	Word Y
hikkigu (writing material)	hyponymy	shapu-pen (mechanical pencil)
eiga (movie)	hyponymy	Star Trek
ice cream	hyponymy	vanilla ice cream
allergen	causality	kikanshi zensoku (asthma bronchiale)
El Nino	causality	ondo josho (rising water temperatures)
Canada	specialty	winter sports
Chiang Mai	specialty	bukkyo-jiin (Buddhist temple)
miso	material	soybean
pannacotta	material	coconut
John Woo	directed film	Red Cliff
J. D. Salinger	work	A boy in France

TABLE 1 – Examples of acquired relations between nouns.

Relation	Number of relations	Accuracy
hyponymy(Wikipedia)	8,591,469	90.0%*
hyponymy(suffix)	1,347,382	82.5%
entity-attribute-value	5,213,455	94.0%*
causality	77,636	75.0%
specialty	183,093	49.0%
material	49,711	73.0%

TABLE 2 – Number of acquired relations and accuracies. * indicates accuracies obtained from the original literature.

Relation	Coverage
All relations	72.8% (68,726/94,456)
Relations other than the hyponymy acquired using suffix information.	47.7% (45,042/94,456)

TABLE 3 – Coverage rate of the acquired relation relative to nouns in the TV program summaries.

relations and the evaluation results.

The total number of relations was 15,462,746, and 3,458,913 nouns appeared in the relations. We checked how these nouns covered the TV program summaries. We picked 25,769 summaries containing 94,456 nouns whose TV programs were available from NHK on demand. TABLE 3 shows the coverage of the acquired relation.

The acquired relations contained 6% ~ 51% errors (TABLE 2). TABLE 3 indicates the nouns in all relations cover 72.8% of the TV program summaries, but that is overstated because the hyponymy relations acquired by using suffix information target the nouns of the summaries. Relations other than the hyponymy relations by suffix cover 47.7% of nouns of the summaries. Although there is room for improvement in accuracy and coverage, we can acquire a huge number of relations and this holds promise for measuring similarities between summaries.

4 Proposed method for measuring similarities between summaries

Here, we describe the proposed method for measuring the similarity between the TV program summaries in NHK on demand using the acquired relations. The average number of characters in the summaries is about 170, and the average number of nouns included in the summaries is 26. These nouns are used as a clue to measure the similarities. The proposed method first generates graph structures which include the TV programs as their nodes. Then it measures the similarity between summaries by measuring the strength of binding of two TV program nodes on the graph. The following subsections describe each step of measuring similarities.

4.1 Generating graph structures

Graph structures are generated from TV program summaries as follows:

1. The TV program names and each noun appearing in the TV program summaries are put on graph nodes.
2. The TV program node and nodes of summary nouns are connected by non-directional edges. The weight to put on each edge between program node p_i and noun node n_j is defined as follows.

$$e(p_i, n_j) = tf(n_j) idf(n_j) / Z_{p_i} \quad (1)$$

   Here, $tf(n_j)$ represents the frequency of noun n_j, $idf(n_j)$ is the inverse document frequency of noun n_j in the summary of TV program p_i, and Z_p is a normalization factor calculated as the sum of the weights of edges which start from node p.
3. The non-directional edges between noun nodes are constructed from the automatically acquired relations between nouns. If two nouns are related, their nodes are connected. For example, if we acquired the hyponymy relation "*lifestyle-related diseases* / *hypertension*", the causality relation "*hypertension* / *smoking*", and the material relation "*smoking* / *Tabaco*",

the edges "*lifestyle-related diseases* ↔ *hypertension* ↔ *smoking* ↔ *Tabaco*" are constructed. The weights for the edges are defined as

$$e(n_i, n_j) = 1/Z_{n_i} \tag{2}$$

Here, Z_n is the normalization factor calculated as the sum of the weights of edges which start from node n.

4.2 Measuring similarities

The next step measures the similarities between summaries by estimating how much one TV program node is related to another program node on the graph. We use a Markov chain theory, called Green Measures (Kemeny et al. 1966), which is a random walk algorithm to measure the similarity of nodes. This method uses a matrix M whose element m_{ij} corresponds to the transition probability from node i to node j. Here, $\Sigma_j m_{ij}=1$. We use the normalized weight for edge defined by equation (1) or (2) for the element m_{ij}. That is, $m_{ij} = e(p_i, n_j)$ if there is an edge between program node p_i and noun node n_j, $m_{ij}=e(n_i, n_j)$ if there is an edge between noun node p_i and noun node n_j, and $m_{ij}=0$ if there is no edge between node i and node j. The Green matrix G is defined as

$$G = \sum_{t=0}^{\infty} (M^t - M^{\infty}) \tag{3}$$

Here, M^t corresponds to the transition matrix of t steps in the random walk. It is known that the Markov chain converges if the chain is both irreducible and aperiodic. Because the chain of our generated graph structure satisfies those conditions, M^t converges exponentially to M^{∞}. The value of element g_{ij} in G indicates how much node i is related to node j. Using the Green matrix G, Yann et al. (2007) proposed a score $S(i, j)$ to indicate the relativeness between two nodes i and j:

$$S(i, j) = g_{ij} \log(1/v_j) \tag{4}$$

Here, v_j is the j-th element of vector v, which is a unique invariant probability measure of the matrix M where $vM=v$. Moreover, against any measure μ where $\Sigma\mu_j=1$, μM^n converges to v as $n \rightarrow \infty$. The logarithmic term $\log(1/v_j)$ can modify the g_{ij} which corresponds to the relativeness score between node i and node j when the value v_j is high. This logarithmic term works like the *idf* value which is used in information retrieval.

We devised two methods for measuring the similarity between two TV program summaries using the relativeness score (eq. (2)) between two nodes. The first method directly uses the relativeness score of the target node directory. The similarity between program nodes p_i and p_j is defined as follows.

$$S_{direct}(p_i, p_j) = S(p_i, p_j) \tag{5}$$

This is the relativeness score of node p_j when the random walk starts from node p_i.

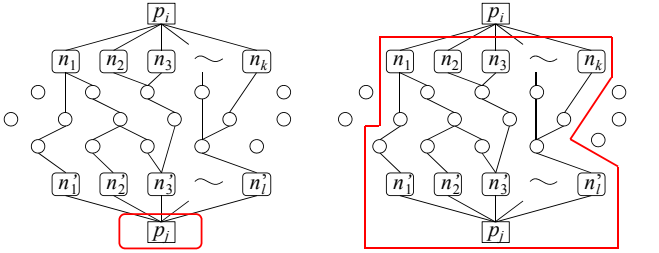


FIGURE 2 – Two methods of measuring similarities between TV program summaries.

The second method uses the scores of all nodes on the route from p_i to p_j. The similarity is defined as follows.

$$S_{related}(p_i, p_j) = \sum_{v \in node(p_i, p_j)} S(p_i, v) \tag{6}$$

Here, $node(p_i, p_j)$ is the set of nodes on the route from p_i to p_j. FIGURE 2 shows the concept of measuring similarities between the summaries of TV programs by the two methods.

5 Experiments

To confirm the effectiveness of the proposed methods, we conducted experiments on measuring the similarities of TV program summaries by using the two proposed methods and four baseline methods. To make the test data, we sampled 352 summaries with the following restrictions.

- All the summaries had different TV program titles
- More than one related TV program were presented for the selected summary in the NHK on-demand service.

The average number of related TV programs to the sampled 352 TV programs was 10.4.

Next, three judges who were not authors ranked the relativeness of the related TV programs against 352 samples. We used Spearman's rank correlation to confirm whether the three judge's rankings were in reasonable agreement. The correlation, which takes into account tie scores of ranks is defined as follows:

$$\rho = T_x + T_y - \sum D^2 \Big/ 2\sqrt{T_x T_y} \tag{7}$$
$$T_x = \left(N^3 - N - \sum_{i=1}^{n_x} (t_i^3 - t_i)\right) \Big/ 12$$
$$T_y = \left(N^3 - N - \sum_{j=1}^{n_y} (t_j^3 - t_j)\right) \Big/ 12$$

Here, D and N indicate the difference between two ranks and the number of related programs. n_x and n_y are the number of tie ranks, and t_i and t_j are the ranks of n_x and n_y. The average correlation between the ranks by the three judges was 0.565, which indicates moderate agreement. We generated the data by arranging these related programs in descending order of average rank and regarded this manually-generated data as the gold-standard ranking data.

We used his gold-standard ranking data as a reference to evaluate the ranking results.

5.1 Baseline methods

Baseline method 1: Okapi BM25

Goto et al. (2010) proposed a method for measuring the similarity between two summaries of TV programs by using Okapi BM25. Okapi BM25 ranks documents according to the relevance to a given query. They substituted the documents and query with summaries. The similarity score $S_{BM}(p_1,p_2)$ between two summaries p_1 and p_2 is defined by the following equation.

$$S_{BM}(p_1,p_2)=\sum_{n\in p_1} idf(n)\cdot\frac{tf_{p_2}(n)\cdot(k+1)}{tf_{p_2}(n)+k\cdot(1-b+\frac{|p_2|}{avgdl})}\cdot\frac{(k'+1)tf_{p_1}(n)}{k'+tf_{p_1}(n)} \quad (8)$$

Here, $tf_p(n)$ is a frequency of noun n in the summary p, $idf(n)$ is the inverse document frequency of n, $|p|$ is the length of the summary p, $avgdl$ is the average length of the summary, and k and k' and b are parameters for which we used k=3.0, k'=100.0, and b=0.75 from the original literature.

Baseline method 2: Cosine with nouns appearing in the summary

Each summary is represented by a vector whose elements are nouns appearing in the summary. The weight $w_{TFIDF}(n)$ for each element is defined by TFIDF.

$$w_{TFIDF}(n)=tf_p(n)\cdot idf(n) \quad (9)$$

The similarity between two summaries p_1 and p_2 is defined by the cosine value of these vectors.

$$S_{TFIDF}(\vec{p}_1,\vec{p}_2)=\frac{\vec{p}_1\cdot\vec{p}_2}{|\vec{p}_1||\vec{p}_2|} \quad (10)$$

Baseline method 3: Cosine with related nouns

This method also represents each summary by a vector. The elements of the vector are composed by nouns appearing in the summary and nouns that are related with any nouns in the summary. For example, if *stroke* appears in a summary, we will add *hypertension* as an element of the vector because *stroke* and *hypertension* have a causality relation. The weight for each noun appearing in the summary is calculated by equation (9). The weight for the expanded noun n_{rel} is defined as follows.

$$w_{rel}(n_{rel}) = w_{TFIDF}(n) / N_{rel}(n) \quad (11)$$

Here, $N_{rel}(n)$ is the number of relations of noun n. The similarity between two summaries is calculated by equation (10).

Baseline method 4 : PLSI

Baseline method 4 uses a statistical latent class model, called PLSI (Hofmann et al. 1999), which associates an unobserved class variable $z \in Z = \{z_1, \ldots, z_k\}$ with each observation of noun $w \in W = \{w_1, \ldots, w_M\}$ in a document $d \in D = \{d_1, \ldots, d_N\}$. The distribution of probability $P(z|d)$ is estimated for each class z and document d by using the EM algorithm. The similarity between two summaries is calculated by computing the distance (Jensen-Shannon divergence) between two probability distributions. The Jensen-Shannon divergence between two probabilities, $P(z|d_1)$ and $P(z|d_2)$, can be calculated as follows.

$$D_{JS}(P(z \mid d_1) \| P(z \mid d_2) = \frac{1}{2}(D_{KL} P(z \mid d_1) \| \frac{P(z \mid d_1) + P(z \mid d_2)}{2}) + D_{KL} P(z \mid d_2) \| \frac{P(z \mid d_1) + P(z \mid d_2)}{2})) \quad (12)$$

Here, D_{KL} indicates the Kullback-Leibler divergence:

$$D_{KL}(P(z \mid d_1) \| P(z \mid d_2) = \sum P(z \mid d_1) \log \frac{P(z \mid d_1)}{P(z \mid d_2)} \quad (13)$$

By using the latent class, it becomes possible to put a non-zero similarity value on summaries that express similar meanings even if they do not share words.

5.2 Experimental results

Targeting the selected summaries for 352 TV programs, we conducted an experiment on ranking related TV programs using the proposed methods and four baseline methods. TABLE 4 shows Spearman's rank correlation with the gold-standard data. In baseline method 4, we tried the following parameter values for the number of unobserved classes z and the temperature parameter β in the process of the EM algorithm and selected z=200 and β=0.75, which gave the best result.

Methods	Rank correlation
Baseline 1 (Okapi-BM25)	0.370
Baseline 2 (cosine with nouns appearing in the summary)	0.350
Baseline 3 (cosine with related words)	0.371
Baseline 4 (PLSI)	0.190
Proposed 1 (using the score of related program node)	0.351
Proposed 2 (using the scores of all nodes on the route to the related program node)	0.423

TABLE 4 – Evaluation result for each method.

z ={10, 50, 100, 200, 300, 400, 500, 1000}

β ={0.6, 0.65, 0.7, 0.75 0.8, 0.85, 0.9, 0.95, 1.0}

The results for proposed method 2, which uses the scores of all nodes on the route between the program nodes, were far better than those of the other methods (TABLE 4). On the other hand, proposed method 1 was not better than baseline methods 1 ~ 3. The score of the node which is connected directly with the start node of the random walk is much larger than one of the nodes which are connected indirectly with the start node. The scores for the related program nodes, which are far from the target program node, are too small to compare with each other. This arises from the shortage of noun relations. If we can acquire enough noun relations, we could avoid this problem. The correlation of baseline method 4 which uses PLSI is much lower than the other methods. This is because the results of the clustering by PLSI were not useful for making TV program recommendations. For example, the words *politics*, *economics*, and *international* belong

[Target TV program title] Marutoku-magazine, exercise ~ hip joint and legs

[Nouns appearing in the corresponding summary] hip joint, stretch, movement, you, easily, range of movement, up, posture, around hip joint, five minutes, velocity, length of stride, exercise, muscle, coverage, one day

Related TV program titles and nouns appearing in the corresponding summary	Gold-standard data (score)	Proposed method 2 (score)	Baseline method 2 (score)
Marutoku-magazine, exercise ~ arm and back [Nouns] back, body, arm, five minutes, condition, tension, everyday, you, head, exercise, posture, muscle, one day	1 (1.333)	1 (0.913)	1 (0.328)
Tameshite-Gatten, Banana revolution ~ declaration of new ingredient [Nouns] banana, fruit, vegetable, cooking, taste, exposure, nourishment, full marks, hand, product, No. 1 of consumption, clear, world, shipping, I, easy, expensive ingredient, ability, ingredient, Gatten's way of cooking bananas, majority, method	2 (1.667)	2 (0.582)	2 (0.0)
Fudangi-no-onsen, Aomori Shimofuro hot spring [Nouns] therapeutic bath, body, bitter cold, Muromachi era, scene, strait, role, people, two, shared hot spring, fishing herring, before World War II, large spa, Yasushi Inoue, place, home town, fist, exchange, information, fisherman, friend, Tsugaru Straits, novel, variety, famous, importance, Shimofuro hot spring, hot spring, core, new hot water	3 (3.333)	3 (0.561)	2 (0.0)
Asaichi, Japan navigation ~ Kyoto [Nouns] Kyoto, Daihachi car, talk, huge, Daikaku temple, autumn, surface of water, Osawa pond, travel, kimono, Maho, vegetable farmer, autumn leaves, fantastic, together, meeting, vegetarian dish, popular, Kyoto vegetable, lighting-up, scene, Hisako Noguchi, 82 years old, temple, Sagano, actor, finding antique kimono, full of autumn leaves, 13th, airiness, world, sight to see, temple master, ancient city	4 (3.667)	4 (0.203)	2 (0.0)

TABLE 5 – Ranking results of proposed method 2 and baseline method 2. All titles and nouns are translated from Japanese nouns.

to the same class, which resulted in miss-selection of the related program for *politics*. The clustering based approach sometimes is affected by from the granularity of each cluster.

TABLE 5 shows a sample of the rankings of proposed method 2 and baseline method 2. The scores of baseline method 2 were zero for three programs, whose summaries did not share words with that of the target program. In contrast, proposed method 2 properly scored these programs. This caused its results to have a higher correlation with the gold-standard data.

5.3 Effectiveness of the using relations

We randomly sampled from all acquired relations in order to investigate the effectiveness of the relations. FIGURE 3 shows the correlation versus the number of sampled relations. Here, the correlation is 0.400 when the number of relations is zero. This value is higher than those of the baseline methods. In the case that we do not use any relations, the edges in the graph structure are composed of one between the program node and the noun node appearing in the program summary. This arises from a heuristic that "the words in the same program summary are similar to each other". This result shows that the more relations we used, the higher the correlation became. We used about 15.5 million relations in this experiment, and we hope we can get higher correlations if we use more relations.

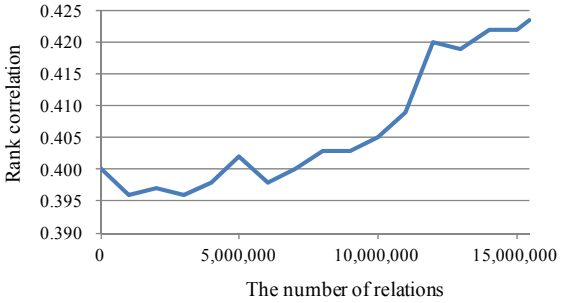


FIGURE 3 – The correlation depended on the number of sampled relations.

5.4 Considerations regarding the edge weighting

In the process of generating graph structures, each edge between the program node and the noun node is weighted by the TFIDF value in equation (1). A random walk tends to move to the node whose TFIDF value is high and which is considered important. However, if a noun node has links to the other nouns appearing in the same summary, the random walk will move on the noun node with higher probability because there are several routes to get to the node.

We experimented with ranking related TV programs by using proposed method 2 without an edge weighting by TFIDF. As a result, the rank correlation value was 0.427, which is comparable to the result of using the weighting. This means that the graph structure implicitly defines the node's importance.

Conclusion

We proposed a method for measuring the similarity between two TV program summaries. The method generates a graph structure whose nodes are composed of TV programs and nouns appearing in the corresponding summary and whose edges are generated from four kinds of relations automatically acquired from the Web and Wikipedia. The similarity between two TV program summaries is calculated on the basis of the relativeness of the two TV program's nodes in the graph structure by using a random walk algorithm. Experiments confirmed that our method provided a proper similarity measure that showed a better correlation with the gold-standard data than the baseline approaches. The experiments using several relations indicated that we would get a higher correlation if we used more relations. Furthermore, we confirmed that we did not need to use the edge weighting by TFIDF because the graph structure implicitly defines the importance for each node.

We used four types of relations in the experiments. Enlarging the number of relations will be of further help in measuring the similarity. Moreover, the relations in the experiments contained 6% ~ 51% errors and covered 47.7% of the nouns appearing in the TV program summaries. In the future, we will determine whether using more relations with higher accuracy and broader coverage.

References

Bond, F., Baldwin, T., Fothergill, R. and Uchimoto, K. (2012). Japanese SemCor: A Sense-tagged Corpus of Japanese, In *Proceedings of the 6th International Conference of the Global WordNet Association* (*GWC*).

De Saeger, S., Torisawa, K., Kazama, J., Kuroda, K. and Murata, M. (2009). Large Scale Relation Acquisition using Class Dependent Patterns, In *Proceedings of the IEEE International Conference on Data Mining* (*ICDM'09*), pp.764-769.

Deerwester, S., Dumais, S. T., Furnas, G. W., Landauer, T. K. and Harshman, R (1990). Indexing by latent semantic analysis. *Journal of the American Society for Information Science*, 41, 391-407.

Goto, J., Sumiyoshi, H., Miyazaki, M., Tanaka, H., Shibata, M. and Aizawa, A. (2010). Relevant TV Program Retrieval using Broadcast Summaries, In *Proceedings of ACM on Intelligent User Interfaces*(*IUI*), pp.411-412.

Hofmann, T. (1999). Probabilistic Latent Semantic Indexing, In *Proceedings of SIGIR'99*, ACM Press, pp.50-57.

Kazama, J. and Torisawa, K. (2008). Inducing Gazetteers for Named Entity Recognition by Large-scale Clustering of Dependency Relations, In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies* (*ACL-08: HLT*), pp. 407-415.

Kemeny, J. G., Snell, J. L. and Knapp, A.W. (1966). Denumerable Markov chains, Van Nostrand Company.

Koren, Y., Bell, R. and Volinsky, C. (2009) Matrix Factorization Techniques for Recommender

Systems, *IEEE Computer*, pp.42-49.

Kuroda, K., Murata M. and Torisawa, K. (2009). When nouns need co-arguments: A case study of semantically unsaturated nouns, In *Proceedings of the 5th International Conference on Generative Approaches the Lexicon 2009*.

Linden, G., Smith, B. and York, J. (2003). Amazon.com recommendations, *IEEE Internet Comput*., vol7, no.1, pp. 76-80.

Melville, P. and Sindhwani, V. (2010). Recommender Systems, Encyclopaedia of Machine Learning, Springer.

Oku, k., Nakajima, S., Miyazaki, J. and Uemura, S. (2007). Context-Aware Recommendation System Based on Context-Dependent User Preference Modeling, *IPSJ journal,* vol. 48, SIG11 (TOD_34), pp. 162-176. (in Japanese)

Robertson, S. and Walker, S. (1999). Okapi/ Keenbow at TREC-8, In *Proceedings of TREC-8*, pp151-162.

Sumida, A., Yoshinaga. N. and Torisawa, K. (2008). Boosting Precision and Recall of Hyponymy Relation Acquisition from Hierarchical Layouts in Wikipedia, In *Proceedings of the sixth Language Resources and Evaluation Conference* (*LREC 2008*), pp.2462-2469.

Yamada, I., Hashimoto, C., Oh, J.-H., Torisawa, K., Kuroda, K., De Saeger, S., Tsuchida, M. and Kazama, J (2010). Generating Information-Rich Taxonomy from Wikipedia, In *Proceeedings of the 4th International Universal Communication Symposium* (*IUCS 2010*), pp. 96-103.

Yann, O. and Pierre S. (2007). Finding Related Pages Using Green Measures: An Illustration with Wikipedia, In *Proceedings of the Twenty-Second AAAI Conference on Artificial Intelligence*, pp.1427-1433.

RelationListwise for query-focused multi-document summarization

Wenpeng Yin Lifu Huang Yulong Pei Lian'en Huang
The Shenzhen Key Lab for Cloud Computing Technology & Applications (SPCCTA)
Shenzhen Graduate School
Peking University, Shenzhen 518055, P.R. China
{mr.yinwenpeng,warrior.fu,paul.yulong.pei}@gmail.com, hle@net.pku.edu.cn

ABSTRACT
Most existing learning to rank based summarization methods only used content relevance of sentences with respect to queries to rank or estimate sentences, while neglecting sentence relationships. In our work, we propose a novel model, RelationListwise, by integrating relation information among all the estimated sentences into listMLE-Top K, a basic listwise learning to rank model, to improve the quality of top-ranked sentences. In addition, we present some unique sentence features as well as a novel measure of sentence semantic relation, aiming to enhance the performance of training model. Experimental results on DUC2005-2007 standard summarization data sets demonstrate the effectiveness of our proposed method.

KEYWORDS: Query-focused multi-document summarization, Listwise, Sentence relation.

Proceedings of COLING 2012: Technical Papers, pages 2961–2976,
COLING 2012, Mumbai, December 2012.

1 Introduction

In this paper, we focus on the task of producing extraction-based query-focused multi-document summaries given a collection of documents, which is usually considered as a sentence ranking problem. Typically, ranking methods calculate the combinational effects of various features which are designed to identify the different aspects of sentences and/or their relevance to queries. Yet so far not much attention has been paid to it. Most commonly, the features are simply combined by a linear function in which the weights are assigned manually or tuned experimentally. In the past, machine learning approaches have been successfully applied in extractive summarization (Ouyang et al., 2007; Shen and Li, 2011), and a new research branch named “learning to rank" has emerged. Its objective is to explore how the optimal weights can be obtained automatically by developing learning strategies. However, previous work mainly considered the content relevance of sentences with respect to certain query while ignoring the relationships among sentences. In this paper, we try to study how to use sentence relatedness to improve the performance of a ranking model. Further, we notice that many learning to rank algorithms have been proposed in recent literature, and these algorithms can be categorized into three types: pointwise, pairwise, and listwise approaches. The pointwise and pairwise approaches transform ranking problem into regression or classification on single object and object pairs respectively, while neglecting the fact that ranking is a prediction task on a list of objects. In listwise approach, object lists instead of object pairs are used as instances in learning, and the major task is how to construct a listwise loss function, representing the difference between the ranking list output by a ranking model and the ranking list given as ground truth. Experimental results showed that listwise approach usually outperforms pointwise and pariwise approaches (Cao et al., 2007; Qin et al., 2008). Accordingly, we mainly concentrate on developing listwise learning to rank in our summarization task.

More exactly, taking into account the specific scenario of summarization, it’s better to base our work on a variant of basic listwise training model: ListMLE Top-K presented in (Xia et al., 2009). It’s because that we usually only need to select a small amount of sentences to construct a summary. ListMLE Top-K , a modification of basic listwise algorithm for more suitability in many real ranking problems where the correct ranking of the entire permutation is not needed, could help us to improve the ranking accuracies of top-K sentences. Based on that, our novel RelationListwise function, having absorbed sentence affinity information, is formed to learn the optimal feature weights.

Apparently, how to design appropriate sentence features and measure sentence similarity matter greatly in influencing the system performance. In most existing approaches about feature design, the authors tended to only consider factors that were supposed to reflect the bias of sentences towards a query while neglecting such a possibility: overestimating the relationship with a query might lead to the competitiveness among relevant sentences, which would eventually do harm to the summary quality. For this reason, some extra processing was usually conducted during sentence selection because some top-ranked sentences usually can not be used to construct a high-quality summary directly. For example, (Shen and Li, 2011) defined a total of 20 sentence features for its ranking SVM model. Whereas, those features were produced by paying no attention to avoiding similar sentences to get close scores. Therefore, in order to keep low information redundancy in the finally generated summary, those authors had to apply diversity penalty algorithm to update the sentence orders which resulted from the training function directly. Namely, some previous work had to take two steps to construct a high-quality summary, including initial sentence ranking coming from ranking model

and sentence order adjustment. Hence, a strong motivation comes to our mind: designing sentence features considering the query-biased factors as well as the competitiveness coming from similar sentences, aiming to produce a high-quality summary through directly selecting some top-ranked sentences.

As for sentence similarity, existing literatures have presented various methods to deal with it. Nevertheless, hard matching of words is commonly a primary obstacle in judging whether two sentences are semantically related. For example, while $wonderful$ and $amazing$ are semantically close, they are treated completely different in many existing methods, such as tf-isf (term frequency and inverse sentence frequency) based cosine measure. Motivated by that, we utilize Log-Bilinear Document Model, proposed in (Maas et al., 2011), to achieve the identification of semantically similar words. As a result, sentences with no same words but similar ones could also be considered having a certain degree of semantic similarity. Extensive experiments on DUC2005-2007 standard summarization data sets are performed and their results point out the good performance of RelationListwise in this task.

Since our proposed training model is on the basis of sentence similarity and sentence features, in following sections, after giving related work in Section 2, we first elaborate how to derive sentence similarity in Section 3 and biased features in Section 4, respectively. Then, detailed description of using sentence similarity to improve listMLE top-K is presented in Section 5. Section 6 shows training data generation. Experiments and results are given in Section 7.

2 Related work

We first introduce some supervised learning approaches applied in query-biased summarization. Then, some typical work about feature design and sentence similarity follows.

Supervised learning approaches have been successfully applied to query/topic-biased summarization. (Zhao et al., 2005) applied the Conditional Maximum Entropy, a classification model, on the DUC 2005 query-based summarization task. (Ouyang et al., 2007) used support vector regression (SVR), a pointwise ranking algorithm, to relate the "true" score of the sentence to its features. (Jin et al., 2010) presented a systematic study of comparing different learning to rank algorithms and comparing different selection strategies for multi-document summarization. Whereas, it focused on the simple comparison of some basic models with no optimization. (CHALI and HASAN, 2011) had deeply investigated and compared the effects of using different automatic annotation techniques on different supervised learning approaches, including SVMs, HMMs, CRFs, and MaxEnt, in the domain of query-focused multi-document summarization. (Shen and Li, 2011) explored the use of ranking SVM, a pairwise learning to rank model, for obtaining credible and controllable solutions for feature combinations. Our main contributions not only lie in our unique design of sentence features and sentence similarity measure, more importantly, we integrate sentence relationships with listwise to improve the ranking model while above literatures ignored the relation information among those sentences

With regard to feature design, (Li et al., 2009) treated summarization as a supervised sentence ranking process, where coverage, balance and novelty properties were incorporated. Whereas, it focused on generic summarization rather than query-biased situation. (Wan et al., 2007) gave explicit definitions of biased information richness and novelty, then, it proposed to compute biased information richness using manifold-ranking process (Zhou et al., 2004), and a modified MMR algorithm was applied to keep low information redundancy in generated summaries. In (Wei et al., 2008), authors proposed query-sensitive sentence similarity. Only the

overlapping of topic-relevant contents was penalized when applying MMR algorithm. In addition, clustering techniques were commonly adopted to identify different and novel aspects of documents (Wan and Yang, 2008). The method presented in (Li et al., 2010) is slightly similar with our work for it considered novelty, coverage and balance wholly. However, sentence features of existing literatures were usually acquired asynchronously. For instance, $novelty$ and $balance$ in (Li et al., 2010) were achieved as an optimization process after authors have identified part of high-quality sentences through the effects of other features.

Calculating sentence similarity appears in many applications. tf-isf based cosine measure is widely used to determine the lexical similarity of two sentences. Whereas, high dimensionality and high sparsity usually lead to disappointing performance. (Erkan, 2006) proposed a graph-based sentence ranking model: Biased LexRank, where edge weight or called sentence similarity was acquired using generation probability between two sentences based on a (unigram) language model. (Islam and Inkpen, 2008) determined sentence similarity by combining string similarity, semantic similarity and common-word order similarity with normalization. The similarity between two short text snippets in (Quan et al., 2010) was calculated based on their common terms and their distinguishing terms relationship which was discovered by examining their probabilities under each topic. Some literatures opted to first deal with word similarity calculation, then combine the result with sentence structure information. For example, (Li et al., 2006) first derived word semantic similarity from a lexical knowledge base, modeling common human knowledge about words in a natural language, and a corpus, adapting to the specific application area. Secondly, it considered the impact of word order on sentence meaning. The derived word order similarity measured the number of different words as well as the number of word pairs in a different order. In (Zhang et al., 2011), word similarity only considered the spellings and ignored the semantic meanings of words. Structure information considered the orders of words and the distances between words, and it ignored the syntactic information of sentences. (Yin et al., 2012) used a similar way with (Quan et al., 2010) to determine word relatedness while its structural similarity of sentences was acquired via longest common subsequence (LCS), weighted longest common subsequence (WLCS) and skip-bigram co-occurrence statistics, respectively.

3 New measure of sentence similarity

Hard matching between words has long been an obstacle in determine the relatedness of two sentences. For example, considering following two sentences:

s_1 : employee enjoy happy *holiday* $\quad$ s_2 : employee enjoy happy *vacation*

where words $holiday$ and $vacation$ would be treated with no relation in traditional VSM based cosine measure. Whereas, they are semantically related very much in the real context. Hence, in our perspective, before computing sentence similarity, we should first solve this problem: identifying the semantic relatedness of words.

3.1 Capturing semantic similarities of words

Authors in (Maas et al., 2011) presented an algorithm to acquire word semantic similarity through learning word vectors via an unsupervised probabilistic model of documents. While it is common to represent words as indices in a vocabulary, but this fails to capture the rich relational structure of the lexicon. Vector-based models do much better in this regard. They

encode continuous similarities between words as distance or angle between word vectors in a high-dimensional space. Next, we briefly introduce that algorithm.

For a document (e.g., d), a probabilistic model is constructed using a continuous mixture distribution over words indexed by a multi-dimensional random variable θ, then the probability of d is determined using a joint distribution over d and θ. As many common treatments did, the algorithm also puts an assumption of independence for words given θ. The probability of a document d is as follows

$$p(d) = \int p(d,\theta)d\theta = \int p(\theta)\prod_{i=1}^{N} p(w_i|\theta)d\theta \tag{1}$$

where N is the number of words in d and w_i is the i^{th} word. θ has a Gaussian prior.

Then define the conditional distribution $p(w_i|\theta)$ using a log-bilinear model (Maas and Ng, 2010) with parameters R and b. R is virtually a word representation matrix $R \in \mathbb{R}^{(\beta\times|V|)}$ where the β-dimensional vector representation of each word w in vocabulary V corresponds to that word's column in R, i.e., $\phi_w = R_w$. The random variable θ is also a β-dimensional vector ($\theta \in \mathbb{R}^{\beta}$), indicating the weights of the β dimensions of words' representation vectors. In addition, a bias b_w is introduced for each word to capture differences in overall word frequencies. The energy assigned to a word w, given these model parameters, is

$$E(w;\theta,\phi_w,b_w) = -\theta^T\phi_w - b_w \tag{2}$$

After normalization, we obtain the distribution $p(w|\theta)$,

$$\begin{aligned} p(w|\theta;R,b) &= \frac{\exp(-E(w;\theta,\phi_w,b_w))}{\sum_{w'\in V}\exp(-E(w';\theta,\phi_{w'},b_{w'}))} \\ &= \frac{\exp(\theta^T\phi_w + b_w)}{\sum_{w'\in V}\exp(\theta^T\phi_{w'} + b_{w'})} \end{aligned} \tag{3}$$

Apparently, for a given θ, a word w's occurrence probability is related to how closely its representation vector ϕ_w matches the scaling direction of θ. Finally, maximum likelihood learning is exploited for this model when given a set of unlabeled documents D. In maximum likelihood learning we maximize the probability of the observed data given the model parameters. Here, we omit the learning details[1].

Having obtained vector representations of words, we could determine the semantic relatedness (SR for short) of two terms (e.g., w_1 and w_2):

$$SR(w_1,w_2) = \phi_{w_1}\cdot\phi_{w_2} \tag{4}$$

3.2 Sentence similarity identification

Based on word relatedness, we construct a word connectivity graph, and use PageRank algorithm, with normalized words' term frequencies as prior distribution, to determine words' importance, e.g., the importance score of word w is denoted as $imp(w)$, and importance score of sentence s is: $Imp(s) = \sum_{w\in s} imp(w)\cdot\sqrt{c_s(w)}$, where $c_s(w)$ is the times of w occurring in

[1] For more details, please refer to (Maas et al., 2011)

s. Note that we have filtered out stop words. We use $\sqrt{\{\cdot\}}$ to reduce the influence of repeated words instead of using sentence length to divide the aggregate score of a sentence, because we believe that a sentence with more important words deserves high importance. Based on our algorithm, a sentence with lots of unimportant words will not get a high importance score.

Given two sentences such as s_1 and s_2, our next step is to find for each word a in one sentence the corresponding word a^*, in the other sentence, that maximizes their mutual semantic relatedness (e.g., a in s_1, a^* in s_2 and vice versa).

$$a^* = \arg\max_{b \in s_2} SR(a, b) \tag{5}$$

Then, we average the semantic relevance scores from all terms in sentence s_1, with reference to their best matches in sentence s_2, as shown in Equation 6.

$$\zeta(s_1, s_2) = \frac{\sum_{w_i \in s_1} imp(w_i) \cdot SR(w_i, w_i^*)}{\sum_{w_i \in s_1} imp(w_i)} \tag{6}$$

We do the same for the opposite direction (i.e., from the words of s_2 to the words of s_1) to cover the cases where the two sentences are not equally important or they receive different similarities from each other. Finally, we derive the similarity between sentences s_1 and s_2 as:

$$sim(s_1, s_2) = \frac{Imp(s_1) \cdot \zeta(s_1, s_2) + Imp(s_2) \cdot \zeta(s_2, s_1)}{Imp(s_1) + Imp(s_2)} \tag{7}$$

4 Feature design

In the case of query-sensitive summarization, we conclude that qualified summary sentences should mainly meet the following typical demands: query-biased relevance (Shen and Li, 2011; Otterbacher et al., 2005), biased information richness (Wan et al., 2007) and biased novelty (Wan et al., 2007). Query-biased relevance requires that the sentences in the summary must overlap with the query in terms of topical content. Query-biased information richness denotes the information degree of a sentence with respect to both the sentence collection and the query. Query-biased information novelty is used to measure the content uniqueness of a sentence based on that sentence's capability in differentiating itself from other sentences as well as responding to the demands of the query. According to above definitions, we design multiple sentence features corresponding to them, respectively.

4.1 Four kinds of relevance

Given a sentence s, we exploit following information available in the DUC2005-2007 datasets:

- t_s: Title of the document containing sentence s.
- d_s: The document containing sentence s.
- c_s: The document cluster containing sentence s.
- q_s: Query of the document collection containing sentence s.

Noting that we do not conduct sentence segmentation if the query consists of more than one question, instead we treat it as a single, long sentence. Consequently, we calculate the following four sentence features using similarity measure discussed in Section 3:

- $r_{t,s}$: Relevance between sentence s and the title of the document to which s belongs.
- $r_{d,s}$: Relevance between sentence s and the document to which s belongs.
- $r_{c,s}$: Relevance between sentence s and the document cluster to which s belongs.
- $r_{q,s}$: Relevance between sentence s and query q.

4.2 Biased information richness (BIR)

Given a sentence collection and a query q, the BIR of sentence s is used to indicate the information degree of s with regard to both the sentence set and q, i.e., the richness of information contained in the sentence s biased towards q.

This feature score for each sentence is obtained via a variant version of the manifold-ranking process proposed in (Zhou et al., 2004). Points $\{s_0, s_1, \cdots, s_n\}$ denote the query statement (s_0) and all the sentences in the document collection ($\{s_i | 1 \leq i \leq n\}$) in a manifold space. The ranking function is denoted by $f = [f_0, f_1, ..., f_n]$. (Wan et al., 2007) hypothesized that all the sentences had blank prior knowledge so their initial scores were all set to zero. Whereas in this study, it is rational to treat the query-sentence relevance discussed in Section 4.1 as prior knowledge of sentences. Since s_0 denotes the query description, the initial score vector of these sentences is y $=[y_0, y_1, ..., y_n]$, where $y_0 = 1$ and $y_i = \text{sim}(s_i, s_0)$ (1≤ i≤ n). The manifold ranking can be performed iteratively using the following equation:

$$f(k+1) = \alpha S f(k) + (1-\alpha) y \tag{8}$$

where S is the symmetrically normalized similarity/relevance matrix as for $\{s_0, s_1, \cdots, s_n\}$, trade-off parameter α is set to 0.6, and k indicates the k^{th} iteration. Obviously, modified initial scores will exert a greater influence to sentence scores than the settings in (Wan et al., 2007) at each step of the iteration process. After convergence, let $f_i^\star$ denotes the limit of the sequence $\{f_i(t)\}$, then the BIR of sentence s_i is:

$$BIR(s_i) = f_i^\star \quad (1 \leq i \leq n) \tag{9}$$

4.3 Biased information novelty (BIN)

In our perspective, those sentences, owning relative high BINs and picked out to generate summary, must have information redundancy as low as possible meanwhile meet the user's information need, expressed by a query, as much as possible. Satisfaction of only one of them will certainly be off the original intention of biased novelty. Hence, we employ DivRank, proposed in (Mei et al., 2010), to acquire this sentence property. DivRank uses a *vertex-reinforced random walk* model to rank graph nodes based on a diversity based centrality. The basic assumption in DivRank is that the transition probability from a node to another is reinforced by the number of previous visits to the target node. Let $p_T(u, v)$ be the transition probability from any state u to any state v at time T. We can define a family of time-variant random walk processes in which $p_T(u, v)$ satisfies

$$p_T(u, v) = (1-\lambda) \cdot p^*(v) + \lambda \cdot \frac{p_o(u, v) \cdot N_T(v)}{D_T(u)} \tag{10}$$

where $D_T(u) = \sum_{v \in V} p_o(u, v) N_T(v)$. Here, $N_T(v)$ is the number of times that node v has been visited up to time T and $p^*(v)$ is a distribution which represents the prior preference of

visiting vertex v. In our task, $p^*(v)$ is set to the normalized similarity between sentence v and the query q, i.e., $p^*(v) = \widetilde{sim}(v,q)$. $p_o(u,v)$ is the *organic* transition probability prior to any reinforcement, which can be estimated in a regular time-homogenous random walk, such as

$$p_o(u,v) = \begin{cases} \gamma \cdot \frac{sim(u,v)}{degree(u)} & \text{if } u \neq v; \\ 1-\gamma & \text{otherwise} \end{cases} \tag{11}$$

If the network is ergodic, after a sufficiently large T, the reinforced random walk defined by Equation 10 also converges to a stationary distribution $\pi(v)$. That is

$$\pi(v) = \sum_{u \in V} p_t(u,v)\pi(u), \qquad \forall t \geq T \tag{12}$$

$\pi(v)$ is then used to denote the BIN of sentence v. In experiments, $\lambda = 0.9$ and $\gamma = 0.25$.

5 RelationListwise ranking function construction

Inspired by the work in (Zhou et al., 2011), which described a general ranking function with relationship information among objects, we modify that model specifically for our summarization task. Firstly, we define some notations used in this section. Query q is associated with a sentence collection $S = \{s_1, s_2, \cdots, s_n\}$, and S is associated with a set of judgments $Y = \{y_1, y_2, \cdots, y_n\}$. Here, n denotes the number of sentences in that collection, and y_i is the relevance judgment of sentence s_i with respect to query q. We could also treat y_i to be the position of sentence s_i in ranking list. Exactly, each sentence s_i is represented as a feature vector $\mathbf{x}_i = \Phi(q, s_i)$, where the acquisition of those features is presented in Section 4. In whole, we can see query q corresponds to a set of sentences S, a set of features vectors $X = \{\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_n\}$, a set of judgements Y, and R, the affinity matrix among sentences in S.

Let $g(\mathbf{x}_i, \mathbf{w})$ denote the basic ranking function of Listwise method, i.e.,

$$g(\mathbf{x}_i, \mathbf{w}) = < \mathbf{x}_i, \mathbf{w} > = \mathbf{x}_i \cdot \mathbf{w} \tag{13}$$

where vector $\mathbf{w}$ is unknown, and is exactly what we want to learn. In this paper, $g(\mathbf{x}_i, \mathbf{w})$, meaning the content relevance of s_i with regard to query q, is defined as a linear function, namely taking the inner product between vector $\mathbf{x}_i$ and $\mathbf{w}$. Based on this step, we use following formula to derive the final RelationListwise ranking score of sentence s_i ($1 \leq i \leq n$), denoted as $f_{\mathbf{w}}(\mathbf{x}_i, R)$, by integrating its initial Listwise ranking score (i.e., $g(\mathbf{x}_i, \mathbf{w})$) with its neighbors' Listwise scores:

$$f_{\mathbf{w}}(\mathbf{x}_i, R) = (1-\tau) g(\mathbf{x}_i, \mathbf{w}) + \tau \sum_{j \neq i}^{n} g(\mathbf{x}_j, \mathbf{w}) \cdot \tilde{R}^{(i,j)}) \tag{14}$$

where $\tilde{R}^{(i,j)} = \frac{R^{(i,j)}}{\sum_{s \neq i} R^{(i,s)}}$ denotes the normalized similarity between sentence s_i and s_j. The second item of Equation 14 can be interpreted as following: if the relevance score of s_j with query q is high and s_j is very related with s_i , then the relevance value between s_i and q will be increased significantly, and vice versa. In Equation 14 we can find that the prestige of sentence s_i is decided not only by the content of itself, but its neighbors' prestige. The coefficient τ is the weight of relation information (the second item of Equation 14). We can change its value to adjust the contribution of similarity information to the whole ranking value. In our experiment, we set it to 0.5.

5.1 RelationListMLE Top-K probability optimization

Before presenting our training algorithm, it's worth mentioning that a crucial difference between summarization scenario and most ranking problems is that summarization task casts more emphasis on the top-ranked sentences in the final list. Because of the length limit of a summary, most sentences assigned relatively low ranking scores will not be selected to construct a summary. Therefore, the correct ranking of the entire sentence permutation is not needed, our goal is to improve the ranking accuracies of top-K sentences (here we set K to a constant on condition that we are confident that K selected sentences satisfy the demand of summarization task about summary length, e.g., $K = 20$ in our experiments).

There are many training algorithms to learn listwise ranking function, such as Likelihood loss, Cosine loss and Cross entropy loss, among which likelihood loss has been proved to have the most comprehensive properties and its corresponding learning algorithm is called ListMLE (Xia et al., 2008). Accordingly, we integrate K value with listMLE to learn our proposed RelationListwise ranking function. This kind of solution was indicated by (Xia et al., 2009) to be more suitable for some real ranking applications where top-K objects are the focus. For convenience, we name it *RelationListMLE Top-K* probability optimization.

Our proposed optimization method also uses stochastic gradient descent algorithm to search the local minimum of loss functions. The stochastic gradient descent algorithm is described as Algorithm 1.

Algorithm 1: Stochastic Gradient Descent

Input: training data $\{\{X_1, Y_1, R_1\}, \{X_2, Y_2, R_2\}, \cdots, \{X_n, Y_n, R_n\}\}$
Parameter: learning rate η, tolerance rate ε
Initialize parameter **w**
repeat
  for $i = 1$ **to** n **do**
    (1)Compute score of each sentence j with current **w** using Equation 14
    (2)Compute the gradient $\Delta\mathbf{w}$ with current **w** using Equation 15
    (3)Update **w**=**w**-$\eta \times \Delta\mathbf{w}$
  end for
  Compute likelihood loss:

$$L = -\sum_{i=1}^{n} \log \prod_{c=1}^{K} \frac{\exp(f_{\mathbf{w}}(\mathbf{x}_{iy_i^c}, R_i))}{\sum_{t=c}^{n_i} \exp(f_{\mathbf{w}}(\mathbf{x}_{iy_i^t}, R_i))}$$

until change of likelihood loss is below ε times the previous loss
Output: w

In RelationListMLE-Top-K, the gradient of likelihood loss $L(f_{\mathbf{w}}(X_i, R_i), Y_i)$ with respect to w_j can be derived as Equation 15:

$$\Delta w_j = \frac{\partial L(f_{\mathbf{w}}(X_i, R_i), Y_i)}{\partial w_j} \tag{15}$$

$$= \sum_{c=1}^{K} \{ \frac{\sum_{t=c}^{n_i} [\exp(f_{\mathbf{w}}(\mathbf{x}_{iy_i^t}, R_i)) \cdot \frac{\partial f_{\mathbf{w}}(\mathbf{x}_{iy_i^t}, R_i)}{\partial w_j}]}{\sum_{t=c}^{n_i} \exp f_{\mathbf{w}}(\mathbf{x}_{iy_i^t}, R_i)} - \frac{\partial f_{\mathbf{w}}(\mathbf{x}_{iy_i^c}, R_i)}{\partial w_j} \}$$

where

$$\frac{\partial f_{\mathbf{w}}(\mathbf{x}_{ir}, R_i)}{\partial w_j} = (1-\tau)\mathbf{x}_{ir}^j + \tau \sum_{p=1, p\neq r}^{n_i} \mathbf{x}_{ip}^j \tilde{R}_i^{(r,p)} \tag{16}$$

and $\mathbf{x}_{ir}^j$ is the j^{th} element in $\mathbf{x}_{ir}$

6 Training data construction

To apply learning to rank in summarization, we should have a labeled training collection in the form of $(\mathbf{x}_1, y_1), \cdots, (\mathbf{x}_n, y_n)$, where $\mathbf{x}_i$ is a sentence (or sentence feature vector) in sentence set S and y_i is the ranking of the sentences. In addition, the relation information among sentences must be constructed too. To estimate the ranking score of a sentence s given the human summary H, we implement manifold-ranking algorithm to achieve our goal, treating H as a query vertex in graph. The reason for using manifold-ranking algorithm rather than directly calculating the relevance of a sentence towards the query (i.e., H) lies in that manifold-ranking process considers the query-sentence relatedness as well as the inter-sentence affinities. It matches our intention of improving the basic listwise with relationships among sentences.

Whereas, it is worth mentioning that we do not treat the human summary H wholly as a long sentence to participate in the computing of query-sentence similarities. Instead, we treat H as a sentence set and our goal is to find a sentence, from H, that has the relatively maximum similarity with an estimated sentence in S. This is because that if a sentence is similar with a summary sentence, it is also supposed to have the potential to become a summary sentence even though it might have no similarity with other sentences in H at all. So, during the manifold-ranking process, similarity between a pair of sentences is acquired via the method discussed in Section 3 while H-sentence relevance is obtained as follows:

$$rel(s, H) = \max_{r \in H} sim(s, r) \tag{17}$$

We name our method for training data generation $sent\_manifold$. Additionally, biased LexRank (Erkan, 2006) is also a feasible solution to rank sentences for training data construction. Its primary idea is to generate a prior distribution for the objects in traditional random walk to reflect the bias degree of objects towards certain query. We will conduct experiments to compare our approach with some representative alternatives.

7 Experimental study

7.1 Data sets and evaluation metrics

We use the popular query-focused summarization benchmark data sets DUC2005[2], DUC2006[3] and DUC2007[4] for our experiments. Each of them consists of document sets and reference/human summaries. For documents, we use the OpenNLP[5] to detect and tokenize sentences. Stop words are removed and remaining words are stemmed using Porter stemmer[6]. In experiments, DUC2005 is used to train the model tested on DUC2006, and DUC2006 is used to train the model tested on DUC2007. Table 1 gives a short summary of the three data sets.

[2]http://www-nlpir.nist.gov/projects/duc/duc2005/tasks.html
[3]http://www-nlpir.nist.gov/projects/duc/duc2006/tasks.html
[4]http://www-nlpir.nist.gov/projects/duc/duc2007/tasks.html
[5]http://opennlp.sourceforge.net/
[6]http://tartarus.org/ martin/PorterStemmer/

	DUC2005	DUC2006	DUC2007
Cluster number	50	50	45
Documents per cluster	25-50	25	25
Summary length limit	250 words	250 words	250words

Table 1: Summary of datasets

We use ROUGE (Lin, 2004) (version 1.5.5) toolkit[7] to measure the summarization performance. In experiments, we report three widely adopted F-measure metrics: ROUGE-1, ROUGE-2 and ROUGE-SU4, among which ROUGE-N means n-gram recall, and ROUGE-SU4 is based on unigram plus skip-bigram match with maximum skip distance of 4.

7.2 Experimental results

7.2.1 Comparison among some typical supervised summarization systems

First, we compare RelationListwise with some competitive and typical supervised summarization algorithms and three top systems of DUC. (1)Ranking-SVM: applying ranking-SVM directly; (2) Ranking-SVM-CSL: Ranking-SVM with Cost Sensitive Loss, proposed in (Shen and Li, 2011) (3)SVR: learning a regression model using SVM, presented in (Ouyang et al., 2007); (4)Listwise: similar to our RelationListwise while taking no account of sentence relationship; (5)top three systems with the highest ROUGE scores that participated in the DUC2006 (S12, S23, S24) and the DUC2007 (S4, S15, S29) for comparison, respectively.

Systems	ROUGE-1	ROUGE-2	ROUGE-SU4
SVR	0.41813	0.09492	0.15116
Ranking-SVM	0.42014	0.09713	0.15326
Ranking-SVM-CSL	0.42179	0.10332	0.15377
S23	0.40973	0.09785	0.14562
S12	0.41053	0.09633	0.15074
S24	0.41081	0.09857	0.15248
Listwise	0.42716	0.10387	0.16008
RelationListwise	**0.43066**	**0.10852**	**0.16324**

Table 2: F-measure comparison on DUC2006

Systems	ROUGE-1	ROUGE-2	ROUGE-SU4
SVR	0.43821	0.11997	0.16508
Ranking-SVM	0.44514	0.12213	0.17326
Ranking-SVM-CSL	0.44839	0.12332	0.17377
S4	0.43603	0.11785	0.17162
S29	0.43159	0.12048	0.17374
S15	0.44481	0.12907	0.17748
Listwise	0.45283	0.12667	0.17549
RelationListwise	**0.45852**	**0.13091**	**0.17824**

Table 3: F-measure comparison on DUC2007

Tables 2 and 3 present the performance of these systems with the metrics ROUGE-1, ROUGE-2, and ROUGE-SU4. From the results we can observe that in this task, listwise based methods

[7] http://www.isi.edu/licensed-sw/see/rouge/

(RelationListwise and Listwise) generally outperform pairwise methods (Ranking-SVM and Ranking-SVM-CSL), and the latter outperforms SVR, a pointwise learning to ranking. Even through (Shen and Li, 2011) developed cost sensitive loss to improve basic ranking SVM, its results are still inferior to that of Listwise based methods, which indicates the correctness of our choosing Listwise learning to rank as basic training model. More importantly, taking into account the sentence relatedness indeed improves the performance of $Listwise$. As the statistics show, $RelationListwise$ outperforms $Listwise$ over all three metrics.

7.2.2 Validation of feature design

Further, in order to investigate the effectiveness of combining our designed features, we compare our method RelationListwise with some baselines which mainly consider individual features: (1)Rel: a method considering only the sentence relevance towards a query and choosing the most relevant sentences to produce summary until length limit is reached. (2)Rel+MMR: similar with (1) except that we use MMR algorithm to control redundancy. It denotes a system considering query-biased relevance as well as information novelty. (3)Coverage: a baseline clustering-based method. It clusters sentences and selects the most relevant sentences from different clusters. Note that the clustering operation is carried to select sentences that have low degree of information overlap. So this baseline is similar with (2) for considering both relevance and novelty. (4)Manifold: ranking the sentences according to the manifold ranking scores and select top-ranked sentences to construct summary directly, where the parameter $\alpha = 0.5$. It corresponds to feature BIR. (5)Manifold+MMR: similar with (4) except to reduce redundancy via MMR algorithm (Wan et al., 2007). (4)Diversity: selecting sentences according to their query-biased diversity scores acquired using DivRank (Mei et al., 2010). It represents the feature: BIN. Tables 4 and 5 show their comparison results.

Systems	ROUGE-1	ROUGE-2	ROUGE-SU4
Rel	0.36775	0.07092	0.12777
Rel+MMR	0.37328	0.07109	0.12884
Manifold	0.38827	0.08028	0.13349
Coverage	0.39004	0.08394	0.13705
Diversity	0.39052	0.08814	0.13721
Manifold+MMR	0.39116	0.08741	0.13729
RelationListwise	**0.43066**	**0.10852**	**0.16324**

Table 4: Comparison results of feature design on DUC2006

Systems	ROUGE-1	ROUGE-2	ROUGE-SU4
Rel	0.38985	0.10075	0.13108
Rel+MMR	0.39938	0.1033	0.14501
Manifold	0.40214	0.10131	0.14833
Coverage	0.41243	0.11196	0.15537
Diversity	0.41440	0.11261	0.15502
Manifold+MMR	0.42015	0.11327	0.15936
RelationListwise	**0.45852**	**0.13091**	**0.17824**

Table 5: Comparison results of feature design on DUC2007

Obviously, the statistics point out the improvement of our approach combining multiple task-specific features over those baselines. While some reference systems involve more than one

information aspect, such as *Rel+MMR* and *Manifold+MMR*, their performances are still limited.

7.2.3 Competitiveness of our similarity measure

In Section 7.2.1, experimental results have validated our proposal that exploiting sentence relation to improve the overall training model. Nevertheless, they could not point out the superiority of our designed measure of sentence similarity. Hence, we keep consistency for our algorithm framework except to replace the part of calculating sentence similarity. Since there are lots of existing sentence similarity measures, and it's hard to compare qualities of them all, we just select following typical alternatives: (1)cosine measure; (2)wordSimi_sentStruct: The measure proposed in (Yin et al., 2012), which determined words semantic similarity by computing the cosine value of the words' distribution representations over latent topics, and identified sentence structure similarity using LCS and etc.; (3)geneProb: Generation probability method presented in (Erkan, 2006). Note that generation probability is not necessarily symmetric, we just average the mutual generation probabilities of two sentences as their final similarity value in experiments. We provide their comparison statistics in Tables 6-7.

Systems	ROUGE-1	ROUGE-2	ROUGE-SU4
Listwise	0.42716	0.10387	0.16008
cosine measure	0.42906	0.10659	0.16239
geneProb	0.42923	0.10671	0.16246
wordSimi_sentStruct	0.43004	0.10726	0.16307
RelationListwise	**0.43066**	**0.10852**	**0.16324**

Table 6: Comparison results of sentence similarity measures on DUC2006

Systems	ROUGE-1	ROUGE-2	ROUGE-SU4
Listwise	0.45283	0.12667	0.17549
cosine measure	0.45626	0.12819	0.17634
geneProb	0.45693	0.12873	0.17616
wordSimi_sentStruct	0.45793	0.12956	**0.17833**
RelationListwise	**0.45852**	**0.13091**	0.17824

Table 7: Comparison results of sentence similarity measures on DUC2007

The two tables demonstrate the influence of different sentence similarity measures on our approach. Among the four kinds of measures, $geneProb$ is close with $cosine$ with slight improvement while $wordSimi\_sentStruct$ is more competitive to our proposed similarity measure. Except that $wordSimi\_sentStruct$ performs slightly better than $RelationListwise$ in ROUGE-SU4 over DUC2007, our proposed measure is more superior in other metrics. Note that we also put method $Listwise$ in the tables, and yet its performance is relatively poor compared with ones involving inter-sentence impacts. It further validates that sentence relatedness is worth considering when dealing with sentence ranking problem.

7.2.4 Training data generation comparison

In this section, we empirically investigate the effects of different strategies for training data generation. In Section 6, we have given the reason why choose to compute the similarity of an estimated sentence towards one sentence in H instead of the whole H. Additionally, we also come up with a novel approach for measuring sentence similarity in Section 3. In general, we

could treat reference summary H as (1): a long and single sentence (labeled as *summ*), or (2): a sentence set (labeled as *sent*); meanwhile, ranking methods could be classified into (1): manifold, (2): biased LexRank, and (3)simi: computing the similarities of estimated sentences towards the summary directly. So, we can combine them into 6 kinds of pairs: $summ\_manifold$, $summ\_biasedLexRank$, $summ\_simi$, $sent\_manifold$, $sent\_biasedLexRank$ and $sent\_simi$. Remember that $sent\_manifold$ is our proposed method for training data construction.

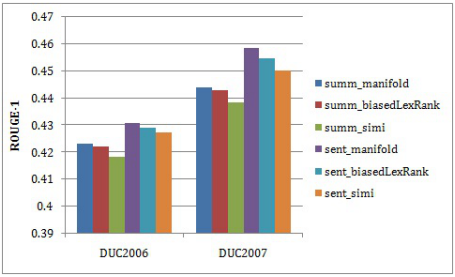


Figure 1: Performance comparison of methods about training data generation.

The comparison results are shown in Figure 1. From the comparison, we observe that:(1)graph-based methods, including $\{\cdot\}\_manifold$ and $\{\cdot\}\_biasedLexRank$, both outperform $\{\cdot\}\_simi$ methods which rank sentences by computing sentences' similarities towards the reference summary directly. It might result from that $\{\cdot\}\_simi$ solution is too simple to take into consideration the prestige of sentences in documents. (2) Using a sentence as the reference is much better than using the whole summary. As the figure shows, all $sent\_\{\cdot\}$ methods outperform $summ\_\{\cdot\}$. This may due to the fact that in constructing training data, we aim to judge the ability of a sentence to be a summary sentence, and it's best to be treated as the ability for the estimated sentence to *replace* certain sentence in H. Therefore, it is more rational to compare our estimated sentence with the sentences in H one by one rather than with the whole H. For example, if a sentence in H (e.g., h_i) has a high similarity (e.g., 0.9) with an estimated sentence s, then s is supposed to be able to replace h_i as a summary sentence. Whereas, if a long sentence s' is also very relevant to the whole H while having very low similarities with sentences in H, it is still not considered to be a good summary sentence.

Conclusion and perspectives

In this work, we propose a novel model named RelationListwise for query-biased multi-document summarization task. More specifically, through defining some unique sentence features and designing a creative measure for sentence relatedness, we integrate sentence relation information with listwise learning to rank to automatically learn feature weights. Experimental results suggest that our modification of basic listwise is considerably in favor of generating high-quality summaries. In future work, we will use more complex features and try some new summarization tasks.

Acknowledgments

This work is financially supported by HGJ Grant 2011ZX01042-001-001 and NSFC Grant 60933004.

References

Cao, Z., Qin, T., Liu, T., Tsai, M., and Li, H. (2007). Learning to rank: from pairwise approach to listwise approach. In *Proceedings of the 24th international conference on Machine learning*, pages 129–136. ACM.

CHALI, Y. and HASAN, S. (2011). Query-focused multi-document summarization: automatic data annotations and supervised learning approaches. *Natural Language Engineering*, 1(1):1–37.

Erkan, G. (2006). Using biased random walks for focused summarization. *Ann Arbor*, 1001:48109–2121.

Islam, A. and Inkpen, D. (2008). Semantic text similarity using corpus-based word similarity and string similarity. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 2(2):10.

Jin, F., Huang, M., and Zhu, X. (2010). A comparative study on ranking and selection strategies for multi-document summarization. In *Proceedings of the 23rd International Conference on Computational Linguistics: Posters*, pages 525–533. Association for Computational Linguistics.

Li, L., Zhou, K., Xue, G., Zha, H., and Yu, Y. (2009). Enhancing diversity, coverage and balance for summarization through structure learning. In *Proceedings of the 18th international conference on World wide web*, pages 71–80. ACM.

Li, X., Shen, Y., Du, L., and Xiong, C. (2010). Exploiting novelty, coverage and balance for topic-focused multi-document summarization. In *Proceedings of the 19th ACM international conference on Information and knowledge management*, pages 1765–1768. ACM.

Li, Y., McLean, D., Bandar, Z., O'Shea, J., and Crockett, K. (2006). Sentence similarity based on semantic nets and corpus statistics. *Knowledge and Data Engineering, IEEE Transactions on*, 18(8):1138–1150.

Lin, C. (2004). Rouge: A package for automatic evaluation of summaries. In *Proceedings of the workshop on text summarization branches out (WAS 2004)*, volume 16.

Maas, A., Daly, R., Pham, P., Huang, D., Ng, A., and Potts, C. (2011). Learning word vectors for sentiment analysis. In *The 49th Annual Meeting of the Association for Computational Linguistics (ACL 2011)*.

Maas, A. and Ng, A. (2010). A probabilistic model for semantic word vectors. In *NIPS 2010 Workshop on Deep Learning and Unsupervised Feature Learning*.

Mei, Q., Guo, J., and Radev, D. (2010). Divrank: the interplay of prestige and diversity in information networks. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1009–1018. ACM.

Otterbacher, J., Erkan, G., and Radev, D. (2005). Using random walks for question-focused sentence retrieval. In *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, pages 915–922. Association for Computational Linguistics.

Ouyang, Y., Li, S., and Li, W. (2007). Developing learning strategies for topic-based summarization. In *Proceedings of the sixteenth ACM conference on Conference on information and knowledge management*, pages 79–86. ACM.

Qin, T., Zhang, X., Tsai, M., Wang, D., Liu, T., and Li, H. (2008). Query-level loss functions for information retrieval. *Information Processing & Management*, 44(2):838–855.

Quan, X., Liu, G., Lu, Z., Ni, X., and Wenyin, L. (2010). Short text similarity based on probabilistic topics. *Knowledge and information systems*, 25(3):473–491.

Shen, C. and Li, T. (2011). Learning to rank for query-focused multi-document summarization. In *Data Mining (ICDM), 2011 IEEE 11th International Conference on*, pages 626–634. IEEE.

Wan, X. and Yang, J. (2008). Multi-document summarization using cluster-based link analysis. In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 299–306. ACM.

Wan, X., Yang, J., and Xiao, J. (2007). Manifold-ranking based topic-focused multi-document summarization. In *Proceedings of IJCAI*, volume 7, pages 2903–2908.

Wei, F., Li, W., Lu, Q., and He, Y. (2008). Query-sensitive mutual reinforcement chain and its application in query-oriented multi-document summarization. In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 283–290. ACM.

Xia, F., Liu, T., and Li, H. (2009). Top-k consistency of learning to rank methods. *Advances in Neural Information Processing Systems*, pages 2098–2106.

Xia, F., Liu, T., Wang, J., Zhang, W., and Li, H. (2008). Listwise approach to learning to rank: theory and algorithm. In *Proceedings of the 25th international conference on Machine learning*, pages 1192–1199. ACM.

Yin, W., Pei, Y., Zhang, F., and Huang, L. (2012). Automatic multi-document summarization based on new sentence similarity measures. In *Proceedings of the 2012 Pacific Rim International Conference on Artificial Intelligence*, pages 832–837.

Zhang, J., Sun, Y., Wang, H., and He, Y. (2011). Calculating statistical similarity between sentences. *Journal of Convergence Information Technology*, 6(2).

Zhao, L., Huang, X., and Wu, L. (2005). Fudan university at duc 2005. In *Proceedings of DUC*, volume 2005.

Zhou, D., Ding, Y., You, Q., and Xiao, M. (2011). Learning to rank documents using similarity information between objects. In *Neural Information Processing*, pages 374–381. Springer.

Zhou, D., Weston, J., Gretton, A., Bousquet, O., and Scholkopf, B. (2004). Ranking on data manifolds. *Advances in neural information processing systems*, 16:169–176.

SentTopic-MultiRank: a novel ranking model for multi-document summarization

Wenpeng Yin Yulong Pei Fan Zhang Lian′en Huang
The Shenzhen Key Lab for Cloud Computing Technology & Applications (SPCCTA)
Shenzhen Graduate School
Peking University, Shenzhen 518055, P.R. China
{mr.yinwenpeng,paul.yulong.pei,fan.zhgf}@gmail.com, hle@net.pku.edu.cn

ABSTRACT
Extractive multi-document summarization is mostly treated as a sentence ranking problem. Existing graph-based ranking methods for key-sentence extraction usually attempt to compute a global importance score for each sentence under a single relation. Motivated by the fact that both documents and sentences can be presented by a mixture of semantic topics detected by Latent Dirichlet Allocation (LDA), we propose SentTopic-MultiRank, a novel ranking model for multi-document summarization. It assumes various topics to be heterogeneous relations, then treats sentence connections in multiple topics as a heterogeneous network, where sentences and topics/relations are effectively linked together. Next, the iterative algorithm of MultiRank is carried out to determine the importance of sentences and topics simultaneously. Experimental results demonstrate the effectiveness of our model in promoting the performance of both generic and query-biased multi-document summarization tasks.

KEYWORDS: Multi-document summarization, Topic decomposition, Heterogeneous network.

Proceedings of COLING 2012: Technical Papers, pages 2977–2992,
COLING 2012, Mumbai, December 2012.

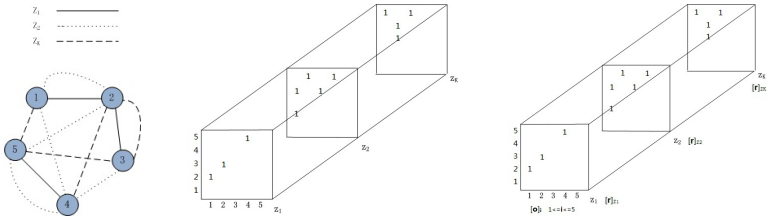
1 Introduction

Multi-document summarization (MDS), having been intensively studied in past decades, is the process of automatically creating a compressed version of a given document collection that provides useful information for a user. It could be mainly classified into two categories: generic MDS and query-focused MDS. The goal of generic MDS is to produce a summary of multiple documents about the same but unspecified topic, while query-focused task requires the generated summary could not only convey the main content of target corpus, but also bias to the information needs of a specific query/topic. Commonly, both of them are treated as a sentence ranking problem.

In most existing summarization systems, sentence ranking was conducted under a single relation assumption. Taking for instance the famous LexPageRank proposed in (Erkan and Radev, 2004). In that model, a sentence connectivity matrix was constructed based on cosine similarity. Then PageRank (Page et al., 1999) algorithm was directly applied to the cosine similarity graph to find the most prestigious sentences in a document. Another compelling example is (Wan et al., 2007), where a weighted network was formed on the sentences by using standard cosine similarity, then authors utilized manifold-ranking algorithm (Zhou et al., 2004) to spread ranking scores from a query to remaining sentences. Apparently, both above literatures were based on the assumption that all sentences existed under a unified relation.

Inspired by some work involving topic decomposition as well as multi-relational data where objects have interactions with others based on different relations, we attempt to map sentence relatedness within multiple topics to heterogeneous relations in this work. More specifically, we assume each topic as a single relation type, and construct an intra-topic sentence network for each relation type. There are many Information Retrieval and Data Mining tasks addressing multi-relational data. For instance, scholars cite other scholars in various conferences, and based on different academic fields, publications cite other publications on the basis of content analysis such as authorship, title, abstract and keywords, web pages link to each other via different anchor texts (Kolda and Bader, 2006). Such a complicated link structure can provide a way of incorporating multiple relations among objects into the derivation of object prestige or popularity. In Figure 1(a), we show an example of multi-relational sentence representation based on our SentTopic-MultiRank model. There are five sentences and K relations among them (K denotes the topic number in LDA, and we only provide three topics z_1, z_2 and z_K as an illustration). We can also represent such multi-relational objects in a tensor shape which is a multi-dimensional array. In the Figure 1(b), a three-way array is provided, where each two-dimensional plane represents an adjacency matrix for one type of relation. The network can be depicted as a tensor of size $5 \times 5 \times K$ where (i, j, k) entry is nonzero if the i^{th} sentence is related to the j^{th} sentence under k^{th} relation.

Then, how to determine the importance of those sentences and relations/topics? (Ng et al., 2011) proposed a general framework named MultiRank to deal with multi-relational data or the corresponding tensor representation for co-ranking purpose. According to that proposal, the MultiRank value of a sentence relies on the number and MultiRank values of all sentences that have multiple relations to this sentence, as well as the MultiRank values of those mutual relations. A sentence, connected via high MultiRank relations by sentences with high MultiRanks, receives a high MultiRank itself. Similarly, the MultiRank of a relation is dependent on which sentences to be linked and their MultiRank scores. A relation, connecting sentences with high MultiRanks, receives a high MultiRank itself. Similar to PageRank, MultiRank's idea



(a) An example of multi-relational sentences in a graph representation

(b) The corresponding tensor representation

(c) The tensor representation with prior

Figure 1: Illustration of SentTopic-MultiRank

is to imagine infinite random surfers in a multi-relational network, and derive a stationary probability distribution of objects and relations as evaluation scores for objects and relations, respectively. We integrate topic decomposition via LDA with MultiRank to produce a novel framework named SentTopic-MultiRank specifically for multi-document summarization.

For generic multi-document summarization, we directly make use of MultiRank algorithm to determine the ranking scores of sentences and topics/relations. While in query-oriented MDS, query bias must be considered. We creatively integrate LDA with generation probability between sentences to design query-oriented prior distributions for sentences and topics, respectively. Motivated by topic-sensitive PageRank (Haveliwala, 2003), as well as HAR (Xutao et al., 2012) which modified MultiRank based on HITS (Kleinberg, 1999) principle for query search, we embed acquired prior distributions into the SentTopic-MultiRank iterative process so that the final ranking results are more desirable for the query statement. Figure 1(c) gives an example for tensor representation of SentTopic-MultiRank with prior knowledge. We apply our approach to generic and query-biased multi-document summarization on standard datasets, experimental results show its good performance in generating high-quality summaries.

The rest of this paper is organized as follows: Section 2 introduces related work. Details of constructing and applying SentTopic-MultiRank model in multi-document summarization task are presented in Section 3. Section 4 gives the process of selecting sentences to generate summary. Finally, experiments and results are showed in Section 5.

2 Related work

In this section, we firstly introduce some representative work about multi-document summarization. Then, some literatures, relevant to topic decomposition, are also presented.

The centroid-based method (Radev et al., 2004) is one of the most popular extractive summarization methods. MEAD is an implementation of the centroid-based method that scores sentences based on features such as cluster centroids, position and TF-IDF. NeATS (Lin and Hovy, 2002) used sentence position, term frequency, topic signature and term clustering to select important content, and used MMR (Goldstein et al., 1999) to remove redundancy. Recently, graph-based methods have been proposed to rank sentences. LexPageRank (Erkan and Radev, 2004) and (Mihalcea and Tarau, 2005) are two such systems using algorithms similar to PageRank and HITS (Kleinberg, 1999) to compute sentence importance. With respect to query-

focused summarization, in (Saggion et al., 2003), a simple query-based scorer by computing the similarity value between each sentence and the query was incorporated into a generic summarizer to produce the query-based summary. (Erkan, 2006) came up with biased LexRank by incorporating prior knowledge into random walk for query summarization task. (Wan et al., 2007) exploited manifold-ranking process proposed in (Zhou et al., 2004) to implement the propagation of ranking scores from the query to remaining sentences, based on a weighted sentence connection network. The spread process was repeated until a global stable state was achieved, and all sentences obtained their final ranking scores. Recently, unsupervised deep learning was first investigated in (Liu et al., 2012) to deal with query summarization.

Furthermore, supervised learning approaches have also been successfully applied in document summarization, where the training data is available or easy to build. The most straightforward way is to regard the sentence extraction task as a binary classification problem. (Kupiec et al., 1995) developed a trainable summarization system which adopted various features and used a Bayesian classifier to learn the feature weights. The system performed better than other systems using only a single feature. (Zhou and Hovy, 2003) applied a HMM-based model and (Shen et al., 2007) proposed a conditional random field based framework. (Ouyang et al., 2007) designed methods for constructing training data based on human summaries and training sentence scoring models based on support vector regression (SVR). (Shen and Li, 2011) introduced a cost sensitive loss to improve ranking SVM, a type of learning-to-rank method, for extractive query-focused multi-document summarization.

Especially, we are mainly inspired by following pioneering work. In recent years, two algorithms were proposed to rank web pages by incorporating topic information within PageRank (Haveliwala, 2003; Nie et al., 2006). The method in (Haveliwala, 2003) decomposed PageRank into various topics, setting the preference values using some conditional probabilities. (Nie et al., 2006) proposed a more complicated ranking framework, where topical PageRanks were performed together. The rationale of (Nie et al., 2006) was, when surfing following a graph link from vertex w_i to w_j, the ranking score on topic z of w_i would have a higher probability to pass to the same topic of w_j and have a lower probability to pass to a different topic of w_j. When the inter-topic jump probability was 0, this method was identical to the approach in (Haveliwala, 2003). (Liu et al., 2010) explicitly defined a new graph-based framework, Topical PageRank, for the task of keyphrase extraction. It first ranked all key-phrases within each latent topic based on a topic-specific phrase graph, then re-ranked them by integrating corpus-topic distribution with intra-topic phrase ranking. Additionally, ToPageRank, a model similar with that of (Liu et al., 2010), was proposed in (Pei et al., 2012) specifically for summarization task. Indeed, a similarity between these literatures with our current idea lies in that we all consider to decompose entire corpus into multiple topics. Whereas, we do not aim to *break up the whole into parts* simply. In their work, there was usually no relation exerted to link each part, so the operation in each part was conducted independently and those objects would not interact with various kinds of relations. Hence, in essence, they still coped with ranking problem under a single relation presupposition, except that having narrowed the scope from a corpus to an individual topic. Differently, in SentTopic-MultiRank, not only are sentences linked with each other by weighted edges, all intra-topic sentence networks are also connected together, though they are considered to be under different relation types, so that we are able to identify the importance of sentences and topics/relations simultaneously. Further, we make full use of sentence-sentence relatedness, sentence-topic interaction and inter-topic impacts to improve the overall ranking performance.

3 SentTopic-MultiRank model

3.1 Topic decomposition of corpus via Latent Dirichlet Allocation (LDA)

In our work, topic model LDA (Blei et al., 2003) is utilized to represent document collection with a mixture of semantic topics. In LDA, it is assumed that observed words in each document are generated by a document-specific mixture of corpus-wide latent topics. We define our corpus of length W with the flat word vector $\mathbf{w} = w_1, \cdots, w_W$. At corpus position i, the element d_i in $\mathbf{d} = d_1, \cdots, d_W$ designates the document containing observed word w_i. Similarly, the vector $\mathbf{z} = z_1, \cdots, z_W$ defines the hidden topic assignments of each observed word. The number of latent topics is fixed to some K, and each topic $z = 1, \cdots, K$ is associated with a topic-word multinomial ϕ_z over the W-word vocabulary. Each ϕ multinomial is generated by a conjugate Dirichlet prior with parameter β. Each document $j = 1, \cdots, D$ is associated with a multinomial θ_j over K topics, which is also generated by a conjugate Dirichlet prior with parameter α. The full generative model is then given by

$$P(\mathbf{w}, \mathbf{z}, \phi, \theta | \alpha, \beta, \mathbf{d}) \propto \left(\prod_{z=1}^{K} p(\phi_z|\beta)\right)\left(\prod_{j=1}^{D} p(\theta_j|\alpha)\right)\left(\prod_{i=1}^{W} \phi_{z_i}(w_i)\theta_{d_i}(z_i)\right)$$

where $\phi_{z_i}(w_i)$ is the w_i-th element in vector ϕ_{z_i}, and $\theta_{d_i}(z_i)$ is the z_i-th element in vector θ_{d_i}. Given an observed corpus $(\mathbf{w}, \mathbf{d})$ and model hyperparameters (α, β), the typical modeling goal is to infer the latent variables $(\mathbf{z}, \phi, \theta)$.

While exact LDA inference is intractable, a variety of approximate schemes have been developed. In this work, we use GibbsLDA++[1], a C/C++ implementation of LDA using Gibbs Sampling, to detect latent topics. This sampling approach iteratively re-samples a new value for each latent topic assignment z_i, conditioned on the current values of all other $\mathbf{z}$ values. After a fixed number of iterations, we estimate the topic-word multinomials ϕ and the document-topic mixture weights θ from the final $\mathbf{z}$ sample, using the means of their posteriors given by

$$\phi_z(w) \propto n_{zw} + \beta$$
$$\theta_j(z) \propto n_{jz} + \alpha$$

where n_{zw} is the number of times word w is assigned to topic z, and n_{jz} is the number of times topic z is used in document j, with both counts being taken with respect to the final sample $\mathbf{z}$. The topic-word multinomials ϕ_z for each topic z are our learned topics; each document-topic multinomial θ_j represents the prevalence of topics within document j.

In experiments, we set $\alpha = 1$ and $\beta = 0.01$. There are already some literatures to study their impacts on LDA performance. Since we pay main attentions to SentTopic-MultiRank model, the influences of LDA hyperparameters α and β are not investigated any more.

3.2 Construction of SentTopic-MultiRank graph

In Section 3.1, we have detected various relation types/topics. The thing left to do in constructing SentTopic-MultiRank is to produce a sentence graph under each relation. Naturally, the sentence similarity should be relation-specific. To capture the similarities of two sentences

[1] GibbsLDA++: http://gibbslda.sourceforge.net

(e.g., x and y) on various latent topics, we represent each sentence as a probability distribution at each topic z ($1 \le z \le K$). Hence, we sample sparse unigram distributions from each ϕ_z using the words in x and y. Their probability distributions given topic-word distribution are denoted as $P_z^x = p(\mathbf{w}_x|z, \phi_z)$ with the word set $\mathbf{w}_x = (w_1, \cdots, w_{|x|})$ in x, and $P_z^y = p(\mathbf{w}_y|z, \phi_z)$ with the word set $\mathbf{w}_y = (w_1, \cdots, w_{|y|})$ in y.

The probability distributions per topic are constructed with only the words in x and y, and the probabilities of remaining words in vocabulary W are set to 0. The W dimensional word probabilities are the expected posteriors obtained from LDA model. Hence, $p_z^x = (\phi_z(w_1), \cdots, \phi_z(w_{|x|}), 0, 0, \cdots) \in (0,1)^W$, $p_z^y = (\phi_z(w_1), \cdots, \phi_z(w_{|y|}), 0, 0, \cdots) \in (0,1)^W$. Given a topic z, the similarity between p_z^x and p_z^y is measured via transformed radius (TR). We first measure the divergence at each topic using TR based on Kullback-Liebler (KL) divergence:

$$TR(p_z^x, p_z^y) = KL(p_z^x || \frac{p_z^x + p_z^y}{2}) + KL(p_z^y || \frac{p_z^x + p_z^y}{2}) \quad (1)$$

where $KL(m||n) = \sum_i m_i \log \frac{m_i}{n_i}$. Then TR is transformed into similarity (Manning et al., 1999):

$$Sim(p_z^x, p_z^y) = 10^{-TR(p_z^x, p_z^y)} \quad (2)$$

Here, we adopt TR rather than the commonly used KL for the reason that with TR there is no problem with infinite values since $\frac{p_z^x + p_z^y}{2} \neq 0$ if either $p_z^x \neq 0$ or $p_z^y \neq 0$, and it is also symmetric, i.e., $TR(x, y) = TR(y, x)$. With the gotten sentence similarities under various topics, we could construct graphs like Figures 1(a) and 1(b) to demonstrate our SentTopic-MultiRank model.

3.3 Sentence ranking using MultiRank algorithm

There are many Data Mining and Machine Learning issues in multi-relational data where objects interact with others under different relations. The work in (Ng et al., 2011) proposed a framework, MultiRank, to determine the importance of both objects and relations simultaneously based on a probability distribution computed from multi-relational data. In our multi-relational sentence network, we apply MultiRank algorithm to derive the importance of sentences and topics. As we analyze sentences under multiple relations and also consider interaction between relations based on sentences, we make use of rectangular tensors to represent them as Figure 1(b). First, we introduce the MultiRank iterative algorithm.

Let R be the real field. We call $\mathscr{A} = (a_{s_1,s_2,z_1})$ where $a_{s_1,s_2,z_1} \in R$, for $s_i = 1, \cdots, n$, $i = 1, 2$ and $z_1 = 1, \cdots, K$, a real (2,1)th order $(n \times K)$-dimensional rectangular tensor. In this setting, we refer (s_1, s_2) to the indices for sentences/objects and z_1 to be the index for topic/relation. For instance, five sentences ($n = 5$) and three relations ($K = 3$) are demonstrated in Figure 1.

First, two transition probability tensors $\mathscr{O} = (o_{s_1,s_2,z_1})$ and $\mathscr{R} = (r_{s_1,s_2,z_1})$ are constructed with respect to sentences and topics by normalizing the entries of $\mathscr{A}$ as follows:

$$o_{s_1,s_2,z_1} = \frac{a_{s_1,s_2,z_1}}{\sum_{i=1}^{m} a_{i,s_2,z_1}}, \quad s_1 = 1, 2, \cdots, n$$

$$r_{s_1,s_2,z_1} = \frac{a_{s_1,s_2,z_1}}{\sum_{j=1}^{K} a_{s_1,s_2,j}}, \quad z_1 = 1, 2, \cdots, K$$

Then we derive the following probabilities:

$$\text{Prob}[X_t = s_1] = \sum_{s_2=1}^{n} \sum_{z_1=1}^{K} o_{s_1,s_2,z_1} \times \text{Prob}[X_{t-1} = s_2, Y_t = z_1] \tag{3}$$

$$\text{Prob}[Y_t = z_1] = \sum_{s_1=1}^{n} \sum_{s_2=1}^{n} r_{s_1,s_2,z_1} \times \text{Prob}[X_t = s_1, X_{t-1} = s_2] \tag{4}$$

where $\text{Prob}[X_{t-1} = s_2, Y_t = z_1]$ is the joint probability distribution of X_{t-1} and Y_t, and $\text{Prob}[X_t = s_1, X_{t-1} = s_2]$ is the joint probability distribution of X_t and X_{t-1}. Suppose our desired equilibrium/stationary distributions of sentences and relations, i.e., the SentTopic-MultiRank values of sentences and relations, are given by

$$\bar{\mathbf{x}} = [\bar{x}_1, \bar{x}_2, \cdots, \bar{x}_n]^T \text{ and } \bar{\mathbf{y}} = [\bar{y}_1, \bar{y}_2, \cdots, \bar{y}_K]^T$$

respectively, with

$$\bar{x}_{s_1} = \lim_{t\to\infty} \text{Prod}[X_t = s_1] \text{ and } \bar{y}_{z_1} = \lim_{t\to\infty} \text{Prod}[Y_t = z_1]$$

for $1 \le s_1 \le n$ and $1 \le z_1 \le K$.

As (3) and (4) are coupled together and they involve two joint probability distributions, a product form of individual probability distributions is employed to replace the joint probability distributions in (3) and (4) by assumption:

$$\text{Prob}[X_{t-1} = s_2, Y_t = z_1] = \text{Prob}[X_{t-1} = s_2]\text{Prod}[Y_t = z_1]$$
$$\text{Prob}[X_t = s_1, X_{t-1} = s_2] = \text{Prob}[X_t = s_1]\text{Prod}[X_{t-1} = s_2]$$

Hence, using the above assumptions and considering t goes to infinity, (3) and (4) becomes

$$\bar{x}_{s_1} = \sum_{s_2=1}^{n} \sum_{z_1=1}^{K} o_{s_1,s_2,z_1} \bar{x}_{s_2} \bar{y}_{z_1}, \quad s_1 = 1, 2, \cdots n \tag{5}$$

$$\bar{y}_{z_1} = \sum_{s_1=1}^{n} \sum_{s_2=1}^{n} r_{s_1,s_2,z_1} \bar{x}_{s_1} \bar{x}_{s_2}, \quad z_1 = 1, 2, \cdots K \tag{6}$$

Under the tensor operation for (5) and (6), we solve the following tensor (multivariate polynomial) equations:

$$\mathscr{O}\bar{\mathbf{x}}\bar{\mathbf{y}} = \bar{\mathbf{x}} \text{ and } \mathscr{R}\bar{\mathbf{x}}^2 = \bar{\mathbf{y}} \tag{7}$$

with

$$\sum_{s_1=1}^{n} \bar{x}_{s_1} = 1 \text{ and } \sum_{z_1=1}^{K} \bar{y}_{z_1} = 1 \tag{8}$$

to obtain the SentTopic-MultiRank values of sentences and relations. An efficient iterative algorithm to solve the tensor equations in (7) is summarized as Algorithm 1. With respect to theoretical analysis of this algorithm, please refer to (Ng et al., 2011) for details.

Algorithm 1: The MultiRank Algorithm

Input: Two tensors $\mathscr{O}$ and $\mathscr{R}$, two initial probability distributions $\mathbf{x}_0$ and $\mathbf{y}_0$ ($\sum_{s_1=1}^{n}[\mathbf{x}_0]_{s_1}=1$ and $\sum_{z_1=1}^{K}[\mathbf{y}_0]_{z_1}=1$) and the tolerance ϵ
Output: Two stationary probability distributions $\bar{\mathbf{x}}$ and $\bar{\mathbf{y}}$
Procedure:

1. Set $t=1$;
2. Compute $\mathbf{x}_t=\mathscr{O}\mathbf{x}_{t-1}\mathbf{y}_{t-1}$;
3. Compute $\mathbf{y}_t=\mathscr{R}\mathbf{x}_t^2$;
4. If $\| \mathbf{x}_t-\mathbf{x}_{t-1} \|+\| \mathbf{y}_t-\mathbf{y}_{t-1} \|<\epsilon$, then stop, otherwise set $t=t+1$ and goto Step 2.

In Algorithm 1, the setting of initial probability distributions does not influence the finally generated stationary probability distributions. We simply set uniform distribution for those sentences while with regard to topics, we set their initial distribution using the topic distribution of the entire corpus. Obviously, above algorithm has nothing to do with any query, so it does not apply to biased summarization task, which is further discussed in subsequent section.

3.4 Prior configuration in query-focused multi-document summarization

Query summarization requires that generated summary could not only express the salient content of target document collection, but also bias to the information needs of query. Then, how to exert the query information to the MultiRank process? Motivated by the idea of topic-sensitive PageRank (Haveliwala, 2003) and random walk with restart (Tong et al., 2008), we consider this issue by assigning desired limiting probability distributions towards sentences and topics, respectively. More specifically, we modify the tensor equations in (7) as follows:

$$(1-\mu)\mathscr{O}\bar{\mathbf{x}}\bar{\mathbf{y}}+\mu\mathbf{o}=\bar{\mathbf{x}} \tag{9}$$

$$(1-\nu)\mathscr{R}\bar{\mathbf{x}}^2+\nu\mathbf{r}=\bar{\mathbf{y}} \tag{10}$$

with Equation 8, where $\mathbf{o}$ and $\mathbf{r}$ are two devised prior probability distributions for sentences and topics/relations, respectively. Here, ($\sum_{i=1}^{n}[\mathbf{o}]_i=1$ and $\sum_{j=1}^{K}[\mathbf{r}]_j=1$), and $0\le\mu,\nu<1$, are two parameters for controlling the influence degree of two prior probability distributions. In experiments, we set $\mu=0.5$ and $\nu=0.9$ simply as (Xutao et al., 2012) indicated. Accordingly, the new iterative algorithm could be presented as Algorithm 2.

Noting that (Xutao et al., 2012) has ever provided a similar solution framework for their proposed HAR model, a modified version of MultiRank on the basis of HITS's principle, to cope with query search. Nonetheless, for one thing, our Algorithm 2 is designed specifically for MultiRank; for another, we further explicitly give the expressions of $\mathbf{o}$ and $\mathbf{r}$ for our summarization application in following sections.

3.4.1 Sentence prior

We try to normalize the relatedness of sentences towards the query description as those sentences' prior distribution. The most commonly used method of computing sentence relatedness is cosine measure. Nevertheless, we opt for an approach that is similar with what adopted by

Algorithm 2: The MultiRank Algorithm Integrated with Prior

Input: Two tensors $\mathscr{O}$ and $\mathscr{R}$, two initial probability distributions $\mathbf{x}_0$ and $\mathbf{y}_0$ ($\sum_{s_1=1}^{n}[\mathbf{x}_0]_{s_1}=1$ and $\sum_{z_1=1}^{K}[\mathbf{y}_0]_{z_1}=1$), two prior distributions of sentences and topics $\mathbf{o}$ and $\mathbf{r}$, two weighting parameters $0\leq\mu,\nu<1$ and the tolerance ϵ
Output: Two stationary probability distributions $\bar{\mathbf{x}}$ and $\bar{\mathbf{y}}$
Procedure:

1. Set $t=1$;
2. Compute $\mathbf{x}_t$=(1-μ)$\mathscr{O}\mathbf{x}_{t-1}\mathbf{y}_{t-1}+\mu\mathbf{o}$;
3. Compute $\mathbf{y}_t$=(1-ν)$\mathscr{R}\mathbf{x}_t^2+\nu\mathbf{r}$;
4. If $\|\mathbf{x}_t-\mathbf{x}_{t-1}\|+\|\mathbf{y}_t-\mathbf{y}_{t-1}\|<\epsilon$, then stop, otherwise set $t=t+1$ and goto Step 2.

(Kurland and Lee, 2010) as well as (Erkan, 2006). They all defined the sentence relatedness (e.g., u and v) as the generation probability of u given v or of v given u. Naturally, sentence-word distribution, as an initial step, must be acquired. They assumed that the weight of a word in certain sentence mainly depends on the word's occurrence frequency in that specific sentence. Given a sentence s, a straightforward way of computing the weights of words contained in s is to normalize their term frequency (TF):

$$P_{TF}(w|s)=c_s(w)/|s| \tag{11}$$

where $c_s(w)$ is times of word w occurring in s, and $|s|$ is the total number of words in s.

However, in our perspectives, it should not be neglected that different words have different degrees of information and instructions, no matter for a document or a sentence. For example, three sentences $s_1=(A,C)$, $s_2=(A:0.9,B:0.1)$ and $s_3=(D,B)$, where A, B, C and D denote 4 discriminative words, and 0.9, 0.1 mean the weights of A and B in the second sentence s_2. Now, if we are asked to distinguish which one of s_1 and s_3 has more similarity with s_2, apparently we should choose s_1. Even though the occurrence frequencies of A and B are the same in sentence s_2 (i.e., their generation probabilities are equal according to Equation 11), they actually have different importance towards s_2. As a result, s_1 is supposed to be more similar with s_2 because it has content s_2 more concerns.

With model LDA, we propose to derive the weight of word w with respect to sentence s where w occurs as follows:

$$P'(w|s)=\frac{P(w|C)}{\sum_{w'\in s}P(w'|C)} \tag{12}$$

where C means the entire corpus, and $P(w|C)$ could be easily acquired by combining corpus-topic distribution and topic-word distribution which are both basic outputs of LDA. To account for the unseen words for s, we smooth the above equation like this:

$$P(w|s)=(1-\lambda)P'(w|s)+\lambda P(w|C) \tag{13}$$

λ is a trade-off parameter and is set to 0.7 in experiments. Consequently, we can talk about

the generation probability of sentence u given another sentence v as:

$$\text{gen}(u|v) = P(u|v)^{\frac{1}{|u|}} = [\prod_{w \in u} P(w|v)]^{\frac{1}{|u|}} \tag{14}$$

We use the length of u (i.e., $|u|$) to conduct normalization so as to avoid the tendency that longer sentences get smaller generation probabilities. Moreover, we use a normalized generation probability to indicate the initial bias of sentences towards the query q, i.e.,

$$[\mathbf{o}]_{s_1} = \frac{\text{gen}(q|s_1)}{\sum_s \text{gen}(q|s)} \quad 1 \le s_1 \le n \tag{15}$$

We mark cosine measure as $cosine-based$, mark Equation 11 based generation probability method as $TF-based$, and label our proposed generation probability approach based on Equation 12 as $weight-based$. In the experimental part, they will be compared with each other to investigate their performance in constructing sentence prior.

3.4.2 Topic prior

When it turns to prior distribution of topics, we must first get the topic distribution of query description, i.e., the conditional probability $P(z_1|q)$, so

$$[\mathbf{r}]_{z_1} = P(z_1|q) = \frac{1}{|q|} \sum_{w \in q} P(z_1|w) \quad 1 \le z_1 \le K \tag{16}$$

where $|q|$ means the word number in q, and $P(z|w)$ is derived via Bayesian rule in LDA.

In experiments, we also set the two initial probability distributions in Algorithm 2 to equal with their corresponding initial distribution in Algorithm 1. Illustration of SentTopic-MultiRank with prior beliefs $\mathbf{o}$ and $\mathbf{r}$ is given as Figure 1(c).

4 Summary generation

After ranking process, sentences usually have close values if they own similar content. Consequently, we perform the modified MMR algorithm, proposed in (Wan et al., 2007), during sentence selection to control the information redundancy of generated summary. The principle of MMR algorithm is to reduce the ranking scores of remaining sentences if they are detected to have a degree of similarities with those sentences having been selected to construct a summary. The algorithm goes as follows:

1. Initiate two sets A=∅, B={s_i|i=1,2, ⋯, n}, and each sentence's initial value is set to its SentTopic-MultiRank score obtained in above section, i.e., $value(s_i)$=$\bar{x}_i$.
2. Sort the sentences in B by their current values in descending order.
3. Suppose s_i is the highest ranked sentence, i.e., the first sentence in B. Move sentence s_i from B to A and update the values of the remaining sentence(s) in B as follows:
   For each sentence s_j in B:
   $$value(s_j) = value(s_j) - \omega \cdot \tilde{S}_{ij} \cdot value(s_i) \tag{17}$$
   where ω is set to 5 in our experiments, $\tilde{S}$ is the normalized sentence similarity matrix.
4. Go to step 2 and iterate until $|B|$=0.

After the final scores are obtained for all sentences, several sentences with highest ranking scores are chosen to produce a summary until length limit is reached.

5 Experiments

5.1 Data sets and evaluation metrics

We validate our algorithm framework in both generic and query-biased multi-document summarization. For generic task, we conduct experiments on the data sets DUC2002[2] and DUC2004[3] in which generic multi-document summarization has been one of the fundamental tasks (i.e., task 2 in DUC2002 and task 2 in DUC2004). For query-related task, experiments are based on the main tasks of DUC2005[4] and DUC2006[5]. Each task has a gold standard data set consisting of document sets and reference summaries. Table 1 gives a short summary of above data sets. Documents are pre-processed by segmenting sentences and splitting words. Stop words are removed and the remaining words are stemmed using Porter stemmer[6].

	DUC2002	DUC2004	DUC2005	DUC2006
Task	Task2	Task2	the only task	the only task
Number of documents	567	500	1593	1250
Number of clusters	59	50	50	50
Data source	TREC	TDT	TREC	TREC
Summary length	200 words	665 bytes	250 words	250 words

Table 1: Summary of data sets

We use the ROUGE (Lin, 2004) (version 1.5.5) toolkit[7] for evaluation, which is officially adopted by DUC for evaluating automatic generated summaries. Here we report the average F-measure scores of ROUGE-1, ROUGE-2 and ROUGE-SU4, which base on Uni-gram match, Bi-gram match, and unigram plus skip-bigram match with maximum skip distance of 4 between the candidate summary and the reference summary, respectively.

5.2 System comparison

5.2.1 Generic multi-document summarization

As for generic multi-document summarization, we compare our proposed method SentTopic-MultiRank with following algorithms. (1)Lead Baseline: The lead baseline takes the first sentences one by one in the last document in a document set, where documents are assumed to be ordered chronologically. (2)Random: The method selects sentences randomly for each document collection. (3)DUC Best: The system with highest ROUGE scores among all the systems submitted. (4)LexPageRank: The method first constructs a sentence connectivity graph based on traditional cosine similarity and then conducts PageRank algorithm to determine global importance scores of sentences (Erkan and Radev, 2004). (5)ToPageRank: The method, proposed in (Pei et al., 2012), decomposes traditional PageRank into multiple PageRank via topic decomposition for generic multi-document summarization.

Tables 2-3 show the experimental results. From those statistics, we observe that ToPageRank is better than LexPageRank while our SentTopic-MultiRank outperforms all the competitors.

[2]http://www-nlpir.nist.gov/projects/duc/data/2002_data.html
[3]http://www-nlpir.nist.gov/projects/duc/data/2004_data.html
[4]http://www-nlpir.nist.gov/projects/duc/duc2005/tasks.html
[5]http://www-nlpir.nist.gov/projects/duc/duc2006/tasks.html
[6]http://tartarus.org/ martin/PorterStemmer/
[7]http://www.isi.edu/licensed-sw/see/rouge/

Systems	ROUGE-1	ROUGE-2	ROUGE-SU4
Lead	0.39860	0.16042	0.20315
Random	0.38469	0.11705	0.18007
LexPageRank	0.47473	0.22484	0.25850
DUC Best	0.49869	0.25229	0.28406
ToPageRank	0.49923	0.25735	0.29630
SentTopic-MultiRank	**0.50491**	**0.26714**	**0.30731**

Table 2: *F*-measure comparison on DUC2002

Systems	ROUGE-1	ROUGE-2	ROUGE-SU4
Lead	0.33182	0.06348	0.10582
Random	0.31857	0.06269	0.11780
LexPageRank	0.37875	0.08354	0.12770
DUC Best	0.38279	0.09216	0.13349
ToPageRank	0.40251	0.09555	0.14027
SentTopic-MultiRank	**0.41016**	**0.09917**	**0.14324**

Table 3: *F*-measure comparison on DUC2004

It suggests that: (1) the decomposition and analysis of topic information indeed benefit the improvement of summary quality, as ToPageRank and SentTopic-MultiRank systems perform. It might result from the fact that not only does target corpus usually involve multiple topics, but a user commonly wishes to obtain information from various aspects by reading a concise text. Analyzing corpus on topic level could deeply and comprehensively identify user desired content. (2)ToPageRank focuses on sentence interaction within a topic's range while SentTopic-MultiRank takes three kinds of relations (sentence-sentence, sentence-topic, topic-topic) into consideration wholly. Hence, propagation of impacts among sentences and topics (noting that topics are treated as relation types) is a bonus to determine the saliency score of sentences.

5.2.2 Query-biased multi-document summarization

To validate SentTopic-MultiRank in query-biased task, following systems are implemented as baselines. (1)Random: The same baseline as that in above generic task. (2)Manifold: ranking the sentences according to the manifold ranking scores. (3)Biased LexRank: A modified version of traditional random walk with prior belief. It was presented in (Erkan, 2006) and the sentence prior knowledge was denoted by sentence's similarity to the query description. (4)top three systems with the highest ROUGE scores that participated in the DUC2005 (S4, S15, S17) and the DUC2006 (S12, S23, S24) for comparison, respectively. Tables 4 and 5 present the performance of these systems on DUC2005 and DUC2006 data sets, respectively.

Systems	ROUGE-1	ROUGE-2	ROUGE-SU4
Random	0.30821	0.03976	0.10625
Biased LexRank	0.37324	0.07113	0.12805
Manifold	0.37497	0.07423	0.12907
S17	0.36933	0.07286	0.12937
S4	0.37584	0.07063	0.12868
S15	0.37656	0.07244	0.13248
SentTopic-MultiRank	**0.38803**	**0.07994**	**0.13532**

Table 4: *F*-measure comparison on DUC2005

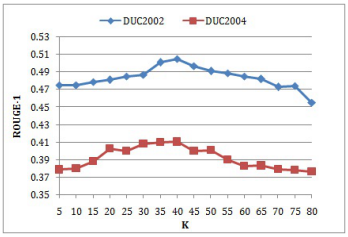
Systems	ROUGE-1	ROUGE-2	ROUGE-SU4
Random	0.34821	0.05297	0.11908
Biased LexRank	0.36814	0.08074	0.12993
Manifold	0.38867	0.08308	0.13307
S23	0.40973	0.09785	0.16162
S12	0.41053	0.09633	0.16074
S24	0.41081	0.09957	0.15248
SentTopic-MultiRank	**0.41567**	**0.10125**	**0.16431**

Table 5: F-measure comparison on DUC2006

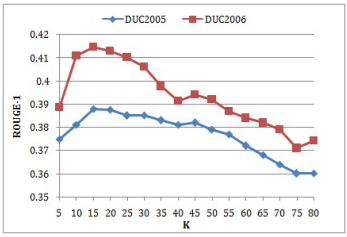
Apparently, our approach SentTopic-MultiRank has the optimal performance, especially outperform the two representative graph-based methods: Biased LexRank and Manifold. Noting that both the sentence networks of Biased LexRank and Manifold are only based on sentence similarity under a single relation type assumption, so the estimation of their sentence ranking scores is limited compared with SentTopic-MultiRank which skillfully considers query-biased information as well as the interactions between sentences and topics.

5.3 Influence of topic number K

LDA is a crucial tool in devising SentTopic-MultiRank and topic number K also denotes the amount of different relation types. In order to investigate how the topic number K exerts influence to system performance, K is varied from 5 to 80. Figures 2(a)-2(b) show the influence of K over ROUGE-1 with respect to generic and query-oriented multi-document summarization, respectively. On the whole, the evaluation scores rise with the increase of K, and reach their respective maximum, then gradually go down when K continues to become larger. Despite some slight fluctuations, they don't matter to the overall conclusion. We could draw the following two meaningful conclusions. (1)It indicates that topic number indeed influences the algorithm result. In fact, our system would be similar with LexPageRank if $K = 1$. (2)Interestingly, two tasks do not reach their optimal situations at an approximate K value, more exactly, generic task requires a relatively large K ($K \approx 40$) while query-biased prefers small K ($K \approx 15$) (here our adoption of an approximation for K is because that we actually vary K every 5 from the start value $K = 5$ in experiments). It may derive from the fact that a query description usually involves few topics, whereas, when applying SentTopic-MultiRank to generic task, we could image the whole corpus as a big query, which concerns about more topics.



(a) ROUGE-1 vs. K for generic multi-document summarization



(b) ROUGE-1 vs. K for query-focused multi-document summarization

Figure 2: Influence of K

5.4 Comparison of three sentence prior methods

In Section 3.4.1, we named *weight-based* for our proposed sentence prior method. In order to confirm its advantage in acquiring more favorable sentence prior distribution, we compare it with two competitors: *cosine-based* and *TF-based* using query-biased summary qualities on DUC2005 and DUC2006. Section 5.3 has indicated that when K is about 15 the quality of query-oriented summary is optimal. For simplicity, we compare the three methods using ROUGE-1 metric under the same condition $K = 15$. Figure 3 gives comparison results. For the space limit, here we give up showing the comparison results on ROUGE-2 and ROUGE-SU4.

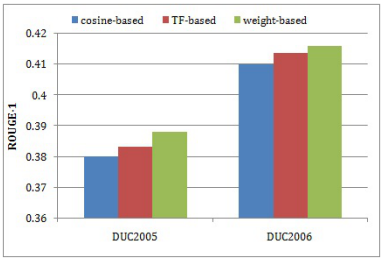


Figure 3: Comparison of three sentence prior methods, $K = 15$.

As Figure 3 shows, generation probability methods, including *TF-based* and *weight-based*, both outperform *cosine-based*. In addition, system using *weight-based* method to generate sentence prior indeed produces summaries with the highest quality. It not only suggests the effectiveness of generation probability methods, but also points out that our method, considering term weights instead of term frequencies to acquire generation probabilities of words given a sentence, is more useful to derive sentence affinity.

Conclusion and perspectives

In this study, we exploit LDA to devise a novel model named SentTopic-MultiRank based on MultiRank algorithm for multi-document summarization. Our intention is to map sentence relatedness over various latent topics to heterogeneous network where multiple topics are treated as various kinds of relations. We apply our approach to both generic and query-biased multi-document summarization. In generic task, we use MultiRank algorithm to directly determine the importance of all sentences, while in query-sensitive task, some query-biased prior beliefs are added to sentences and topics of the SentTopic-MultiRank network, so that some top-ranked sentences are supposed to not only demonstrate the core content of target corpus, but also bias to the information needs of a user's query. Extensive experiments have been performed to prove the effectiveness of our approach in improving the summary quality.

In future work, for one thing, we may apply our model to new summarization task, such as updated summarization; for another, integrating manifold-ranking process with SentTopic-MultiRank might be an interesting idea in some query-related applications.

Acknowledgments

This work is financially supported by Grant SKLSDE-2010KF-03 and NSFC Grant 60933004.

References

Blei, D., Ng, A., and Jordan, M. (2003). Latent dirichlet allocation. *The Journal of Machine Learning Research*, 3:993–1022.

Erkan, G. (2006). Using biased random walks for focused summarization. In *Proceedings of the 2006 Document Understanding Conference*.

Erkan, G. and Radev, D. (2004). Lexpagerank: Prestige in multi-document text summarization. In *Proceedings of EMNLP*, volume 4.

Goldstein, J., Kantrowitz, M., Mittal, V., and Carbonell, J. (1999). Summarizing text documents: sentence selection and evaluation metrics. In *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 121–128. ACM.

Haveliwala, T. (2003). Topic-sensitive pagerank: A context-sensitive ranking algorithm for web search. *Knowledge and Data Engineering, IEEE Transactions on*, 15(4):784–796.

Kleinberg, J. (1999). Authoritative sources in a hyperlinked environment. *Journal of the ACM (JACM)*, 46(5):604–632.

Kolda, T. and Bader, B. (2006). The tophits model for higher-order web link analysis. In *Workshop on Link Analysis, Counterterrorism and Security*, volume 7, pages 26–29.

Kupiec, J., Pedersen, J., and Chen, F. (1995). A trainable document summarizer. In *Proceedings of the 18th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 68–73. ACM.

Kurland, O. and Lee, L. (2010). Pagerank without hyperlinks: Structural reranking using links induced by language models. *ACM Transactions on Information Systems (TOIS)*, 28(4):18.

Lin, C. (2004). Rouge: A package for automatic evaluation of summaries. In *Proceedings of the workshop on text summarization branches out (WAS 2004)*, volume 16.

Lin, C. and Hovy, E. (2002). From single to multi-document summarization: A prototype system and its evaluation. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, pages 457–464. Association for Computational Linguistics.

Liu, Y., Zhong, S., and Li, W. (2012). Query-oriented multi-document summarization via unsupervised deep learning. In *Twenty-Sixth AAAI Conference on Artificial Intelligence*.

Liu, Z., Huang, W., Zheng, Y., and Sun, M. (2010). Automatic keyphrase extraction via topic decomposition. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 366–376. Association for Computational Linguistics.

Manning, C., Schutze, H., and MITCogNet (1999). *Foundations of statistical natural language processing*, volume 999. MIT Press.

Mihalcea, R. and Tarau, P. (2005). A language independent algorithm for single and multiple document summarization. In *Proceedings of IJCNLP*, volume 5.

Ng, M., Li, X., and Ye, Y. (2011). Multirank: co-ranking for objects and relations in multi-relational data. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1217–1225. ACM.

Nie, L., Davison, B., and Qi, X. (2006). Topical link analysis for web search. In *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 91–98. ACM.

Ouyang, Y., Li, S., and Li, W. (2007). Developing learning strategies for topic-based summarization. In *Proceedings of the sixteenth ACM conference on Conference on information and knowledge management*, pages 79–86. ACM.

Page, L., Brin, S., Motwani, R., and Winograd, T. (1999). The pagerank citation ranking: Bringing order to the web.

Pei, Y., Yin, W., Zhang, F., and Huang, L. (2012). Generic multi-document summarization using topic-oriented information. In *Proceedings of the 2012 Pacific Rim International Conference on Artificial Intelligence*, pages 435–446.

Radev, D., Jing, H., Stys, M., and Tam, D. (2004). Centroid-based summarization of multiple documents. *Information Processing & Management*, 40(6):919–938.

Saggion, H., Bontcheva, K., and Cunningham, H. (2003). Robust generic and query-based summarisation. In *Proceedings of the tenth conference on European chapter of the Association for Computational Linguistics-Volume 2*, pages 235–238. Association for Computational Linguistics.

Shen, C. and Li, T. (2011). Learning to rank for query-focused multi-document summarization. In *Data Mining (ICDM), 2011 IEEE 11th International Conference on*, pages 626–634. IEEE.

Shen, D., Sun, J., Li, H., Yang, Q., and Chen, Z. (2007). Document summarization using conditional random fields. In *Proceedings of IJCAI*, volume 7, pages 2862–2867.

Tong, H., Faloutsos, C., and Pan, J. (2008). Random walk with restart: fast solutions and applications. *Knowledge and Information Systems*, 14(3):327–346.

Wan, X., Yang, J., and Xiao, J. (2007). Manifold-ranking based topic-focused multi-document summarization. In *Proceedings of IJCAI*, volume 7, pages 2903–2908.

Xutao, L., Michael, K., and Yuming, Y. (2012). Har: Hub, authority and relevance scores in multi-relational data for query. In *Proceedings of the 2012 SIAM International Conference on Data Mining*, pages 141–152.

Zhou, D., Weston, J., Gretton, A., Bousquet, O., and Schlkopf, B. (2004). Ranking on data manifolds. *Advances in neural information processing systems*, 16:169–176.

Zhou, L. and Hovy, E. (2003). A web-trained extraction summarization system. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1*, pages 205–211. Association for Computational Linguistics.

Language Modeling for Spoken Dialogue System based on Filtering using Predicate-Argument Structures

Koichiro Yoshino[1] *Shinsuke Mori*[1] *Tatsuya Kawahara*[1]
(1) School of Informatics, Kyoto University, Yoshida-Honmachi, Sakyo-ku, Kyoto, Japan.

ABSTRACT
We present a novel scheme of language modeling for a spoken dialogue system by effectively filtering query sentences collected via a Web site of wisdom of crowds. Our goal is a speech-based information navigation system by retrieving from backend documents such as Web news. Then, we expect that users make queries that are relevant to the backend documents. The relevance measure can be defined with cross-entropy or perplexity by the language model generated from the documents in a conventional manner. In this article, we propose a novel criteria that considers semantic-level information. It is based on predicate-argument (P-A) pairs and their relevance to the documents (or topic) is defined by a naive Bayes score. Experimental evaluations demonstrate that the proposed relevance measure effectively selects relevant sentences used for a language model, resulting in significant reduction of the word error rate of speech recognition as well as the semantic-level error rate.

KEYWORDS: Language Modeling, Predicate Argument Structure, Spoken Dialogue System.

Proceedings of COLING 2012: Technical Papers, pages 2993–3002,
COLING 2012, Mumbai, December 2012.

1 Introduction

The tasks of spoken dialogue systems have been extended from simple transactions to general information navigation based upon user requests. Ideally, these systems should handle not only simple, keyword-based queries that current voice search systems respond to but also vague and complex user requests related to, for example, tourist guides or news briefings. This type of application can be achieved through document retrieval in a corresponding domain. For example, we can turn to tourist guidebooks or relevant Wikipedia entries for information on the tourist domain (Misu and Kawahara, 2010). An intelligent dialogue system can be created by restricting the domain and using the knowledge from that domain (Kawahara, 2009). An interactive news navigator that generates dialogues based on news article archives has been developed along this concept (Yoshino et al., 2011).

The automatic speech recognition (ASR) module for spoken dialogue systems (SDSs) needs an appropriate language model (LM) adapted to the task domain and style. Even an ASR system with a very large vocabulary cannot cover all proper nouns or named entities (NEs), which are critical in information retrieval. Ideally, an LM should be trained with a large-scale matched corpus, but in many cases this is not realistic. Therefore, two approaches are commonly adopted. The first involves mixing document texts of the target domain with a dialogue corpus of spoken-style expressions. The other involves collecting relevant texts, possibly from spoken-style sentences, from the Web (Sarikaya et al., 2005; Sethy et al., 2005; Misu and Kawahara, 2006; Bulyko et al., 2007). These approaches try to cover the target domain and style of speech in an indirect way, but the resultant model will inevitably contain a large amount of irrelevant texts.

In general, information navigation systems with speech interfaces have a set of backend documents for retrieval (Schalkwyk et al., 2010). These systems require matching between the backend documents and the user utterance. Based on this assumption, we define a similarity measure of collected sentences from the Web (= expected user utterances) with the backend documents, and select well-matched sentences for LM training.

The overall flow of the proposed method is shown in **Figure 1**. In this paper, we assume two corpora: backend documents (D) and collected sentences (q) from the Web. Superficial matching, in which the method filters sentences based on the similarity in word sequences to the backend documents, is described in Section 2. N-gram model likelihood based on KL divergence is used in this method, and it is equivalent to the conventional perplexity-based method. In Section 3, we propose using deep semantic similarity based on P-A structures. This proposed method provides better filtering, considering not only surface words but also semantic cases. It is suitable forinformation navigation systems that require P-A structures. In Section 4, we propose the combined usage of the two filtering methods mentioned above. The method enables us to take both advantages of the two methods. We evaluate the conventional and proposed methods with ASR accuracy (word error rate (WER)), predicate-argument structure error rate (PAER), and test set perplexity in Section 5.

2 KL Divergence of N-gram Models

We construct an LM for ASR by using a collection of question-style queries (=sentences) from the Web, and matching them with the target backend documents D. For the matching of a question-style sentence q and backend documents D, first, we use KL divergence of LMs for surface word matching. KL divergence is a non-symmetric measurement of the difference

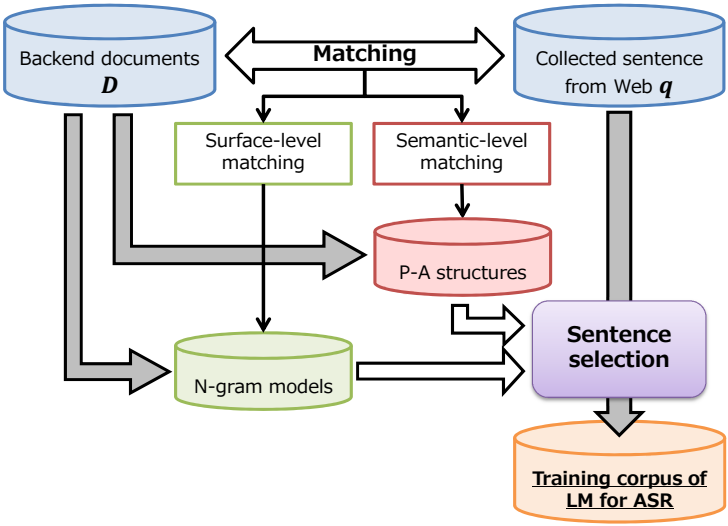


Figure 1: Overview of the proposed method

between two probability distributions (Kullback and Leibler, 1951). KL divergence of sentence q and backend documents D is defined as

$$KL(q||D) = \sum_m P_q(w_m) \log_2 \frac{P_q(w_m)}{P_D(w_m)}, \quad (1)$$

where $w_1 w_2, ..., w_n$ is the word sequence in sentence q. P_D and P_q are the probability distributions of D and q. We define these probability distributions with n-gram (tri-gram) models. We assume that query q consists of one sentence, and most of the time the n-gram distribution trained from only one sentence becomes unique. Then, the probability $P_q(W)$ becomes to 1. We can then make the approximation to KL divergence:

$$KL(q||D) \approx \sum_m \log_2 \frac{1}{P_D(w_m)} \quad (2)$$

$$= -\sum_m \log_2 P_D(w_m). \quad (3)$$

This formulation is equivalent to cross-entropy XE:

$$XE(q,D) = -\sum_m P_q(w_m) \log_2 P_D(w_m) \quad (4)$$

$$\approx -\sum_m \log_2 P_D(w_m). \quad (5)$$

The cross-entropy has two components: self information $P_q(w_m)$ and mutual information $P_D(w_m)$, but the self information $P_q(w_m)$ is equal to 1 as discussed above. These formulations

can be mapped to the definition of perplexity PP.

$$H(q,D) = -\frac{1}{n}\sum_{m=1}^{n}\log_2 P_D(w_m). \quad (6)$$

$$PP(q,D) = 2^{H(q,D)}. \quad (7)$$

There is already a method to filter sentences q based on the perplexity defined with targeted backend documents D (Misu and Kawahara, 2006). In this approach, we construct an n-gram model from the backend documents D and use the perplexity as a superficial similarity measure. This is equivalent to the KL divergence between q and D. We compute the perplexity $PP(q,D)$ to every collected sentence, and then rank them for the filtering.

3 Similarity based on Predicate-Argument Structure

In the conventional filtering method described in the previous section, the matching is performed using word-level similarity measure. However, it is difficult to select sentences that semantically match the backend documents because this method is based on surface information. This is problematic because information navigation systems work based on semantic structures of user utterance (Yoshino et al., 2011). We therefore propose a similarity measure based on P-A structures to select sentences that match the backend documents on the deep semantic level. In this section, our focus is on the semantic structures that are defined with P-A structures.

In the conventional approach, the similarity measure is defined with a generative model $P_D(w_i)$. In this section, we adopt a discriminative approach and calculate $P(D|w_i)$ to predict the significant sentences.

3.1 Predicate-Argument Structure

The P-A structure is automatically generated by a semantic parser (**Figure 2**). This P-A structure has a sub-structure that contains a predicate (w_p), argument (w_a), and its semantic case (w_s) (called a "P-A pair"). The P-A structures consist of various arguments that depend on one predicate with its semantic case. We used the JUMAN/KNP [1] analyzer to parse sentences and obtain the structures automatically. However, not every P-A pair is meaningful in information navigation; actually, only a fraction of the patterns are useful. For example, in the baseball domain, key patterns include "[A (agent) beat B (object)]" and "[A (agent) hit B (object)]", and in the business domain, "[A (agent) sell B (object)]" and "[A (agent) acquire B (object)]". The useful information structure is depending on the domain, and information extraction techniques have been investigated (Grishman, 2003). Conventionally, templates for information extraction have been hand-crafted (Ramshaw and Weischedel, 2005), but this heuristic process is so costly that it cannot be applied to the wide variety of domains on the Web. A method to automatically define domain-dependent templates for information extraction to be used in a flexible information navigation system has been proposed (Yoshino et al., 2011).

3.2 Significance Measure based on P-A structures

We extract important P-A pairs from the backend documents D and create matches between the extracted pairs and a question sentence q to measure the similarity based on the semantic

[1] http://nlp.ist.i.kyoto-u.ac.jp/index.php?JUMAN
http://nlp.ist.i.kyoto-u.ac.jp/index.php?KNP

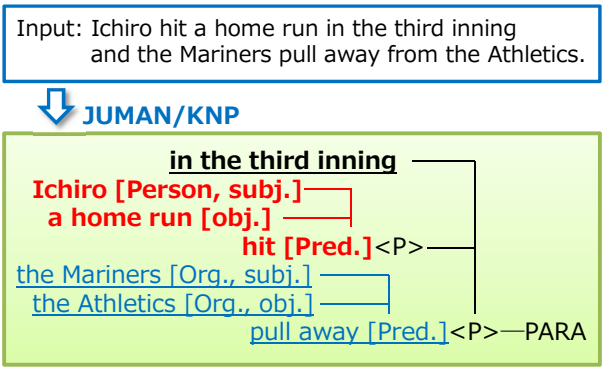


Figure 2: Example of predicate-argument (P-A) structure extraction.

significance. A previous study (Yoshino et al., 2011) has shown that an extraction method based on a Naive Bayes classifier is effective. In this method, the conditional probability of a document D (e.g., baseball) given a word w_i is defined as

$$P(D|w_i) = \frac{C(w_{i,D}) + x_D\gamma}{C(w_i) + \gamma}, \quad (8)$$

where γ is a smoothing factor estimated with a Dirichlet prior (Teh et al., 2006) using the Chinese Restaurant Processes (CRP). To calculate the conditional probabilities, we use sentences $\bar{D}$ of other domains that are extracted at random with the same population of D. x_D, a normalization factor that depends on the size of D, is defined as

$$x_D = \frac{\sum_j C(w_j, D)}{\sum_k C(w_k)}. \quad (9)$$

The P-A structure has a minimum sub-structure P-A pair (PA_i) containing the predicate (w_p), argument (w_a), and its semantic case (w_s). With the above definition, we define the conditional probability $P(D|PA_i)$ as

$$P(D|PA_i) = \sqrt{P(D|w_p, w_s) \times P(D|w_a)}. \quad (10)$$

3.3 Clustering of Named Entities

The statistical method often encounters the problem of data sparseness due to mismatch between the training set and the test set, especially with the named entities (NEs). To solve this problem, we cluster NEs that appear in the training set. NEs are one of the information structures that can be automatically generated by a semantic parser in accordance with a pre-defined category. An example of automatically labeled NEs is shown in the **Figure 2**, in which a labeler assigns person and organization labels to the entities.

We perform clustering to classify P-A structures that have the same trio of predicate, semantic case, and NE. In the example shown in **Figure 3**, two P-A structures that have the same abstract

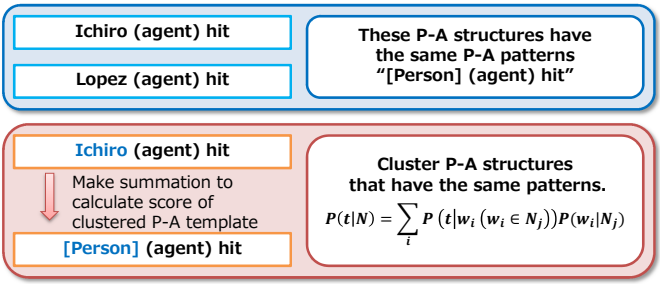


Figure 3: Clustering of named entities (NEs).

P-A pairs are clustered to the same template. We extend the probability of argument $P(w_a)$ as

$$P(D|N_i) = \sum_{k(w_k \in N_i)} P(D|w_k)P(w_k), \quad (11)$$

where N_i is the NE class that is included in arguments w_a. This clustering enables us to reduce lexicon mismatches between the backend documents D and question sentences q, leading to more robust matching.

3.4 Filtering with P-A Templates

For each question sentence q, we calculate the mean of every $P(D|PA_i)$ contained in the sentence q, which is defined as $P(D|q_{PA})$. An example of this scoring is shown in **Figure 4**. The input sentence q has four P-As. We take the mean of their scores to calculate $P(D|q_{PA})$.

Every sentence is ranked with $P(D|q_{PA})$, for selection to be used to train the LM for ASR. With this method, we can select sentences that are more relevant to the backend documents and more likely to be asked by users.

4 Combination of Sentence Selection Methods

We described two similarity measures for the backend documents D and question sentences q in Section 2 and 3. Considering their advantages, we propose a combination of the two methods using ranks and scores.

4.1 Method based on Sentence Rank

In this method, we sort question sentences q with the above-described $PP(q, D)$ and $P(D|q_{PA})$ and rank them as PP_{rank} and PA_{rank}. We then re-sort the sentences by summing the two ranks: $PP_{rank} + PA_{rank}$, for final selection.

4.2 A Method using Normalized Score

We re-define a new score by using $PP(q, D)$ and $P(D|q_{PA})$. The range of $P(D|q_{PA})$ is $0 < P(D|q_{PA}) < 1$, but the range of $PP(q, D)$ is not $0 < PP(q, D) < 1$, so we content it via the

P-A structure

q = "**Ichiro hit a game-winning double when the bases were loaded with two outs in the final inning.**"
PA = ["[Person]/subject/hit",
"a game-winning double/object/hit",
"the bases were loaded with two outs/locative/hit",
"final inning/modifier/hit"]

P-A templates

Score	Argument	case	Predicate	
0.99599	middle relievers		subject	lose
0.99519	relief pitcher		subject	lose
0.98716	final inning		modifier	hit
0.98202	a game-winning double		object	hit
0.98201	the bases were loaded with two outs		locative	hit
0.78062	[Person]		subject	hit
0.09994	share price		subject	slide
0.09994	charge		subject	increase
	...			

Figure 4: An example of $P(D|q_{PA})$ calculation.

sigmoid function.

$$PP' = \frac{1}{1 + e^{-PP}}. \tag{12}$$

A mixing ratio of 3:7 was set after a trial experiment.

5 Experimental Evaluation

We evaluated LMs constructed by the proposed methods with ASR. These LMs were constructed from selected sentences by using the four proposed filtering methods. We compared these models by using three indexes: word error rate (WER), predicate-argument structure error rate (PAER), and adjusted perplexity. PAER is based on the parsing accuracy of recognized sentences. Since the LMs have a different vocabulary size, we used adjusted perplexity, which penalizes smaller-vocabulary LMs.

5.1 Experimental Setting

We prepared an evaluation task using a news navigation system (Yoshino et al., 2011) in the professional baseball domain. Details of the test sets are shown in **Table 1**.

To train $PP(q, D)$ and $P(D|q_{PA})$, we prepared backend documents D of baseball articles from Mainichi newspaper articles (CD-Mainichi newspaper database 2000–2009). To train the LM for ASR, we used question-style sentences q taken from the baseball domain in the Yahoo! QA corpus [2] (a collection of queries on a Web site). The specification of the corpus are shown in **Table 2**. We used Julius[3] (Lee and Kawahara, 2001) as the ASR engine.

[2]This corpus was provided by Yahoo!JAPAN and National Institute of Informatics, Japan.
[3]http://julius.sourceforge.jp

Table 1: Specification of test sets.

Task	Users	Utterances
News navigation	10	2,747

Table 2: Specification of training sets.

Usage	Corpus	Sentences
Backend documents D	Mainichi newspaper articles	176,852
Pool of sentences q for training	Yahoo!QA entertainment: Baseball	403,602

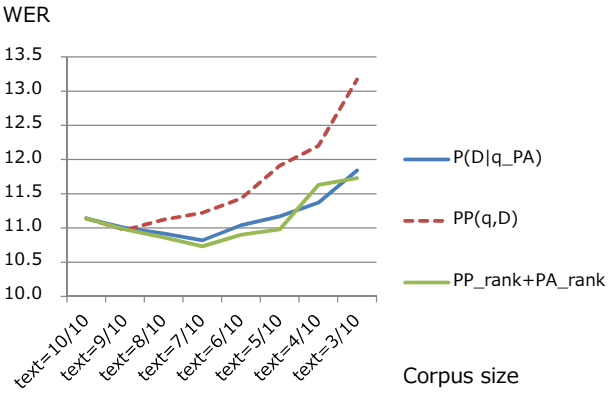


Figure 5: WER on news navigation task.

5.2 Experimental Results

The WER, PAER, and adjusted perplexity results are shown in **Figure 5, 6, and 7**, respectively. The horizontal axes are the relative size of the training set. There was a significant difference between the proposed method ($P(D|q_{PA})$, text $= 7/10$) and the baseline (text $= 10/10$, no filtering), with the significance level of less than 0.05 ($p < 0.05$.) There is no significant difference between the proposed method $P(D|q_{PA})$ and the combined $PP_{rank} + PA_{rank}$ rank or the combined $PP(q, D)$ and $P(D|q_{PA})$ score. Only the combined $PP_{rank} + PA_{rank}$ rank is shown in these graphs because there was not a distinguishable difference between the rank-based method and the normalized score-based method.

The experimental evaluation of WER shows that the similarity measure based on the semantic-level performed better than the conventional surface-level matching using n-gram models. The PAER evaluation also showed that the proposed method using a combination of ranks performed better than the conventional method. This demonstrates that using deep semantic-level similarity can improve the ASR accuracy.

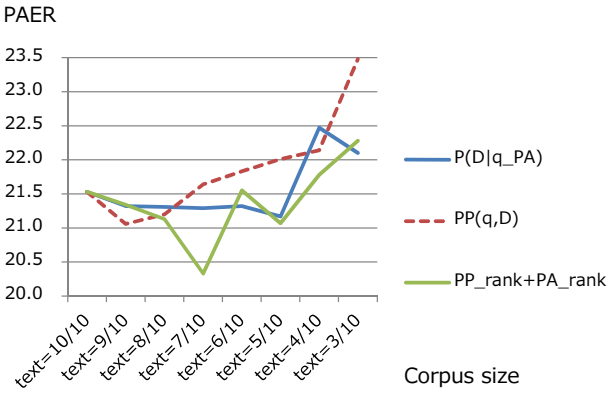


Figure 6: PAER on news navigation task.

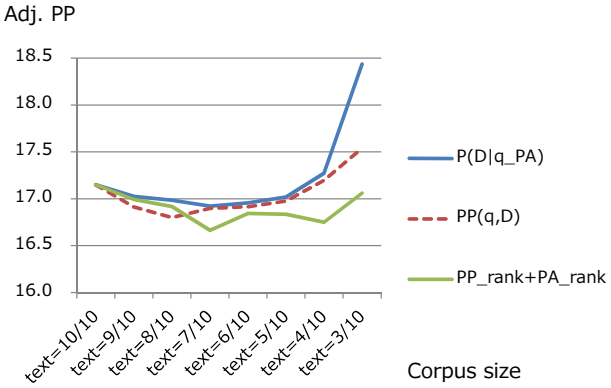


Figure 7: Adjusted perplexity on news navigation task.

6 Conclusion

We have proposed a method of sentence selection for language modeling for information navigation systems. The proposed method features sentence filtering based on semantic-level similarity. Experimental results showed that the proposed method performs better than the conventional method which uses only filtering based on surface-level perplexity. Moreover, the combinational usage of these two measures improve the ASR performance. Especially, filtering based on the P-A structure contributes the improvement of P-A structures accuracy (=reduce the PAER). As a future work, we plan to apply the method to a variety of domains.

References

Bulyko, I., Ostendorf, M., Siu, M., Ng, T., Stolcke, A., and Çetin, O. (2007). Web resources for language modeling in conversational speech recognition. *ACM Trans. Speech Lang. Process.*, 5(1):1:1–1:25.

Grishman, R. (2003). Discovery methods for information extraction. In *Proc. ISCA & IEEE Workshop on Spontaneous Speech Processing and Recognition*, pages 243–247.

Kawahara, T. (2009). New perspectives on spoken language understanding: Does machine need to fully understand speech? In *Proc. IEEE-ASRU*, pages 46–50.

Kullback, S. and Leibler, R. (1951). On information and sufficiency. *The Annals of Mathematical Statistics*, 22(1):79–86.

Lee, A. and Kawahara, T. (2001). Julius–an open source real-time large vocabulary recognition engine. In *Proc. EuroSpeech*, pages 1691–1694.

Misu, T. and Kawahara, T. (2006). A bootstrapping approach for developing language model of new spoken dialogue system by selecting web texts. In *INTERSPEECH*, pages 9–13.

Misu, T. and Kawahara, T. (2010). Bayes risk-based dialogue management for document retrieval system with speech interface. *Speech Communication*, 52(1):61–71.

Ramshaw, L. and Weischedel, R. M. (2005). Information extraction. In *IEEE-ICASSP*, volume 5, pages 969–972.

Sarikaya, R., Gravano, A., and Gao, Y. (2005). Rapid language model development using external resources for new spoken dialog domains. In *Proc. ICASSP*, volume 1, pages 573–576.

Schalkwyk, J., Beeferman, D., Beaufays, F., Byrne, B., Chelba, C., Cohen, M., Garret, M., and Strope, B. (2010). Google search by voice: A case study.

Sethy, A., Georgiou, P. G., and Narayanan, S. (2005). Building Topic Specific Language Models from Webdata Using Competitive Models. In *Proc. Interspeech*, pages 1293–1296.

Teh, Y. W., Jordan, M. I., Beal, M. J., and Blei, D. M. (2006). Hierarchical dirichlet processes. *Journal of the American Statistical Association*, 101:1566–1581.

Yoshino, K., Mori, S., and Kawahara, T. (2011). Spoken dialogue system based on information extraction using similarity of predicate argument structures. In *Proc. of SIGDIAL*, pages 59–66.

Detecting Word Ordering Errors in Chinese Sentences for Learning Chinese as a Foreign Language

Chi-Hsin Yu, Hsin-Hsi Chen
Department of Computer Science and Information Engineering, National Taiwan University
#1, Sec.4, Roosevelt Road, Taipei, 10617 Taiwan
jsyu@nlg.csie.ntu.edu.tw, hhchen@ntu.edu.tw

ABSTRACT

Automatic detection of sentence errors is an important NLP task and is valuable to assist foreign language learners. In this paper, we investigate the problem of word ordering errors in Chinese sentences and propose classifiers to detect this type of errors. Word n-gram features in Google Chinese Web 5-gram corpus and ClueWeb09 corpus, and POS features in the Chinese POS-tagged ClueWeb09 corpus are adopted in the classifiers. The experimental results show that integrating syntactic features, web corpus features and perturbation features are useful for word ordering error detection, and the proposed classifier achieves 71.64% accuracy in the experimental datasets.

協助非中文母語學習者偵測中文句子語序錯誤

自動偵測句子錯誤是自然語言處理研究一項重要議題，對於協助外語學習者很有價值。在這篇論文中，我們研究中文句子語序錯誤的問題，並提出分類器來偵測這種類型的錯誤。在分類器中我們使用的特徵包括：Google 中文網路 5-gram 語料庫、與 ClueWeb09 語料庫的中文詞彙 n-grams及中文詞性標注特徵。實驗結果顯示，整合語法特徵、網路語料庫特徵、及擾動特徵對偵測中文語序錯誤有幫助。在實驗所用的資料集中，合併使用這些特徵所得的分類器效能可達 71.64%。

KEYWORDS : ClueWeb09, computer-aided language learning, HSK corpus, word ordering error detection
KEYWORDS IN L_2 : ClueWeb09, 電腦輔助語言學習, HSK 語料庫, 語序錯誤偵測

Proceedings of COLING 2012: Technical Papers, pages 3003–3018,
COLING 2012, Mumbai, December 2012.

1 Introduction

Non-native language learners usually encounter problems in learning a new foreign language and are prone to generate ungrammatical sentences. NLP systems that detect and correct grammatical errors are important and invaluable to language learners. Error detection, which tells if there exists a special type of errors in a given sentence, is the first step for this kind of applications (Leacock, Chodorow, Gamon, & Tetreault, 2010).

Sentences with various types of errors are written by language learners of different backgrounds. The distribution of errors varies across different learner groups. For example, the most frequent error for native English learners is missing comma after introductory element, while the most frequent error for ESL (English as a Second Language) students is comma splice, which joins two sentences by using comma instead of a conjunction (Leacock et al., 2010). In addition, learners may generate sentences with multiple types of errors. This complicates the error detection. To simplify the problem and evaluate the performance of the proposed methods, researchers usually detect one type of error at a time, such as common grammatical errors (Islam & Inkpen, 2011; Wagner, Foster & Genabith, 2007) and prenominal adjective ordering errors (Lin et al., 2009; Malouf, 2000).

A training dataset is indispensable for learning an error detection system. In English, there are many well-established corpora for this purpose such as Cambridge Learner Corpus (CLS). In Chinese, Beijing Language and Culture University built an HSK dynamic composition corpus (动态/dynamic 作文/composition 语料库/corpus) and announced a publicly accessible online system to query this corpus[1]. The HSK corpus contains Chinese compositions written by Chinese language learners in the university. The students' compositions were collected and annotated with different error types. In this paper, we will deal with the word ordering errors in the HSK corpus.

This paper is organized as follows. In Section 2, we survey the related work. In Section 3, we introduce the HSK corpus and specify the type of word ordering errors in Chinese. In Section 4, we describe a web-scale Chinese POS-tagged dataset developed from the ClueWeb09 dataset[2] for this study. In Section 5, we present our feature extraction approaches in detail. Section 6 specifies the experimental setups, and Section 7 presents and analyzes the results. The uses of POS bigrams in the detection of different error types are discussed. Finally, we conclude our study.

2 Related Work

Word ordering errors (WOEs) are defined to be the cases where words are placed in the wrong places in sentences. Word ordering error detection is not only useful for language learning, but also beneficial for many applications such as machine translation. For example, we can use a word ordering error detection module to assess the quality of the machine generated sentences. Lee et al. (2007) employ machine translated sentences as training data to rank the fluency of sentences written by non-native learners in English. Their approaches result in accuracy 76.2%.

[1] http://202.112.195.192:8060/hsk/login.asp ; accessd 2012/08/14.
[2] http://lemurproject.org/clueweb09/ ; accessd 2012/08/14.

Leacock et al. (2010) give thorough surveys in automated grammatical error detection for language learners (native and second language learners). Error types, available corpora, evaluation methods, and approaches for different types of errors are specified in the book.

Several approaches have been proposed to detect English grammatical errors (Chodorow & Leacock, 2000) and Japanese Learners' English Spoken Data (Izumi, Uchimoto, Saiga, Supnithi, & Isahara, 2003). Chodorow and Leacock's approach is tested only on 20 specific English words. Wagner et al. (2007) deal with common grammatical errors in English. They consider frequencies of POS n-grams and the outputs of parsers as features. Their classification accuracy with decision tree is 66.0%. Gamon et al. (2009) identify and correct errors made by non-native English writers. They first detect article and preposition errors, and then apply different techniques to correct each type of errors. Word information such as heads of noun phrases and verb phrases, and POS n-grams in a given context is adopted for error detection. A POS context is defined to be the left and the right of a given preposition. The detection performance varies across different error types. A language model is trained on English Gigaword corpus (Linguistic Data Consortium [LDC], 2003) for error correction. The corrected version whose score is higher than a threshold is proposed as the result.

In addition to newswire text data, a large scale web corpus is also explored. Bergsma, Lin, and Goebel, (2009) adopt Google English Web 1T 5-gram corpus (Brants & Franz, 2006) to compute the frequencies of word n-grams and achieve 95.7% in spelling correction and 75.4% accuracy in preposition selection. Islam and Inkpen (2010) adopt the same corpus for unsupervised preposition error correction. An improved version of Google Web 1T 5-gram corpus called Google N-gram V2 (Lin et al., 2009; Lin et al., 2010) is constructed by adding POS information. Google N-gram V2 contains POS sequence information for each word N-gram pattern. Consider the following unigram example. A word "files" occurs as NNS 611,646 times among all its occurrences:

files 1643568 NNS|611646 VBZ|1031992

Lin et al. (2009) present tools for processing Google N-gram V2 and report tasks that can be benefited by using this corpus, such as base noun phrase parsing and prenominal adjective ordering.

The above research shows that features extracted from English POS-tagged Web corpus are useful. In this paper, we will explore the usefulness of a Chinese POS-tagged web corpus in the application of word ordering error detection for learning Chinese as a foreign language. In addition to the traditional POS and parsing features extracted from this corpus, we will study the effects of different Chinese segmentation methods and perturbation of terms in training the error detector.

The major contributions of this paper cover the following four aspects: (1) application aspect: detecting a common type of Chinese written errors of foreign learners with HSK corpus; (2) language aspect: considering the effects of character-based (without segmentation) and word-based (with segmentation) features in Chinese sentences; (3) resource aspect: exploring the feasibility of using a web-scale word and POS-tagged corpus; and (4) technology aspect: exploring the syntactic features and the web corpus features, and comparing the effects of training corpora by foreign learners and native writers.

3 Word Ordering Errors in Chinese

This section introduces the HSK corpus in our study, and specifies the type of word ordering errors in Chinese.

3.1 HSK Corpus

HSK corpus was built in a project led by Cui Xiliang from 1992 to 2005. This corpus contains 4.24 million characters and 2.83 million words in 11,569 students' compositions. Those students come from different countries to study Chinese in Beijing Language and Culture University. Their compositions are scanned, translated to texts, and annotated with different error types. This corpus also contains students' metadata such as age, country, and language skill level. Since its announcement in 2006, this corpus inspires many research efforts in different fields such as Chinese language teaching and learning.

In the HSK corpus, total 46 error types are labeled. The errors range from character level, word level, sentence level, to discourse level. For some error types, such as missing word error, correct answers are also annotated, so that they can be employed to investigate error detection and correction problems.

3.2 Word Ordering Errors in Chinese

The types of word ordering errors (WOEs) in Chinese are different from that in English. In English, characters are meaningless, while each Chinese character has its own meaning in context. Learners taking Chinese as a foreign language often place character(s) in the wrong places in sentences, and that results in wrong word(s) or ungrammatical sentences. In the HSK corpus, there are 35,884 errors at sentence level, and WOEs are the most frequent type of errors at this level. Table 1 lists the top 10 error types on sentence level in HSK. They belong to different error categories defined in HSK.

#	Error Category	Error Type	Count
1	Other Error Types	Word ordering error	8,515
2	Missing Component	Adverb	3,244
3	Missing Component	Predicate	3,018
4	Grammatical Error	"Is (是)… DE(的)" sentence	2,629
5	Missing Component	Subject	2,405
6	Missing Component	Head Noun	2,364
7	Grammatical Error	"Is (是)" sentence	1,427
8	Redundant Component	Predicate	1,130
9	Other Error Types	Uncompleted sentence	1,052
10	Redundant Component	Adverb	1,051

TABLE 1 – Top 10 types of sentence level errors in HSK

In the HSK website, 8,515 sentences contain WOEs. We collect these sentences and sample a subset of 200 sentences for deep analysis. In this subset, we further classify these WOEs into five categories, including adverb ordering error, subject/verb/object ordering error, prepositional phrase (PP) position error, prenominal adjective ordering error, and others. C*overt error*

sentences (Carl, 1998), which can be interpreted correctly in some specific way, belong to the "others" category. Table 2 lists an example for each category along with its distribution.

Adverb ordering error (35.0%)	也 她 很 关心 also she (is) very concerned (她也很关心)
Subject/verb/object ordering error (32.0%)	就这样 我 休学 大学 来 中国 了 therefore I drop out university (and) come to China (就这样我大学 休学 来中国了)
PP position error (20.5%)	我 留学 在 贵国 I (am) studying in your country (我 在贵国 留学)
Prenominal adjective ordering error (6.0%)	我 遇到了 才貌双全的 一位 女人 I meet beautiful and wise a woman (我遇到了一位 才貌双全的女人)

TABLE 2 – Categories of word ordering errors

The word(s) in an erroneous position are underlined. English words are added to make the examples easy to read. In addition, the correct Chinese sentence is parenthesized below the English one. We can see that adverb ordering and subject/verb/object ordering errors are the two most frequent error categories. Because grammatical error is one category of word ordering errors, parsing may be helpful in this task.

The second and third examples are two interesting examples. In the second example, learner translate "drop out" to 休学 and "university" to 大学 in the same order, but, in Chinese, the reversed order 大学 休学 is more fluent. In the third example, learner translate "am studying" to 留学 and "in your country" to 在贵国, but, in Chinese, changing the position of the PP 在贵国 results in a more fluent sentence.

4 A Web-Scale Chinese POS-Tagged Corpus

We use a publicly available Chinese Web POS 5-gram corpus (CP5g) (Yu, Tang & Chen, 2012) for our experiments. The Chinese POS-tagged corpus has been developed based on the ClueWeb09 dataset, which is a huge web document collection crawled by LTI at CMU. The ClueWeb09 corpus contains documents in ten languages including English, Chinese, and so on. There are 177,489,357 Chinese pages in the ClueWeb09. Although most of the Chinese records are crawled from mainland China, there are still some pages in other languages, such as Japanese. In addition, a Chinese web page may be in different encodings, e.g., Big5, CNS 11643, GBK, GB2312, and Unicode. Thus, the encoding detection and language identification problems have to be dealt with in the development of the CP5g dataset.

After identifying the encoding scheme and the written language, they use the Stanford Chinese word segmenter (Tseng, Chang, Andrew, Jurafsky, & Manning, 2005) to segment the text, and adopt the Stanford POS tagger (Toutanova, Klein, Manning, & Singer, 2003) to tag Chinese sentences. After all n-gram (n=1, 2, 3, 4, 5) patterns are extracted, those patterns that occur less than 40 times are filtered out. The output format is similar to Google N-gram V2.

We also use the Google Chinese Web 5-grams (GC5g) (Liu et al., 2010) corpus for comparison. The size of GC5g is larger than that of CP5g, but GC5g does not contain linguistic information. In addition, GC5g and CP5g are developed by different segmentation systems, i.e., a maximum probability segmenter and a CRF-based segmenter, respectively. That may result in different word patterns and introduce uncertainty into the systems. We will consider these issues when discussing feature extraction of our approaches.

5 Detection of Word Ordering Errors

The detection of word ordering errors (WOEs) is formulated as a binary classification problem. We employ LIBSVM (Chang & Lee, 2011) to build a classifier which detects whether there is a word ordering error in a given Chinese sentence. Different types of features are extracted from various sources in the following sections.

5.1 Syntactic Features

Syntactic features include tagging and parsing results of a sentence. The motivation behind this approach is that the sentence with word ordering errors may result in abnormal POS and parsing patterns, and these linguistic clues are useful for WOE detection.

For the application of tagging features, a sentence is segmented and tagged by using the Stanford segmenter and tagger (Toutanova et al., 2003; Tseng et al., 2005). The performance of POS tagging is 94.13%. The POS *n*-grams (n = 2, 3, 4) in the sentence are extracted, and used as in the bag-of-words approach. In other words, these POS n-grams are treated like words and considered as the features for classification. We use B_2, B_3 and B_4 to refer to the approaches of POS bi-gram, tri-gram, and 4-gram, respectively.

For the applications of parsing features, the Stanford parser is used to analyze the dependency relations in a sentence. Consider the following example.

绿色/JJ 食品/NN 当然/AD 是/VC …
green food definitely is

The Stanford parser generates a set of relations as follows for this example, where the number after each word denotes the word position in the sentence.

amod(*食品-2, 绿色-1*) {*amod*(*food-2,green-1*)}
advmod(*是-4,当然-3*) {*advmod*(*is-4,definitely-3*)}
…

The word in a relation is replaced by its POS tag. For example, *amod*(*食品-2*, *绿色-1*) {*amod*(*food-2*,*green-1*)} is converted into *amod*(*NN*, *JJ*). These kinds of relations are collected, and used as features similar to the bag-of-words approach, in which the relation between two tags is regarded as a word. We use B_p to denote this approach.

Parsing features can be used individually, or integrated with all tagging features to form a larger feature vector B = (B_2, B_3, B_4, B_p) for a sentence. We will explore the effectiveness of all the alternatives in the later experiments.

5.2 Web Corpus Features

Web corpus features are extracted from two reference corpora, i.e., GC5g and CP5g corpora. Intuitively, the sentences with word ordering errors are less likely to occur in the language usages of real world. To measure the probability of a sentence with respect to a reference corpus, we use point-wise mutual information (PMI) (Church & Hanks, 1990). The PMI of a word pair (w_1, w_2) and the score c_g for a sentence that uses GC5g corpus are defined as follows.

$$PMI\ (w_1, w_2) = \log \frac{p(w_1, w_2)}{p(w_1)\,p(w_2)} \quad (1)$$

$$c_g = \frac{1}{L-1} \sum_{l=1}^{L-1} \log \frac{p(w_l, w_{l+1})}{p(w_l)\,p(w_{l+1})} \quad (2)$$

In (2), L is the sentence length in terms of the number of words. We divide the sum of PMIs by sentence length minus 1 for normalization.

To avoid the zero count problems in the above computation, we need some smoothing method. Section 4 specifies that those word pairs occurring less than 40 have been removed from the reference corpora. Thus, there are two cases for an unknown pair: a pair does not appear before, or it occurs, but its total occurrences are less than 40. However, we cannot know which case an unknown pair belongs to. Here, we adopt the same smoothing approach for these two cases. We introduce a $k \in \{1, 10, 20, 30, 39\}$ to replace the zero count, where $k = 1$ means we ignore this unknown word pair, and $k = 39$ is the largest value because the minimum occurrence in the reference corpora is 40.

This approach results in a 5-tuple feature vector $C_g = (c_{g,k})$, $k \in \{1, 10, 20, 30, 39\}$ for a sentence, when the GC5g corpus is regarded as a reference corpus. Similarly, the feature vector $C_b = (c_{b,k})$ is used if the CP5g corpus is adopted to compute the occurrence frequency.

The two feature vectors, C_g and C_b, can be directly input to an SVM classifier. In this way, there is no need to set a fixed threshold. As a usual preprocessing step of SVM training, we also scale each dimension to range [-1, 1].

In addition to word level features, we also consider POS level features. We utilize the tag information in the CP5g corpus to derive a probability of a sentence. The probability is defined as follows.

$$c_{p1} = \frac{1}{L} \sum_{l=1}^{L} \log\ p(t_l \mid w_l) = \frac{1}{L} \sum_{l=1}^{L} \frac{p(t_l, w_l)}{p(w_l)} \quad (3)$$

$$c_{p2} = \frac{1}{L-1} \sum_{l=1}^{L-1} \log \frac{p(t_l, t_{l+1}, w_l, w_{l+1})}{p(w_l, w_{l+1})} \quad (4)$$

where t_l is the POS tag of word w_l, c_{p1} is a normalized probability of a sentence to be tagged as $t_1 t_2 \ldots t_L$ by POS unigrams, and c_{p2} is a normalized probability of a sentence by POS bigrams. Similar to C_g, we have to face the unknown pair problem. Here, we adopt the similar smoothing approach, i.e., let $k \in \{1, 10, 20, 30, 39\}$ be the occurrence count of an unknown pair. We represent a sentence by three alternatives: a 5-tuple vector $C_{p1} = (c_{p1,k})$, a 5-tuple vector $C_{p2} = (c_{p2,k})$, or a 10-tuple vector $C_p = (C_{p1}, C_{p2})$.

5.3 Perturbation of Terms in a Sentence

Motivated by Gamon et al.'s work (2009) that calculates scores derived from a language model to filter out improper correction, we perturb a sentence to see how the features respond to the perturbation. We randomly choose two words and switch their positions to simulate the word ordering errors generated by language learners. After that, we extract different types of features from the perturbed sentences as the procedures described in Sections 5.1 and 5.2.

Let pB_n be a perturbed version of B_n for the perturbed sentence. Similarly, we have all perturbed versions of features. As an alternative to pB_n, we consider the difference to the original sentence and use $\triangle pB_n = pB_n - B_n$ as the features.

5.4 Combining Features

We combine features from different sources to get better models. At first, we cosine-normalize a vector to a unit vector. For example, C_b is normalized by the following formula.

$$\overline{C}_b = \frac{C_b}{\| C_b \|} \tag{5}$$

We then combine normalized vectors for experiments. Here, we equally emphasize different types of features. For example, when combining vectors B_2 and C_b, we normalize B_2 and C_b into $\overline{B}_2$ and $\overline{C}_b$, and combine them to a new vector $(\overline{B}_2, \overline{C}_b)$.

6 Experimental Setups

We collect all 8,515 sentences containing WOEs from the HSK web site. We restrict the sentence length to 6~60 Chinese characters, filter out sentences with multiple error types, and remove duplicate sentences. There are 1,100 error sentences with WOEs for this study.

Considering the ratio of correct sentences, we use balanced datasets in the experiments. The most important reason is that the percentage of WOEs in the real world is relatively small. There are 2.83 million words in HSK. Assume a sentence contains 10 words on average. There are 0.283 million sentences in HSK. If we build an experimental dataset in this ratio, i.e., 8,525 vs. 283,000, the number of correct sentences will dominate the system's performance. Therefore, we build a balanced dataset to prevent this bias and to have a fair examination.

In the study of automatic error detection, collecting linguistic errors in the real world is indispensable, but time-consuming. It needs a lot of efforts to tag the error types and the possible corrections. Therefore, we plan to best utilize the error sentences collected in the HSK corpus to get more results for discussion. From the HSK corpus, we collect 4,400 correct sentences to generate four balanced datasets, each containing 1,100 error sentences and 1,100 correct sentences. These four datasets are called the HSK-HSK datasets.

In addition to using HSK correct sentences as positive sentences, we consider sentences by native Chinese writers as positive sentences to analyze which training set is useful to detect error sentences written by non-native language learners. We collect another 4,400 sentences from Chinese articles written by native Chinese writers, which talk about the same themes as those in HSK corpus, i.e., health, life, job application, classmate and friendship. The sentences from native learners are assumed to be correct. Those 4,400 web sentences are used to generate

another four balanced datasets. The resulting four datasets are named as the NAT-HSK datasets. In total, we have eight datasets in two groups for our experiments.

For each dataset, the corresponding 2,200 sentences are further randomly split into five folds. The experimental results are reported by averaging the 20 folds (4 datasets by 5 folds). In this way, we decrease the variations from the dataset generation, and make more reliable analyses about our approaches. We use RBF kernel and adopt grid search to determine the best SVM parameters in training set.

7 Experimental Results

In this section, we report the average accuracy of HSK-HSK datasets and NAT-HSK datasets. Paired Student's *t*-test is used for significant test. The null hypothesis is reject if $|t| > t_{19,\ 0.975} = 2.093$, where there are 19 degrees of freedom and the probability that *t* is less than 2.093 is 0.975. Because the WOE problem is formulated as a binary classification problem and we use balanced datasets, we adopt accuracy as our evaluation metric, which is more adequate than precision and recall. In addition, we ignore the analysis of confusion matrix in the results because we find that there are no big differences between the results of positive and negative samples.

7.1 Basic Features

Firstly, we want to know the performance of the syntactic features. Table 3 shows the experimental results. B_2, B_3, and B_4 denote feature vectors of POS *n*-grams (*n*=2, 3, 4) and B_p denotes feature vector of dependency relations. (B_2,B_3,B_4,B_p) denotes the combined syntactic feature vectors of POS n-grams (n=2, 3, 4) and parsing feature B_p. The resulting model is named model B.

Features	HSK-HSK		NAT-HSK	
	accuracy (%)	stdev.	accuracy (%)	stdev.
B_2	62.63	2.12	67.61	2.17
B_3	60.81	2.14	65.47	2.24
B_4	57.83	1.98	61.28	2.18
B_p	61.86	2.21	63.17	2.57
B=(B_2,B_3,B_4,B_p)	**63.89**	2.17	**69.61**	2.04

TABLE 3 –Performance of basic features

The accuracy of using POS 2-gram features (i.e., B_2) is better than that of using other individual features significantly except the parsing feature (i.e., B_p) in HSK-HSK datasets. It shows that POS 2-grams are very useful in detecting Chinese word ordering errors.

Another interesting finding is the usefulness of parser. The accuracy of using POS 2-gram features (i.e., B_2) is better than that of using parsing features (i.e., B_p) in NAT-HSK datasets significantly. Its accuracy reaches 67.61%. That reflects the effect of training with native Chinese writers' sentences. Intuitively, the sentences of native Chinese learners are more fluent than those of foreign learners, so that it is easier to detect wrong sentences with POS 2-grams learned from the correct sentences.

Because we use balanced datasets, the trivial baseline is 50% accuracy. Using all the above features outperform the baseline significantly. The combined model $B=(B_2,B_3,B_4,B_p)$ using all syntactic features outperforms the features using only one syntactic feature in both HSK-HSK and NAT-HSK datasets significantly.

7.2 Web Corpus Features

In Chinese, characters are segmented into a sequence of words and then the word frequencies are counted. In the next set of experiments, we aim to know how word segmentation influences the results of error detection. Two issues, including the performance difference without/with segmentation, and the effects of different segmentation systems, are considered.

When calculating $p(w_i,w_{i+1})$ in Equation (2), we use different approaches to get the frequencies of word pairs (w_i,w_{i+1}). The first approach ignores the word boundary between w_i and w_{i+1}, regards them as a single string, and uses string matching to get its count. This is a without-segmentation approach $C_{\bullet W}$, where the subscript character $\bullet$ denotes a reference corpus, i.e., GC5g or CP5g in our experiments. The second approach regards w_i and w_{i+1} as two separate words, and uses exact word matching to count their co-occurrences. This is a with-segmentation approach $C_{\bullet S}$. As a result, when we compute frequencies of word pairs, there are six combinations C_{gS}, C_{gW}, C_{bS}, C_{bW}, C_{p2S}, and C_{p2W}, where the subscript g means the Google Chinese Web 5-gram corpus (GC5g) is adopted, the subscript b means the Chinese Web POS 5-gram corpus (CP5g) is adopted, and the subscript p2 means Equation (4) and CP5g are adopted. We compare the effects of segmentation and the use of different kinds of corpora to see if they are complementary. Table 4 shows the results.

Features	HSK-HSK		NAT-HSK	
	accuracy (%)	stdev.	accuracy (%)	stdev.
C_{gW}	50.84	2.26	63.17	1.54
C_{gS}	52.64	2.01	65.01	2.23
$\overline{\overline{C}}_g=(C_{gW}, C_{gS})$	52.59	2.21	64.77	2.29
C_{bW}	50.90	2.94	63.90	1.81
C_{bS}	51.95	2.67	65.09	1.69
$\overline{\overline{C}}_b=(C_{bW}, C_{bS})$	56.99	1.98	65.93	2.01
C_{p1}	49.33	2.59	53.33	2.16
C_{p2W}	53.19	2.06	60.32	2.19
C_{p2S}	52.10	2.17	59.68	1.53
$\overline{\overline{C}}_p=(C_{p1},C_{p2W},C_{p2S})$	57.01	1.35	61.27	1.68
$(\overline{\overline{C}}_g, \overline{\overline{C}}_b, \overline{\overline{C}}_p)$	59.35	2.48	66.69	1.92
$B=(B_2,B_3,B_4,B_p)$	63.89	2.17	69.61	2.04
$(B, \overline{\overline{C}}_g)$	63.70	1.84	69.82	1.66
$(B, \overline{\overline{C}}_b)$	64.11	2.13	69.85	1.99
$(B, \overline{\overline{C}}_p)$	63.80	2.25	70.23	2.15
$(B, \overline{\overline{C}}_g, \overline{\overline{C}}_b, \overline{\overline{C}}_p)$	**64.34**	2.35	**71.18**	2.29

TABLE 4 –Performance of web corpus features

Comparing the uses of the individual features, i.e., C_{gS}, C_{gW}, C_{bS}, C_{bW}, C_{p1}, C_{p2S}, and C_{p2W}, on the HSK-HSK and NAT-HSK datasets, we find the performance of using NAT-HSK datasets is better than that of using HSK-HSK datasets. That meets our expectation because the correct sentences in NAT-HSK datasets are selected from native Chinese writers' web pages. Thus, we fix the datasets to NAT-HSK in the next discussion.

Because two different segmentation systems are applied in the development of the two reference corpora, we compare C_{gS} and C_{bS} features to see if the segmentation results have different effects on the performance. Their performance, 65.01% vs. 65.09%, does not have a significant difference in NAT-HSK datasets. Thus, we conclude that the two different segmentation systems do not have different effects.

In addition, we compare the performance of C_{gW} (63.17%) vs. C_{gS} (65.01%), and C_{bW} (63.90%) vs. C_{bS} (65.09%) to see if segmentation is necessary. The accuracies using word-based matching (i.e., with segmentation $C_{\cdot S}$) are better than those using string-based matching (i.e., without segmentation $C_{\cdot W}$) significantly. This phenomenon also holds in HSK-HSK datasets. Thus, we conclude that using segmentation systems result in better performance.

Next we analyze the usefulness of individual POS features. Table 4 shows that the performance of using C_{p1} features is 49.33% and 53.33% in HSK-HSK and NAT-HSK datasets, respectively. It seems that POS unigram information does not help. Comparatively, the performance of C_{p2W} shows that POS bigrams are more useful than POS unigrams, but POS-based features (C_{p2W}) cannot compete with word-based features (C_{gW} and C_{bW}). We analyze the eight datasets and find that only 7.9% of the tuple $(t_l,t_{l+1},w_l,w_{l+1})$ in Equation (4) can be found in the CP5g corpus, while 77.9% and 78.6% of the word pair (w_l,w_{l+1}) can be found in the GC5g corpus and the CP5g corpus, respectively. Therefore, we conjecture that with a larger dataset, POS N-grams will make a significant difference in this task.

Finally, we examine the complementary effects of the individual features by different integration. Integrating the individual web corpus features $\overline{\overline{C}}_g$, $\overline{\overline{C}}_b$, and $\overline{\overline{C}}_p$ with all syntactic features B outperforms using only the syntactic features B. The accuracy (71.18%) of the approach integrating all web corpus features ($\overline{\overline{C}}_g$, $\overline{\overline{C}}_b$, $\overline{\overline{C}}_p$) with all syntactic features B is significantly better than the accuracy (69.61%) of the approach using B. This confirms that the web corpus features are useful in WOEs detection.

7.3 Perturbation

Table 5 shows parts of perturbation experiments. The system $B=(B_2,B_3,B_4,B_p)$ significantly outperforms all the perturbation features. It seems that the perturbation does not help in all variations. One possible explanation is that we perturb words, but the word ordering errors usually involve many words such as the example "我 留学 在 贵国" (I am studying in your country) in Table 1. The word-based swapping "在 留学" (in studying) cannot capture the PP "在 贵国" (in your country). A phrase-based swapping should be adopted.

Features	HSK-HSK		NAT-HSK	
	accuracy (%)	stdev.	accuracy (%)	stdev.
B_2	62.63	2.12	67.61	2.17
pB_2	60.83	3.03	64.49	1.81

TABLE 5 –Performance of perturbations

ΔpB_2	51.14	3.30	53.28	2.08
C_b	56.99	1.98	65.93	2.01
pC_b	57.25	2.29	65.32	1.54
ΔpC_b	57.23	2.14	65.57	1.67
$B=(B_2,B_3,B_4,B_p)$	**63.89**	2.17	**69.61**	2.04
$(pB_2, \Delta pB_2, pC_b, \Delta pC_b)$	62.34	2.91	66.33	1.99

TABLE 5 –Performance of perturbations (*continued*)

7.4 Combined Features

We combine all features, including syntactic features (B), web corpus features (C_g, C_b, and C_p), and perturbation features, and report their performance in Table 6. We can see that using all features results in the highest accuracy (71.64%) in NAT-HSK datasets. It is better than the accuracy (69.61%) of the system B significantly. Also, using all features also results in the highest accuracy (64.81%) in HSK-HSK datasets.

Features	HSK-HSK		NAT-HSK	
	accuracy (%)	stdev.	accuracy (%)	stdev.
All features	**64.81**	3.45	**71.64**	1.85
$(B, \overline{C}_g, \overline{C}_b, \overline{C}_p)$	64.34	2.35	71.18	2.29
$B=(B_2,B_3,B_4,B_p)$	63.89	2.17	69.61	2.04

TABLE 6 –Performance of combining features

7.5 Analyses

At first, we want to know the relationship between sentence length and accuracy. We apply the models trained by using all features and one of HSK-HSK datasets, as well as one of NAT-HSK datasets to gather the statistics. These two datasets are denoted as HSK-HSK-1 and NAT-HSK-1, respectively. Figure 1 shows the results. We can see that the average accuracies in both two sets are high when the sentence length is in range [10, 19] characters. Moreover, the performance of using NAT-HSK-1 dataset is better than that of using HSK-HSK-1 dataset in all various lengths. That confirms our conclusion in Section 7.1.

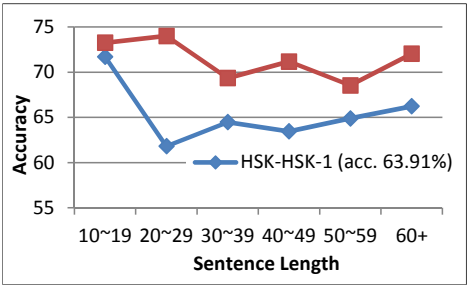


FIGURE 1 – Accuracy analysis by sentence length

Next, we want to know which category of errors is more difficult to detect. Among 200 sentences in Table 1, 197 WOE sentences are used in our experiments. We use HSK-HSK-1 and NAT-HSK-1 to gather the performance in different categories. The overall accuracy of these 197 sentences in HSK-HSK-1 dataset is 74.11% and is 70.56% in NAT-HSK-1 dataset. The detail result is listed in Table 7. The overall average accuracy of HSK-HSK-1 is 63.91% (in Figure 1 and Table 7), which is smaller than the average accuracy 74.11% of 197 sentences. Therefore, the 197 sentences may not be a good representation of HSK-HSK-1 dataset. Comparatively, the overall accuracy of 197 sentences in NAT-HSK-1 dataset is 70.56%, which is near 71.22%. Thus, these 197 sentences may be representative in NAT-HSK-1 dataset. We use the last column to analyze the difficulties of error detection in different categories. Of the former 3 major categories of errors, the accuracy of detecting PP position errors is higher than the overall accuracy. This indicates that the fluent sentences written by native Chinese learners are especially useful for detecting PP position errors in the sentences by non-native learners.

Category	#sentences	HSK-HSK-1	NAT-HSK-1
		accuracy (%)	accuracy (%)
Adverb ordering error	70	77.14	67.14
S/V/O ordering error	65	70.77	67.69
PP position error	41	73.17	**80.49**
Prenominal adjective error	10	**80.00**	80.00
Others	11	72.73	63.64
accuracy (total 197 sentences)		74.11	70.56
accuracy (total 2,200 sentences)		63.91	71.22

TABLE 7 –Performance analysis by category

Finally, we want to know the error detection performance with respect to the students' nationality. We collect the nationality information of the 197 sentences from the HSK website. The results are shown in Table 8. We can find that Korea and Japanese students are the major Chinese learners in the 197 WOE sentences. In HSK-HSK-1 dataset, 77.78% of WOEs from Japanese students can be detected. But this trend is not the same in NAT-HSK-1 dataset. In NAT-HSK-1 dataset, nationality does not make large difference on the error detection performance.

Nationality	#sentences	HSK-HSK-1	NAT-HSK-1
		accuracy (%)	accuracy (%)
Korea	73	69.86	**72.60**
Japan	72	**77.78**	68.06
Other countries	52	75.00	71.15
accuracy (total 197 sentences)		74.11	70.56
accuracy (total 2,200 sentences)		63.91	71.22

TABLE 8 –Performance analysis by nationality

Conclusion

In this paper, we deal with the detection of Chinese word ordering errors for learning Chinese as a foreign language from the application, language, resource and technology aspects. Different categories of word ordering errors in the sentences written by non-native language learners are

analyzed. The experiments show that syntactical features, web corpus features and perturbation features are all useful in detecting word ordering errors. The system using all the features from different sources has the best accuracy, 71.64%, when the native writers' sentences are selected as positive sentences. The proposed system is significantly better than the baseline systems.

In Chinese, word segmentation is inherent in many applications. Even though word ordering errors in Chinese sentences may result in incorrect segmentations, the experiments show that the word-based approach (i.e., with segmentation) is better than the character-based approach (i.e., without segmentation) irrespective of the use of different segmentation systems and reference corpora. On the other hand, although the Chinese Web POS 5-gram corpus is smaller than the Google Chinese Web 5-gram corpus, the experiments show that they have similar performances on the WOE detection. With this important finding, we plan to use linguistic information in the CP5g corpus for other Chinese NLP tasks such as sentiment analysis.

The use of web scale linguistic information provides a solid basis of further error detection and correction. In the future, extending this task to detect other types of errors, using web-scale linguistic corpus to identify the potential error positions and proposing the correct sentences are the works to be investigated.

Acknowledgments

This research was partially supported by Excellent Research Projects of National Taiwan University under contract 101R890858.

References

Bergsma, S., Lin, D. and Goebel, R. (2009). Web-Scale N-gram Models for Lexical Disambiguation. In *the 21st International Joint Conference on Artificial Intelligence*, pages 1507–1512, Pasadena, California, USA.

Brants, T. and Franz, A. (2006). Web 1T 5-gram Version 1. Linguistic Data Consortium, Philadelphia.

Carl, J. (1998). Errors in Language Learning and Use: Exploring Error Analysis. Addison Wesley Longman.

Chang, C.-C. and Lin, C.-J. (2011). LIBSVM: A Library for Support Vector Machines. *ACM Transactions on Intelligent Systems and Technology*, 2(3), 27:1–27:27.

Chodorow, M. and Leacock, C. (2000). An Unsupervised Method for Detecting Grammatical Errors. In *the 1st North American chapter of the Association for Computational Linguistics conference*, pages 140–147, Seattle, Washington, USA.

Church, K. W. and Hanks, P. (1990). Word Association Norms, Mutual Information, and Lexicography. *Computational Linguistics*, 16(1), 22–29.

Gamon, M., Leacock, C., Brockett, C., Dolan, W. B., Gao, J., Belenko, D. and Klementiev, A. (2009). Using Statistical Techniques and Web Search to Correct ESL Errors. *CALICO Journal*, 26(3), 491–511.

Islam, A. and Inkpen, D. (2010). An Unsupervised Approach to Preposition Error Correction. In *the 6th IEEE International Conference on Natural Language Processing and Knowledge Engineering*, pages 1–4, Beijing, China.

Islam, A. and Inkpen, D. (2011). Correcting Different Types of Errors in Texts. In *the 24th Canadian Conference on Advances in Artificial Intelligence*, pages 192–203, St. John's, Canada.

Izumi, E., Uchimoto, K., Saiga, T., Supnithi, T. and Isahara, H. (2003). Automatic Error Detection in the Japanese Learners' English Spoken Data. In *the 41st Annual Meeting on Association for Computational Linguistics*, pages 145–148, Sapporo, Japan.

Leacock, C., Chodorow, M., Gamon, M. and Tetreault, J. (2010). Automated Grammatical Error Detection for Language Learners. Morgan and Claypool Publishers.

Lee, J., Zhou, M. and Liu, X. (2007). Detection of Non-Native Sentences Using Machine-Translated Training Data. In *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics*, pages 93–96, Rochester, NY.

Lin, D., Church, K., Ji, H., Sekine, S., Yarowsky, D., Bergsma, S., Patil, K., Pitler, E., Lathbury, R., Rao, V., Dalwani, K. and Narsale, S. (2009). Unsupervised Acquisition of Lexical Knowledge from N-grams: Final Report of the 2009 JHU CLSP Workshop.

Lin, D., Church, K., Ji, H., Sekine, S., Yarowsky, D., Bergsma, S., Patil, K., Pitler, E., Lathbury, R., Rao, V., Dalwani, K. and Narsale, S. (2010). New Tools for Web-Scale N-grams. In *the Seventh Conference on International Language Resources and Evaluation*, pages 2221–2227, Malta.

Linguistic Data Consortium (LDC). (2003). English Gigaword.

Liu, F., Yang, M. and Lin, D. (2010). Chinese Web 5-gram Version 1. Linguistic Data Consortium, Philadelphia.

Malouf, R. (2000). The Order of Prenominal Adjectives in Natural Language Generation. In *the 38th Annual Meeting on Association for Computational Linguistics*, pages 85–92, Hong Kong.

Toutanova, K., Klein, D., Manning, C. D. and Singer, Y. (2003). Feature-Rich Part-of-Speech Tagging with a Cyclic Dependency Network. In *the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology*, pages 173–180, Edmonton, Canada.

Tseng, H., Chang, P., Andrew, G., Jurafsky, D. and Manning, C. (2005). A Conditional Random Field Word Segmenter. In *Fourth SIGHAN Workshop on Chinese Language Processing pages*, pages 168–171, Jeju, Korea.

Wagner, J., Foster, J. and Genabith, J. V. (2007). A Comparative Evaluation of Deep and Shallow Approaches to the Automatic Detection of Common Grammatical Errors. In *the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 112–121, Prague, Czech Republic.

Yu, C.-H., Tang, Y. and Chen, H.-H. (2012). Development of a Web-Scale Chinese Word N-gram Corpus with Parts of Speech Information, In *the Eighth International Conference on Language Resources and Evaluation*, pages 320–324, Istanbul, Turkey.

Machine Translation by Modeling Predicate-Argument Structure Transformation

Feifei ZHAI, Jiajun ZHANG, Yu ZHOU and Chengqing ZONG
National Laboratory of Pattern Recognition, Institute of Automation,
Chinese Academy of Sciences, Beijing, China
{ffzhai, jjzhang, yzhou, cqzong}@nlpr.ia.ac.cn

ABSTRACT

Machine translation aims to generate a target sentence that is semantically equivalent to the source sentence. However, most of current statistical machine translation models do not model the semantics of sentences. In this paper, we propose a novel translation framework based on predicate-argument structure (PAS) for its capacity on grasping the semantics and skeleton structure of sentences. By using PAS, the framework effectively models both semantics of languages and global reordering for translation. In the framework, we divide the translation process into 3 steps: (1) *PAS acquisition*: perform semantic role labeling (SRL) on the input sentences to acquire source-side PASs; (2) *Transformation*: convert source-side PASs to their target counterparts by predicate-aware PAS transformation rules; (3) *Translation*: first translate the predicate and arguments of PAS and then adopt a CKY-style decoding algorithm to translate the entire PAS. Experimental results show that our PAS-based translation framework significantly improves the translation performance.

KEYWORDS: Predicate-argument structure; Semantic role labeling; PAS transformation; PAS-based translation

Proceedings of COLING 2012: Technical Papers, pages 3019–3036,
COLING 2012, Mumbai, December 2012.

1 Introduction

Statistical machine translation (SMT) has made significant progress from word-based models (Brown et al., 1993) to phrase-based models (Koehn et al., 2003; Och and Ney, 2004) and syntax-based models (Galley et al., 2006; Liu et al., 2006; Marcu et al., 2006) over the past decades. However, the existing SMT models are always criticized for not modeling the semantics of languages. Furthermore, reordering is always one of the most difficult and important research problems in SMT. However, although current translation models are much good at local reordering[1], most of them are weak to cope with global reordering[2]. The two weaknesses restrict current translation models a lot, which urges us to seek a new translation framework to model both the semantics of languages and global reordering.

Formally, predicate-argument structure (PAS) is a structure that depicts the relationship between a predicate and its associated arguments, and it always indicates the semantic frame and skeleton structure of a sentence. From the characteristics of PAS, we can see that it provides not only a good semantic representation for modeling semantics, but also a skeleton structure for global reordering. Moreover, Fung et al. (2006) and Wu and Fung (2009b) have shown that PASs of the both sides are more consistent with each other than syntax structures. Considering current syntax-based translation models are always impaired by cross-lingual structure divergence (Eisner, 2003; Zhang et al., 2010), PAS will be a better alternative for building translation models.

Therefore, in this paper, aiming at building a PAS-based translation framework, we propose a novel translation method based on PAS transformation. FIGURE 1 is an overview of our method. Specifically, we divide the entire translation process into 3 steps:

(1) *PAS acquisition*: perform semantic role labeling (SRL) on the input sentences to achieve their PASs, i.e., *source-side PASs*.

(2) *Transformation*: convert source-side PASs to target-side-like PASs by predicate-aware PAS transformation rules, which are extracted from the result of bilingual semantic role labeling (Zhuang and Zong, 2010b). Here, *target-side-like PAS* denotes a list of general non-terminals in target language order, where a non-terminal aligns to a source element. Henceforward, we use source elements to denote the predicate and arguments of source-side PAS (similarly for target elements).

(3) *Translation*: just as FIGURE 1 shows, this step is further divided into two parts: (a) *element translation* is to translate each source element respectively; (b) *translation by global reordering* is to combine the translation candidates of source elements to translate the entire PAS based on the target-side-like PAS.

This method performs translation based on the PASs of sentences. In the transformation step, we model the source-side PAS by PAS transformation rules and convert it to target-side-like PAS. This means that we transform the skeleton structure of source sentence into the skeleton structure of target language. Obviously, this transformation process relates both sides on the skeleton level and would be potential to handle the global reordering problem.

[1] In this paper, global reordering refers to perform reordering based on the entire sentence structure. The other reordering operations are actually all local ones, even for the long-distance reordering without considering the global sentence structure.

[2] Only syntax-based models have tried to model global reordering. However, it needs large translation rules to take the entire sentence structure into account. This requirement always leads to a severe sparsity problem for translation. Therefore, the global reordering problem is not well addressed in these models.

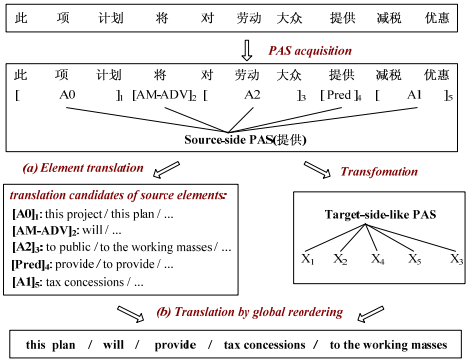


FIGURE 1 Three steps of our PAS-based translation framework: (1) *PAS acquisition*; (2) *Transformation*; (3) *Translation*. In the figure, the same subscript denotes the one-to-one alingment between source elements and non-terminals of target-side-like PAS.

Intuitively, when a human interpreter translates a sentence, he/she segments the sentence according to his/her understanding and translates each part respectively and then he/she translates the entire sentence by combining the partial translation of all parts. From this sense, the translation process of our PAS-based translation framework is similar to human translation to some extent. We believe that this work is a big step towards semantics-based machine translation.

Remainder of the paper is structured as follows. Section 2 elaborates the automatic process of extracting the predicate-aware PAS transformation rules. Section 3 details the translation process of our method. Section 4 describes how to decode the whole sentence with our method. In section 5, we evaluate the effectiveness of our method and in section 6, we introduce the related work. Finally, we end with the conclusion and perspectives.

2 PAS Transformation Rule Extraction

In this section, we introduce the method of bilingual semantic role labeling (SRL) and present how to extract PAS transformation rules based on the bilingual SRL result.

2.1 Bilingual Semantic Role Labeling

Bilingual SRL is to perform SRL on bitext simultaneously. In order to do this, (Zhuang and Zong, 2010b) proposed a method to infer bilingual semantic roles jointly. At first, they looked for aligned bilingual predicates and generated multiple monolingual SRL results by monolingual SRL systems. Then they adopted an integer linear programming method to find the best bilingual SRL result. They not only achieved the start-of-the-art monolingual SRL performance to date, but acquired the mapping between bilingual arguments. Thus, we follow their work to achieve bilingual SRL results for our training set. FIGURE 2(a) shows an example of bilingual SRL.

2.2 Rule Extraction

With the bilingual SRL result in FIGURE 2(a), we can easily generate an *exact transformation rule*, of which the left and right side is the PASs on the two sides, just as FIGURE 2(b) shows. Using the

rule, we can project the translation candidates of source elements to their aligned target elements and then translate the entire PAS by combining these candidates.

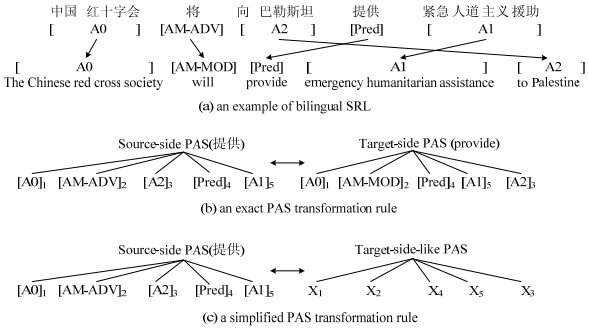


FIGURE 2 – An example of bilingual SRL and the corresponding PAS transformation rules: In (b) and (c), the same subscript at the source and target side denotes the aligned elements in PASs.

Obviously, semantic roles of target elements are not used in the above translation process[3]. Therefore, we can simplify the exact transformation rule by substituting target elements' semantic roles with general non-terminals. We call the achieved target-side PAS as *target-side-like PAS* and name the rule as *simplified transformation rule*, just like the rule in FIGURE 2(c). Basically, a simplified transformation rule $\boldsymbol{r}$ is a triple $< Pred, SP, TP >$:

- *Pred* is the specific source-side predicate where rule $\boldsymbol{r}$ is extracted.
- *SP* denotes the source-side PAS, which is a list of source elements in source language order.
- *TP* is the target-side-like PAS, i.e., a list of general non-terminals in target language order.

For example, the rule in FIGURE 2(c) is a tripe where *Pred* is Chinese verb "提供", *SP* is the source element list $< [A0]_1 [AM\text{-}ADV]_2 [A2]_3 [Pred]_4 [A1]_5 >$, and *TP* is the list of non-terminals $< X_1 X_2 X_4 X_5 X_3 >$. The same subscript in *SP* and *TP* refer to the one-to-one mapping between a source element and a target non-terminal. Obviously, the transformation rule can easily grasp the interrelation of bilingual PASs. Note that the target predicate "provide" in FIGURE 2(b) is ignored because its counterpart predicate "提供" will be translated by the element [Pred] in *SP*.

Virtually, in order to project the translation candidates of source elements to target-side-like PAS, we require that a source argument only aligns to a target argument. However, the result of bilingual SRL usually does not satisfy this requirement. There exist many unaligned source arguments, and sometimes a source argument might align to more than one target argument.

To resolve this problem, we refine the bilingual SRL result via word alignment. We focus on source arguments and refine the corresponding target arguments. For the unaligned source arguments, we look for their target spans via word alignment. If the source argument and its target span are consistent with word alignment[4], and its target span does not overlap with the

[3] Semantic roles of target elements can be used to evaluate the quality of translation candidates. Here we do not consider this point and we take it as our future work.

[4] Two spans are consistent with word alignment means that words in source span only align to words in target span via word alignment, and vice versa.

target span of other source arguments, we take the target span as a virtual target argument for rule extraction. Otherwise, we ignore the source argument.

Towards the source argument aligning to more than one target argument, we check the minimal continuous target span covering all its aligned target arguments. If the span does not overlap with other target arguments, we also take the span as a virtual argument for rule extraction. Otherwise, we discard the source argument. In addition, for the predicate whose multiple arguments align to one or more target arguments (many-to-one/many case), we do not extract rules from that predicate. According to our final statistics, only 6.9% of the aligned predicate pairs are discarded.

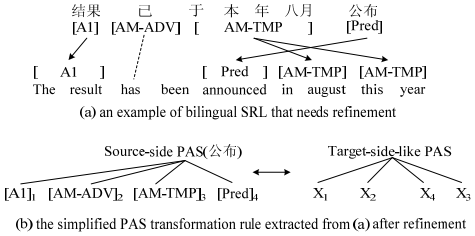


FIGURE 3 – An example for refining the bilingual SRL result.

For example, in FIGURE 3, although the source argument [AM-ADV] is unaligned, we align it to target word *"has"* via word alignment. For source argument [AM-TMP], the minimal span that covers the two target argument [AM-TMP]s does not overlap with other target arguments. We take that span as a big virtual argument for rule extraction. At last, we extract the simplified transformation rule in FIGURE 3(b).

Finally, the transformation rules are organized into a Trie structure. In order to store a rule, we use the rule's *Pred* and *SP* as the key, and *TP* as the value of Trie node. Henceforward, we utilize *TRTrie* to denote the Trie structure encoding all the transformation rules.

2.3 Rule Extension

Basically, some modifier arguments[5] are actually not necessary for the skeleton of sentences. For example, source argument [AM-TMP] in FIGURE 3(a) is a modifier. If we ignore it and its target counterpart, the remaining PAS is still reasonable. Therefore, we extend the PAS transformation rules based on this insight. For a specific PAS transformation rule, we traverse all its modifiers and discard each one in turn, and meanwhile, construct a simplified transformation rule with the remaining arguments of the PAS. For instance, if we ignore the source argument [AM-TMP] in FIGURE 3(a), we can get a simplified transformation rule where *Pred* is verb "公布", *SP* is the source element list $<[\text{A1}]_1\ [\text{AM-ADV}]_2\ [\text{Pred}]_3>$, and *TP* is $<\text{X}_1\ \text{X}_2\ \text{X}_3>$.

2.4 Rule Probabilities

To distinguish different transformation rules during decoding, we design two probabilities for each transformation rule: *predicate-conditioned rule probability* $p_{pred}(r)$ and *source-PAS-conditioned rule probability* $p_{sp}(r)$:

[5] The argument that utilizes AM as its prefix.

$$p_{pred}(r) = \frac{c(r)}{\sum_{r':Pred(r')=Pred(r)} c(r')}$$

$$p_{SP}(r) = \frac{c(TSP(r))}{\sum_{r':SP(r'))=SP(r)} c(TSP(r'))}$$

In the two formulas, $Pred(r)$ and $SP(r)$ denote *Pred* and *SP* of rule ***r*** respectively. $TSP(r)$ refers to the combination of rule ***r***'s *SP* and *TP*. $c(r)$ is the count of rule ***r*** (similarly for $c(TSP(r))$). The two probabilities will serve as features for decoding. Generally, the first feature is mainly used to evaluate which transformation rule is more possible for the specific source predicate. The second feature is used to evaluate which *TP* is more appropriate for the specific *SP*[6]. The two features indicate the distribution of bilingual PASs from two different angles, which will be helpful for the decoder to choose effective PAS transformation rules.

3 PAS-based Translation Framework

In the *PAS acquisition* step, we perform SRL on each test sentence with a monolingual SRL system. To alleviate the negative impact of SRL errors, we use multiple SRL results. We provide the monolingual SRL system with 3-best parse trees of Berkeley parser (Petrov and Klein, 2007), 1-best parse tree of Bikel parser (Bikel, 2004) and Stanford parser (Klein and Manning, 2003). FIGURE 4(a) shows an example of multiple SRL results. In the *transformation* step, we match the multiple SRL results with PAS transformation rules and convert them to target-side-like PASs. Then in the *translation* step, we decode the PAS based on these target-side-like PASs.

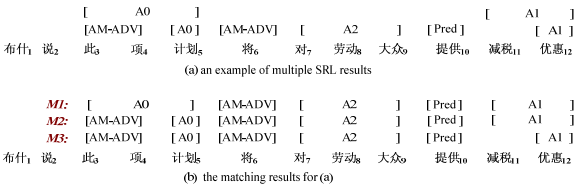


FIGURE 4 – Multiple SRL results and the final mathcing result of the example sentence.

3.1 PAS Transformation

In this section, we describe how to match the multiple SRL results with PAS transformation rules and transform them to target-side-like PASs. We design *Algorithm 1* to achieve our purpose. First, we look for the predicate in *TRTrie* and get the matching Trie node *P_N*. With this node, we continuously match the elements of PAS in order, and meanwhile, expand along *TRTrie*. Finally, we achieve all possible PASs that can match transformation rules. We only preserve the ones covering the largest number of source words or elements. We believe that only the PAS satisfying one of the two conditions is possible to stand for the real skeleton of a sentence and capture a good global reordering operation. For example, FIGURE 4(b) shows the matching result of FIGURE 4(a). The result M1 in FIGURE 4(b) covers the largest number of source words, and M3 carries the largest number of elements, and moreover, M2 satisfies the two conditions. After that,

[6] Actually, this feature should base on the entire rule ***r***, rather than TSP(r). However, this leads to severe data sparseness for rules. Therefore, we pursue the general rules and ignore the predicate here.

we can get target-side-like PASs from the transformation rules. Algorithm 1's complexity is exponential, but its speed is fast in practice because a predicate only carries very few arguments.

Algorithm 1: PAS Transformation Rule Matching

Input: *predicate* ***P****, a list* ***L*** *including all the source elements of* ***P****, and TRTrie*
Output: *a list* ***TPL*** *preserving all the achieved target-side-like PASs*
1: **function** Matching (*P*, *L*, *TRTrie*):
2: sort *L* first by the element's start position and then by its length from small to large
3: find *P* in *TRTrie* and get the Trie node *P_N*, **if** not find *P*, **return** ◁ match the predicate first
4: **for** *c_arg* in *L* **do**: ◁ consider all elements in turn
5: **for** *p_arg* that is before *c_arg* in *L* **do**: ◁ check all partial matching PASs
6: **if** *p_arg* does not overlap with *c_arg*:
7: **for** Trie node *t_n* in *p_arg* **do**:
8: **if** *c_arg* in descendents of *t_n*, **then** store that node into *c_arg* ◁ expand along *TRTrie*
9: find *c_arg* in descendents of ***P_N,*** **if** find, store that node into *c_arg* ◁ PAS might begin with any element
10: check all Trie nodes stored in L's elements, consider the rules covering the largest number of arguments or source words, and save *TP*s of these rules into *TPL* ◁ store the target-side-like PASs
11: return *TPL*

We use *matching score* to evaluate the matching PASs. For a PAS $A_{m1},...,A_{mn}$, such as <[A0][AM-ADV][A2][Pred][A1]> (the matching result M1 in FIGURE 4(b)), its matching score is:

$$p_{MS}(A_{m1},...,A_{mn}) = \frac{\prod_j p(A_{mj} \mid S, pred)}{\sum_{m'} \prod_j p(A_{m'j} \mid S, pred)}$$

where *S* and *pred* denote the test sentence and the predicate respectively. $p(A_{mj} \mid S, pred)$ denotes the probability that the SRL system assigns to element A_{mj}[7]. Additionally, the denominator sums the score of all matching PASs. This matching score will serve as a feature in the final decoder. It is mainly used to reward the good skeleton structure of sentences.

3.2 Gap Word Attachment

In a matching PAS, adjacent source elements might be separated by *gap words* in the sentence. For example, in the matching result M3 of FIGURE 4(b), [Pred] and [A1] are separated by a gap word "减税". For the PAS whose elements are separated by gap words, we cannot translate it only based on the target-side-like PAS because it is not continuous. Therefore, to address this problem, we attach the gap words to their neighbouring left or right elements via parse tree. We look for the lowest common ancestor nodes of the gap word and its left or right neighbouring elements respectively. We compare these two ancestor nodes and attach the gap word to the element whose corresponding ancestor node is lower in the parse tree. For example in FIGURE 5, the common ancestor node of word "减税" and [A1] is node $NP_{11,12}$, while it is node $VP_{10,12}$ for [Pred]. Hence, we attach word "减税" to [A1] and transform the PAS_1 to PAS_2 in FIGURE 5.

In practice, it is common that the neighboring left and right elements get the same ancestor node. This is because a father node can dominate many children nodes in parse trees. To address this problem, we employ the head binarization method (Wang et al., 2007) to binarize the parse trees.

[7] We average the five probabilities given by the 5 parse trees as this probability.

We make the final attachment decision by voting with the abovementioned five parse trees. After attachment, some PASs may be identical to each other, such as the matching result M2 and M3 of FIGURE 4(b). We only retain the one whose matching score is larger.

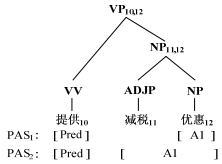


FIGURE 5 – An example of gap word attachment using parse tree.

3.3 PAS Translation

In the *translation* step, we translate each source element by a traditional translation method. Then we combine these candidates to translate the entire PAS based on the target-side-like PAS, just as FIGURE 1 shows. Intuitively, the combination can be operated directly by cube pruning (Chiang, 2007). However, since the source elements are translated independently and many source elements' spans are very short, numerous phrase translation rules are ignored during translation. This fact leads to a narrow decoding space and poor translation accuracy. To alleviate this problem, we design a CKY-style decoding algorithm for each target-side-like PAS.

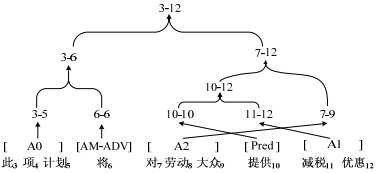


FIGURE 6 – An example of our CKY-style decoding algorithm for target-side-like PAS. In this example, only one path is generated for the final span 3-12. In practice, there can be many paths.

In the CKY-style decoding algorithm, we organize the source elements in target language order based on the target-side-like PAS. For example, in FIGURE 6, we use the rule in FIGURE 2(c) and create the span list [3,5], [6,6], [10,10], [11,12], [7,9]. Then we combine these spans in a bottom-up manner, just like traditional CKY algorithm works. The difference is that we only check all the possible combinations of small spans to form big spans, rather than checking all the split points of a big span. Moreover, if the adjacent spans are not adjacent at the source side, we do not combine them. For instance, in FIGURE 6, span [6,6] and [10,10] are adjacent in target order, but they are not adjacent at the source side. In addition, the translation candidates of newly generated spans, such as span [3,6], come from two parts: combining the translation candidates of its two sub-spans by cube pruning, or using phrase translation rules. These combined spans help to enlarge the search space a lot and yield a good translation performance.

Basically, only when the target-side-like PAS can be binarized, our decoding algorithm can be implemented. According to our statistics, almost all the target-side-like PASs can be binarized. We will detail the statistics in sub-section 5.2. If a target-side-like PAS cannot be binarized, we combine the partial translations of its elements by cube pruning straightforwardly.

4 Decoding with PAS-based Translation Framework

Formally, PAS represents the main structure of a sentence. However, sometimes the sentence cannot be fully covered by a PAS, especially when there are several predicates in the sentence. In order to translate the whole sentence, we design a decoding algorithm in terms of our PAS-based translation framework. The algorithm we adopted here follows the CKY-style framework.

In the decoder, we organize the search space of translation candidates into a hypergraph. For the span covered by PAS (named as *PAS span*), we use a multiple-branch hyperedge to connect that span to the PAS's elements. For the span not covered by PAS (named as *non-PAS span*), we consider all the binary segmentations of that span and use binary hyperedges to link them, just as FIGURE 7 shows. As a realistic example, FIGURE 8(a) shows a sentence and the PAS of its predicate "说(say)". The PASs of another predicate "提供(provide)" in the sentence are shown in FIGURE 4(b). The final decoding hypergraph is shown in FIGURE 8(b).

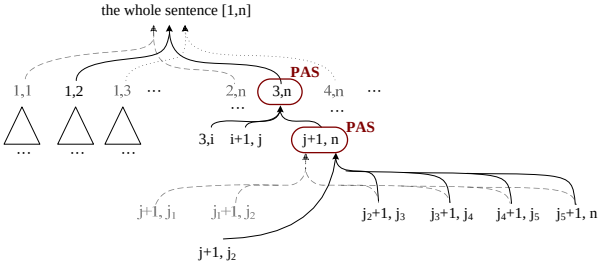


FIGURE 7 – An illustration of the decoding hypergraph. In the FIGURE, *n* refers to the length of sentence. Span [3,n] and [j+1,n] denote PAS spans and their descendent spans are all spans of elements in PAS.

After the hypergraph is constructed, we fill the spans with translation candidates in a bottom-up manner. When we encounter a PAS span, the algorithm described in sub-section 3.3 is used. Otherwise, the traditional translation method is utilized. Obviously, any CKY-based translation method can be used to generate translation candidates, such as BTG translation model and hierarchical phrase-based translation model. In this process, PAS span and non-PAS span are used equally for translating bigger spans. This is because bad PASs might harm the translation accuracy and the competition of PAS spans and non-PAS spans will help to choose good PASs.

For a specific span, we distinguish its translation candidates from different PASs by the two rule probabilities in sub-section 2.4 and the matching score in sub-section 3.1. These probabilities and scores are served as the PAS features for decoding. Their weights are tuned together with other features, such as language model. We call this translation system as *PAS transformation system*.

In the decoder, we can see that the translation candidates of PAS span are generated only by PAS transformation rules, while the traditional translation method also has its own way to translate the same PAS span. We believe that they complement each other because they perform translation from different angles. Thus, to capture this complementation, for the PAS span in the decoding hypergraph, we can use both our PAS-based translation method and the traditional translation method. This leads to a combination system which we call *PAS combination system*.

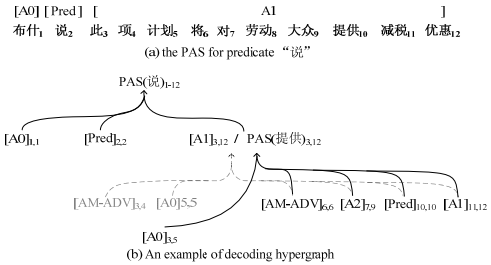


FIGURE 8 – An illustration of the decoding hypergraph. In the FIGURE, the PASs for predicate “提供 (provide)” are the result M1 and M2 in FIGURE 4(b). We omit the non-PAS spans here.

5 Experiment

5.1 Experimental Setup

The experiment is conducted on Chinese-to-English translation. The training data includes 260K bilingual sentence pairs[8]. To guarantee the accuracy of bilingual SRL, the length of each sentence is among 10 and 30 words. We use this data for both bilingual SRL and training the translation system. We first run GIZA++ and employ the *intersection* and *grow-diag-final-and (gdfa)* strategy respectively to produce symmetric word alignments. Then we use the intersection alignment to find the aligned predicates and adopt Zhuang and Zong (2010b)’s method to do bilingual SRL. After that, we refine the result in terms of the gdfa alignment and extract PAS transformation rules as described in section 2.

For machine translation, we train a 5-gram language model with the Xinhua portion of English Gigaword corpus and target part of training data. The development set and test set are the NIST evaluation test data (from 2003 to 2005). To get accurate SRL results, we also only extract sentences whose lengths are among 10 and 30 words. As a result, 595 sentences from NIST MT03 serve as the development set. 1,786 sentences from NIST MT04 and MT05 compose the test set. We perform SRL for the two sets by Zhuang and Zong (2010b)’s method. The translation quality is evaluated by case-insensitive BLEU-4 with shortest length penalty. The statistical significance test is performed by the re-sampling approach (Koehn, 2004). We employ our in-house BTG system used in (Zhang and Zong, 2009) to serve as our baseline translation method. We use PAS(BTG) to denote the PAS transformation system and PAS+BTG to represent the PAS combination system.

5.2 PAS Transformation Rules

In the training data, we acquire 226,968 aligned predicate pairs. From these predicate pairs, we extract 62,597 different simplified PAS transformation rules and then we extend them to 92,278 ones. Among the rules, 99.55% of their TPs can be binarized. Therefore, our decoding algorithm in sub-section 3.3 can be used in almost all cases. To detail our PAS transformation rules, we give the top 5 monotone rules and reordering rules respectively in TABLE 1.

[8] It is extracted from the LDC corpus. The LDC category number : LDC2000T50, LDC2002E18, LDC2003E07, LDC2004T07, LDC2005T06, LDC2002L27, LDC2005T10 and LDC2005T34.

Top 5 monotone rules			Top 5 reordering rules		
Pred	SP	TP	Pred	SP	TP
说(say)	$[A0]_1\ [Pred]_2\ [A1]_3$	$X_1\ X_2\ X_3$	提供(provide)	$[A0]_1\ [A2]_2\ [Pred]_3\ [A1]_4$	$X_1\ X_3\ X_4\ X_2$
认为(think)	$[A0]_1\ [Pred]_2\ [A1]_3$	$X_1\ X_2\ X_3$	支持(support)	$[A0]_1\ [AM\text{-}ADV]_2\ [Pred]_3\ [A1]_4$	$X_2\ X_1\ X_3\ X_4$
希望(hope)	$[A0]_1\ [Pred]_2\ [A1]_3$	$X_1\ X_2\ X_3$	说(say)	$[A0]_1\ [Pred]_2\ [A1]_3$	$X_3\ X_1\ X_2$
想(think)	$[A0]_1\ [Pred]_2\ [A1]_3$	$X_1\ X_2\ X_3$	表示(express)	$[A0]_1\ [A3]_2\ [Pred]_3\ [A1]_4$	$X_1\ X_3\ X_4\ X_2$
有(have)	$[A0]_1\ [Pred]_2\ [A1]_3$	$X_1\ X_2\ X_3$	举行(hold)	$[A1]_1\ [AM\text{-}LOC]_2\ [Pred]_3$	$X_1\ X_3\ X_2$

TABLE 1 – Top 5 monotone rules and reordering rules. The counts of monotone rules range from 5,101 to 1,745, and the counts of reordering rules range from 339 to 157.

Let us investigate the reordering rules first. The transformation rule for Chinese verb "提供(provide)" moves its argument [A2] behind [Pred] and [A1]. In general, [A2] is usually a prepositional phrase, which begins with a prepositional word, such as "为(for)" or "向(to)". This is reasonable because we always move the prepositional phrase behind verb phrase during Chinese-to-English translation, just as FIGURE 2(a) shows. From the transformation rules, we can see that we reorder the arguments based on the entire PAS. This demonstrates that our PAS-based translation method is good at global reordering.

For the monotone rules, we can see that all top 5 rules focus on [A0], [Pred] and [A1]. This fact demonstrates that Chinese and English are mostly Subject-Verb-Object (SVO) languages. Therefore, during Chinese-to-English translation, we can maintain the main skeleton structure of sentences according to the monotone rules.

5.3 Translation Result

TABLE 2 illustrates the final translation results of our experiments. As we can see, our in-house BTG system outperforms Moses (Koehn et al., 2007) by 0.33 BLEU points, indicating that our BTG system is a strong baseline system. Moreover, from TABLE 2, we can see that system PAS(BTG) only improves the baseline BTG system slightly, by 0.38 BLEU points. However, the PAS+BTG system significantly outperforms the baseline BTG system by 1.14 BLEU points. This comparison means that the PAS can better play its role by combining with BTG model. We will conduct a deep analysis on these results in the next sub-section.

		n-gram precisions			
System	Test Set	1	2	3	4
Moses	32.42	74.91	41.86	24.4	14.43
BTG	32.75	74.39	41.91	24.75	14.91
PAS(BTG)	33.13	75.13	42.55	25.10	15.02
PAS+BTG	**33.89***	74.98	43.17	25.91	15.72

TABLE 2 – Result of BTG system and our PAS-based translation method. The "*" denotes that the result is significantly better than BTG ($p<0.01$).

5.4 Analysis and Discussion

According to our statistics, in the total 1,786 test sentences, there are 1,747 ones have involved in matching PAS transformation rules. However, only 386 sentences in system PAS(BTG) but 1,017 sentences in system PAS+BTG have utilized PASs to generate final translations. Why they have such great difference in the two systems? After analysis, there are two main reasons.

On one hand, decoding space is narrowed and limited by the rigid spans under the PAS-based framework. As we described in sub-section 3.3, we use a CKY-style algorithm to enlarge the decoding space. However, even so, a lot of spans are still ignored during decoding. Moreover, the predetermined spans of arguments also restrict the usage of phrase translation rules.

On the other hand, the accuracy of SRL is not high. To our best knowledge, the F-score of current monolingual Chinese SRL system is only about 80% on the Treebank data. Moreover, this evaluation focuses on arguments, rather than the entire PASs. We can imagine that it would reduce greatly on the non-well-formed training and test data. In addition, according to our statistics, there are 26,809 different matching PASs in the test set in total, in which 16,489 ones (61.5% of all) have a father PAS or child PAS. This means such PAS is an argument of a bigger PAS or carries an argument which is actually a smaller PAS, just as FIGURE 8 shows. This hierarchical structure magnifies the negative impact of bad PASs in system PAS(BTG). Many accurate PASs are thus ignored because of its bad father PAS or child PAS.

Due to the narrow decoding space and bad PASs, the comprehensive translation score of PASs' translation candidates would be too low to be utilized in system PAS(BTG). Therefore, numerous PASs are bypassed by the decoder and only a slight improvement is achieved by system PAS(BTG). To address this problem, we propose system PAS+BTG. It not only combines the decoding space of our PAS-based translation framework and BTG translation model, but also breaks up the close connection between father PAS and child PAS by introducing BTG model's translation candidates for PASs. At last, it achieves significant improvement over BTG system and more PASs in 1,017 sentences are utilized in the system.

	# PAS-Span-Covered-Rate (named as cover-rate)			
PAS(BTG)	[0,50%)	[50%,100%)	100%	total
	181	65	225	471
	# PAS-Span-Covered-Rate (named as cover-rate)			
PAS+BTG	[0,50%)	[50%,100%)	100%	total
	613	775	125	1613

TABLE 3 – Statistics about PAS spans used for generating the final best translations. In the TABLE, for example, column 2 of system PAS(BTG) denotes that 65 PASs covering 50%~100% words of source sentences are utilized in system PAS(BTG).

To verify our above analysis, we further give TABLE 3. As we can see, comparing with PAS(BTG), much more PASs are used in PAS+BTG (471 vs 1613). Moreover, the number of PASs in PAS(BTG) reduces when the *cover-rate* increases[9], while the number for PAS+BTG grows. Just as we discussed above, this is because the big PAS in PAS(BTG) usually depends not only on itself, but also on its child PAS. Once the big PAS carries a bad child PAS, its translation would be also bad due to this child PAS. Therefore, the number of big PASs used in PAS(BTG) reduces. In contrast, the child PAS in PAS+BTG is only a choice but not essential for translating its father PAS. Hence, the number of big PASs used in PAS+BTG increases.

From TABLE 3, we can also see that most of the PASs cover more than 50% words of source sentences. We call these PASs as *sen-wide PAS*. In system PAS+BTG, the number of *sen-wide*

[9] There is an exception when the *cover-rate* is 100% in system PAS(BTG). This is because the 225 test sentences are fully covered by PASs. In system PAS(BTG), the translation of these sentences must be generated by the PAS spans whose *cover-rate* are 100%. Obviously, this is a rigid constraint. We relax this constraint in PAS+BTG system to ignore the bad PASs and 125 ones are kept for the final translation.

PASs is 900 (i.e., 775+125 in TABLE 3) and the number for system PAS(BTG) is 290 (i.e., 225+65 in TABLE 3). Each of these PASs belongs to one individual sentence because they all cover more than 50% words of the sentences. Consequently, 88.5% (900/1,017) sentences in PAS+BTG system and 75% (290/386) sentences in PAS(BTG) system have utilized these *sen-wide PASs*, by which the skeleton structure of sentences are well modeled for translation. Hence, we can conclude that our PAS-based translation method performs global reordering based on these *sen-wide PASs* and achieves improvements over the baseline BTG system.

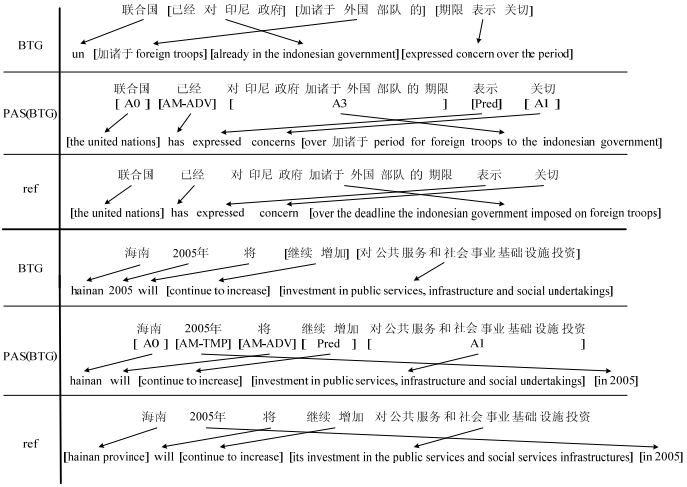


TABLE 4 – Two translation examples of BTG system, PAS(BTG) system, and reference.

We further give two translation examples in TABLE 4 to specially show the effectiveness of our PAS-based translation method. For the first example, BTG system chooses a wrong manner to segment the big prepositional phrase "对 印尼 政府 加诺于 外国 部队 的 期限" into 3 parts. This is because BTG system only tries to get a translation with an average distribution of phrase segmentation. Moreover, since its translation model does not consider any information of sentence structure, it wrongly segments the test sentence and produces a bad translation. Conversely, our PAS(BTG) system segments the sentence based on its PAS. Since a correct PAS denotes the skeleton structure of the sentence, it performs both reasonable sentence segmentation and better global phrase reordering for translation. Furthermore, in the second example, our PAS-based method successfully recognizes the [AM-TMP] argument "2005年" and move it to the end of sentence. However, the BTG system only performs translation without any reordering.

6 Related Work

Previous work utilizing PAS in SMT can be roughly categorized into three directions.

One direction is to do pre-processing or post-processing. Komachi and Matsumoto (2006) and Wu et al. (2011) used PAS-based heuristic rules and automatic rules respectively to pre-order the

input sentences. Wu and Fung (2009b) performed SRL on the outputs of phrase-based system Moses and then reordered the achieved semantic roles to match the roles of input sentences.

Some other works tried to design proper PAS-based features and integrate them into decoder. Liu and Gildea (2010) projected source-side PASs to target side via word alignment and designed a "Semantic Role Re-ordering" feature and a "Deleted Roles" feature for tree-to-string model. Xiong et al. (2012) adopted semantic features to translate verbal predicates and predict the relative position between predicates and arguments.

Some other works focused on utilizing semantic roles to refine the non-terminals of syntax-based translation model. Liu and Gildea (2008) substituted the syntactic labels with semantic roles or combined them together for a tree-to-string model. Aziz et al., (2011) used semantic roles and base-phrase tags to create shallow semantic trees. Gao and Vogel (2011) used target side semantic roles to create SRL-aware non-terminals for hierarchical phrase-based model.

Our work is different from the existing work in the following aspects: (1) we induce PAS transformation rules to model the interrelation between source-side PAS and its target counterpart; (2) we utilize multiple SRL results to alleviate the negative impact of bad PASs; (3) we design a CKY algorithm to translate the entire PAS according to the target-side-like PAS. The algorithm can be easily integrated with any CKY-based decoder to generate better translation hypotheses.

Conclusion and Perspectives

In this paper, we focus on building a PAS-based translation framework for modeling semantic structures in translation model. We first extract PAS transformation rules to model the intrinsic connection between source-side and target-side PASs. Then we perform machine translation in 3 steps: *PAS acquisition*, *transformation* and *translation*. Experimental results demonstrate that our PAS-based translation method improves the translation performance significantly.

Our method improves the translation performance in the following aspects: (1) take advantage of PAS, which keeps consistency well across languages; (2) use PAS transformation rules to perform global reordering in a skeleton scenario; (3) design reasonable strategies to exert the merit of PAS to segment sentences for translation; (4) the PAS-based translation framework can be easily integrated with any CKY-based translation models to generate better translations. In all, the translation process of our PAS-based translation method is similar to human translation to a great extent and it still has much room to improve with the upgrading of SRL performance. We believe it would be a big step towards semantics-based translation model.

In the next step, we will conduct further experiments on other language pairs to demonstrate the effectiveness of our PAS translation method, especially the translation between an SVO language and an SOV language. In addition, we also will utilize the target-side semantic roles to evaluate the quality of translation candidates and the structural integrity of translations.

Acknowledgments

The research work has been funded by the Natural Science Foundation of China under Grant No. 6097 5053 and the Hi-Tech Research and Development Program ("863" Program) of China under Grant No. 2012AA011102 and 2011AA01A207. We would also like to thank Tao Zhuang for the help in generating bilingual SRL results, and the anonymous reviewers for their valuable comments.

References

Aziz, W., Rios, M., and Specia, L. (2011). Shallow semantic trees for smt. In *Proceedings of the Sixth Workshop on Statistical Machine Translation*, WMT '11, pages 316–322, Stroudsburg, PA, USA. Association for Computational Linguistics.

Bikel, D. (2004). Intricacies of Collins parsing model. Computational Linguistics, 30(4):480-511.

Brown, P. F., Pietra, V. J. D., Pietra, S. A. D., and Mercer, R. L. (1993). The mathematics of statistical machine translation: parameter estimation. *Comput. Linguist.*, 19(2):263–311.

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

Eisner, J. (2003). Learning non-isomorphic tree mappings for machine translation. In *The Companion Volume to the Proceedings of 41st Annual Meeting of the Association for Computational Linguistics*, pages 205–208, Sapporo, Japan. Association for Computational Linguistics.

Fung, P., Wu, Z., Yang, Y. and Wu, D. (2006). Automatic learning of chinese english semantic structure mapping. In *IEEE/ACL 2006 Workshop on Spoken Language Technology (SLT 2006)*, Aruba, December.

Fung, P., Wu, Z., Yang, Y. and Wu, D. (2007). Learning bilingual semantic frames: shallow semantic sarsing vs. semantic sole projection. In *Proceedings of the 11th Conference on Theoretical and Methodological Issues in Machine Translation*, pages 75-84.

Galley, M., Graehl, J., Knight, K., Marcu, D., DeNeefe, S., Wang, W., and Thayer, I. (2006). Scalable inference and training of context-rich syntactic translation models. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 961–968, Sydney, Australia. Association for Computational Linguistics.

Gao, Q. and Vogel, S. (2011). Utilizing target-side semantic role labels to assist hierarchical phrase-based machine translation. In *Proceedings of the FifthWorkshop on Syntax, Semantics and Structure in Statistical Translation*, SSST-5, pages 107–115, Stroudsburg, PA, USA. Association for Computational Linguistics.

Klein, D. and Manning, C. D. (2003). Accurate unlexicalized parsing. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, pages 423–430, Sapporo, Japan. Association for Computational Linguistics.

Koehn, P., Och, F. J. and Marcu, D.. (2003). Statistical phrase-based translation. In *Proceedings of the 2003 Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics*, pages 58–54, Edmonton, Canada, May-June.

Koehn, P. (2004). Statistical significance tests for machine translation evaluation. In Lin, D. and Wu, D., editors, *Proceedings of EMNLP 2004*, pages 388–395, Barcelona, Spain. Association for Computational Linguistics.

Koehn, P., Hoang, H., Birch, A., Callison-Burch, C., Federico, M., Bertoldi, N., Cowan, B., Shen, W., Moran, C., Zens, R., Dyer, C., Bojar, O., Constantin, A., and Herbst, E. (2007).

Moses: Open source toolkit for statistical machine translation. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics Companion Volume Proceedings of the Demo and Poster Sessions*, pages 177–180, Prague, Czech Republic. Association for Computational Linguistics.

Komachi, M. and Matsumoto, Y. (2006). Phrase reordering for statistical machine translation based on predicate-argument structure. In *Proceedings of the International Workshop on Spoken Language Translation: Evaluation Campaign on Spoken Language Translation*, pages 77–82.

Liu, D. and Gildea, D. (2008). Improved tree-to-string transducer for machine translation. In *Proceedings of the Third Workshop on Statistical Machine Translation*, StatMT '08, pages 62–69, Stroudsburg, PA, USA. Association for Computational Linguistics.

Liu, D. and Gildea, D. (2010). Semantic role features for machine translation. In *Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010)*, pages 716–724, Beijing, China. Coling 2010 Organizing Committee.

Liu, Y., Liu, Q., and Lin, S. (2006). Tree-to-string alignment template for statistical machine translation. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 609–616, Sydney, Australia. Association for Computational Linguistics.

Marcu, D., Wang, W., Echihabi, A., and Knight, K. (2006). Spmt: Statistical machine translation with syntactified target language phrases. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, pages 44–52, Sydney, Australia. Association for Computational Linguistics.

Och, F. J. (2003). Minimum error rate training in statistical machine translation. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, pages 160–167, Sapporo, Japan. Association for Computational Linguistics.

Och, F. J. and Ney, H. (2004). The alignment template approach to statistical machine translation. *Computational Linguistics*, 30:417−449.

Papineni, K., Roukos, S., Ward, T., and Zhu, W.-J. (2002). Bleu: a method for automatic evaluation of machine translation. In *Proceedings of 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

Petrov, S., Barrett, L., Thibaux, R., and Klein, D. (2006). Learning accurate, compact, and interpretable tree annotation. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 433–440, Sydney, Australia. Association for Computational Linguistics.

Stolcke, A. (2002). Srilm – an extensible language modeling toolkit. In *Proceedings of the 7th International Conference on Spoken Language Processing*, pages 901–904, Denver, Colorado, USA, September.

Wang, W., Knight, K., and Marcu, D. (2007). Binarizing syntax trees to improve syntax-based machine translation accuracy. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 746–754, Prague, Czech Republic. Association for Computational

Linguistics.

Wu, D. and Fung, P. (2009a). Can semantic role labeling improve smt. In *Proceedings of the 13th Annual Conference of the EAMT*, pages 218–225, Barcelona, May.

Wu, D. and Fung, P. (2009b). Semantic roles for smt: A hybrid two-pass model. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics, Companion Volume: Short Papers*, pages 13–16, Boulder, Colorado. Association for Computational Linguistics.

Wu, S. and Palmer, M. (2011). Semantic mapping using automatic word alignment and semantic role labeling. In *Proceedings of the Fifth Workshop on Syntax, Semantics and Structure in Statistical Translation*, SSST-5, pages 21–30, Stroudsburg, PA, USA. Association for Computational Linguistics.

Wu, X., Sudoh, K., Duh, K., Tsukada, H., and Nagata, M. (2011). Extracting pre-ordering rules from predicate-argument structures. In *Proceedings of 5th International Joint Conference on Natural Language Processing*, pages 29–37, Chiang Mai, Thailand. Asian Federation of Natural Language Processing.

Xiong, D., Liu, Q., and Lin, S. (2006). Maximum entropy based phrase reordering model for statistical machine translation. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 521–528, Sydney, Australia. Association for Computational Linguistics.

Xiong, D., Zhang, M., and Li, H. (2011). A maximum-entropy segmentation model for statistical machine translation. *IEEE Transactions on Audio, Speech and Language Processing*, 19(8):2494–2505.

Xiong, D., Zhang, M., and Li, H. (2012). Modeling the translation of predicate-argument structure for smt. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 902–911, Jeju Island, Korea. Association for Computational Linguistics.

Xue, N. (2008). Labeling chinese predicates with semantic roles. *Computational Linguistics*, 34(2): 225-255.

Zhai, F., Zhang, J., Zhou, Y., and Zong, C. (2011). Simple but Effective Approaches to Improving Tree-to-Tree Model. *MT-Summit-11*. pages 261-268.

Zhang, H., Zhang, M., Li, H., and Chng, E. S. (2010). Non-isomorphic forest pair translation. In Proceedings *of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 440–450, Cambridge, MA. Association for Computational Linguistics.

Zhang, J., Zhai, F., and Zong, C. (2011). Augmenting string-to-tree translation models with fuzzy use of source-side syntax. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 204–215, Edinburgh, Scotland, UK. Association for Computational Linguistics.

Zhang, J. and Zong, C. (2009). A framework for effectively integrating hard and soft syntactic rules into phrase based translation. In *Proceedings of the 23rd Pacific Asia Conference on Language, Information and Computation*, pages 579–588, Hong Kong. City University of Hong Kong.

Zhuang, T. and Zong, C. (2010a). Joint inference for bilingual semantic role labeling. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 304–314, Cambridge, MA. Association for Computational Linguistics.

Zhuang, T. and Zong, C. (2010b). A minimum error weighting combination strategy for chinese semantic role labeling. In *Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010)*, pages 1362–1370, Beijing, China. Coling 2010 Organizing Committee.

Tree-based Translation without Using Parse Trees

Feifei ZHAI, Jiajun ZHANG, Yu ZHOU and Chengqing ZONG
National Laboratory of Pattern Recognition, Institute of Automation,
Chinese Academy of Sciences, Beijing, China
{ffzhai, jjzhang, yzhou, cqzong}@nlpr.ia.ac.cn

ABSTRACT

Parse trees are indispensable to the existing tree-based translation models. However, there exist two major challenges in utilizing parse trees: 1) For most language pairs, it is hard to get parse trees due to the lack of syntactic resources for training. 2) Numerous parse trees are not compatible with word alignment which is generally learned by GIZA++. Therefore, a number of useful translation rules are often excluded. To overcome these two problems, in this paper we make a great effort to bypass the parse trees and induce effective unsupervised trees for tree-based translation models. Our unsupervised trees depend only on the word alignment without utilizing any syntactic resource or linguistic parser. Hence, they are very beneficial for the translation between resource-poor languages. Our experimental results have shown that the string-to-tree translation system using our unsupervised trees significantly outperforms the string-to-tree system using parse trees.

KEYWORDS : Tree-based translation; Unsupervised tree; EM algorithm.

Proceedings of COLING 2012: Technical Papers, pages 3037–3054,
COLING 2012, Mumbai, December 2012.

1 Introduction

Recently, ***tree-based models***[1] have been widely studied in statistical machine translation (SMT). The existing tree-based models include ***string-to-tree models*** (Galley et al., 2006; Marcu et al., 2006; Shen et al., 2008), ***tree-to-string models*** (Quirk et al., 2005; Liu et al., 2006; Huang et al., 2006;), and ***tree-to-tree models*** (Eisner, 2003; Ding and Palmer, 2005; Cowan et al., 2006; Zhang et al., 2008; Liu et al., 2009). Due to the effective use of syntactic information, tree-based models have achieved comparable (Liu et al., 2009) and even better performance over phrase-based models (Marcu et al., 2006).

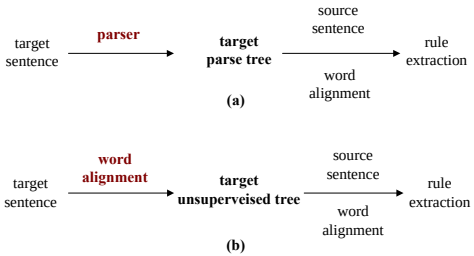


FIGURE 1 – Rule extraction for string-to-tree translation model: (a) using parse trees versus (b) using our unsupervised trees.

In the existing tree-based translation models, parse trees are essential to extracting translation rules. FIGURE 1(a) illustrates the rule extraction process of string-to-tree translation model. The parse tree is usually generated by a linguistic parser which is trained on a manually annotated corpus, such as Treebank. However, the manually annotated corpus is always too inadequate to fully display the strengths of tree-based models. In particular, traditional tree-based systems can not work at all for language pairs without any syntactic resource, which has greatly restricted their application.

From FIGURE 1(a), we can also discover that syntactic parsing is completely independent of word alignment. The separation of parser and alignment leads to a severe incompatibility problem between them. Together with the widely existing parsing errors, numerous useful translation rules are excluded during rule extraction.

To overcome the above two problems of current tree-based models, in this paper, we give up using parse trees and induce better alternatives for tree-based translation models. The alternative tree structures depend only on the word alignment without utilizing any syntactic resource or linguistic parser (see FIGURE 1(b) for illustration). Specifically, the entire process of inducing such tree structures for tree-based translation models is summarized as follows:

1. Based on a word aligned parallel corpus, we first transform those aligned bilingual sentence pairs into packed forests.

[1] The translation models using parse trees on one side or both sides are defined as tree-based models here.

2. Based on the obtained packed forests, we design an EM algorithm to learn an effective synchronous tree substitution grammar (STSG) and then acquire Viterbi tree structures according to the achieved STSG.

In step 1, in order to create a packed forest for a bilingual sentence pair, we first segment the sentence pair into several shorter ones to reduce the huge generation space of tree structures. Then, according to *frontier node assumption*, we compress all the tree structures with the largest number of frontier nodes into a packed forest. We will detail the process of constructing packed forest in Section 3. After all packed forests are generated, we exploit an EM algorithm in step 2 to learn an STSG and then generate Viterbi tree structures for translation. The adopted EM algorithm will be elaborated in Section 4.

Obviously, the above process of inducing tree structures is unsupervised. The syntactic resources and linguistic parsers are not necessary. Hence, comparing with parse trees, the proposed unsupervised trees can be applied to build translation models for more language pairs. Furthermore, by maximizing the number of frontier nodes, the unsupervised trees are compatible with word alignment and thus could achieve a better rule coverage for translation.

Since the existing tree-based translation models are usually restricted by parse trees, using unsupervised trees would be a promising direction for these models. To our best knowledge, this paper is the first effort to introduce effective unsupervised tree structures for tree-based translation models. The most significant contribution of this paper lies in this point. In order to achieve this goal, a series of useful techniques are employed innovatively and meaningfully in the paper. Moreover, the experimental results show that our unsupervised trees significantly outperform the parse trees in the state-of-the-art string-to-tree translation system.

2 Related Work

Our work focuses on inducing effective unsupervised tree structures, and meanwhile, resolving the incompatibility problem between tree structures and word alignment for tree-based translation.

Several researchers have studied unsupervised tree structure induction for different objectives. Blunsom et al. (2008, 2009, 2010) utilized Bayesian methods to learn synchronous context free grammar (SCFG) from a parallel corpus. The obtained SCFG grammar is further used in a phrase-based and hierarchical phrase-based system (Chiang, 2007). Denero and Uszkoreit (2011) adopted a parallel parsing model to induce unlabeled tree structures for syntactic pre-reordering. Different from above works, we concentrate on producing effective and labeled unsupervised trees for tree-based translation models. Moreover, since most of the current tree-based translation models are based on synchronous tree substitution grammar (STSG), our unsupervised trees are thus learned according to STSG, rather than SCFG.

On relieving the incompatibility problem between tree structures and word alignment for translation, previous works mainly focus on two directions:

One direction is to adapt the parse tree structure. Wang et al., (2007) binarized the parse trees and adopted an EM algorithm to select the best binary tree from their parallel binarization forest. Mi et al., (2008b) and Liu et al., (2009) compressed thousands of parse trees into packed forests. Zhang et al. (2011a) applied a CKY binarization method on parse trees to get binary forests for forest-to-string model. Burkett and Klein (2012) adopted a transformation-based method to learn a sequence of monolingual tree transformations for translation. They differ from our work in that

they were all based on parse trees. Compared with them, we construct effective unsupervised tree structures according to the word alignment and do not need any syntactic resource.

The other direction is to integrate the alignment information into parsing. Burkett and Klein (2008) and Burkett et al. (2010) made efforts to do joint parsing and alignment. They utilized the bilingual Treebank to train a joint model and achieved better results on both parsing and word alignment. Liu et al. (2012) re-trained the linguistic parsers bilingually based on word alignment. Our work is different from theirs in that we are pursuing better unsupervised tree structures for better translation performance.

As a whole, compared with previous works, our unsupervised trees are generated fully depending on word alignment. Therefore, by using our tree structures, the incompatibility problem between tree structures and word alignment can be well resolved.

3 Packed Forest Generation

In this section, we introduce how to compress all the reasonable tree structures into a packed forest for the given flat sentence. Packed forest is a compact representation of many tree structures. Generally, it is a pair $<V, E>$ where V is the set of *forest nodes* and E is the set of *hyperedges*. Each hyperedge $e \in E$ is a pair $< h(e), t(e) >$ where *h(e)* is its head node and *t(e)* denotes the vector of its tail nodes. FIGURE 2 illustrates a packed forest that encodes two different tree structures.

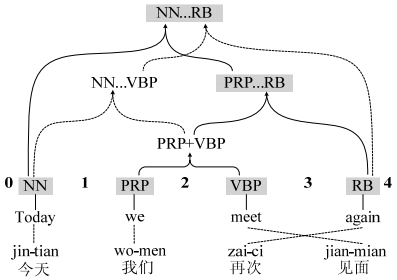


FIGURE 2 – An example of packed forest that encodes two different tree structures. In the FIGURE, shaded nodes denote frontier nodes

3.1 Space Reduction

Basically, there are an exponential number of possible tree structures for a given sentence. These tree structures result in a very huge packed forest. For example, considering a sentence of length L, there will be $0.5L(L+1)$ non-leaf nodes in the forest (each span corresponds to one node). In addition, a forest node covering ***m (m>=2)*** words emits ***m***-1 binary hyperedges[2], leading to $O(L^3)$ hyperedges in total[3]. Consequently, there would be an exponential number of parameters

[2] In our forest, the binary structure serves as the basic unit, which will be demonstrated later.

[3] There are a total of $L + \sum_{i=2}^{L}(L-i+1)\times(i-1) = \frac{1}{6}(L^3+5L)$ edges, including the edges linking to the leaf nodes.

for the STSG achieved from the forests. Such many parameters would cause a difficult estimation problem, especially for the EM algorithm adopted in this paper. Therefore, to reduce the huge generation space of tree structures, we first segment the bilingual sentence pair into several shorter ones and then impose *frontier node assumption* to construct the packed forest.

3.1.1 Bilingual Sentence Segmentation

Bilingual sentence segmentation is to segment a sentence pair into several short sub-sentence pairs whose source sub-sentence and target sub-sentence are translations to each other. Theoretically, in a sub-sentence pair, all included words cannot align to words outside it. However, since many words are wrongly aligned via the automatic word alignment, numerous correct aligned sub-sentence pairs are often excluded under this restriction. Therefore, to relax the restriction, we adopt the following constrains after analyzing the erroneous alignments[4]: (a) the bidirectional length ratios of a sub-sentence pair must be all smaller than 1:3; (b) as a sub-sentence pair, it must contain more than 4 words on each side; (c) in a sub-sentence pair, more than 30% words on each side must be aligned to its counterparts; (d) considering all the alignment links emitted by a sub-sentence, the erroneous ones (align to words outside the sub-sentence pair) account for at most 30% of all links.

To guarantee the segmentation accuracy, we only extract split point candidates based on the punctuations[5] which always denote the boundary of sub-sentences. Complying with the above constraints, we traverse all the split point candidates to search for the optimal split point with minimum number of wrongly aligned words (i.e., minimizing the number of words that align to words outside the sub-sentence pair). Then we segment the sentence pair into two shorter ones at that split point. We recursively segment the newly acquired sub-sentence pairs until no split point candidate is left. FIGURE 3(a) shows the segmentation result of an example sentence pair.

After bilingual sentence segmentation, only the spans inside the sub-sentence pairs are used in the forest. Under this condition, a large amount of useless spans are discarded and the forest is effectively simplified. For example, in FIGURE 3(b), the span “meet again, but” in the English sentence is discarded because it does not belong to any sub-sentence pair.

Note that the method of bilingual sentence segmentation we use here is only a simple segmentation strategy. It can also be substituted by any other segmentation methods. Additionally, after sentence segmentation, we realign words based on the sub-sentence pairs to get a new alignment where all words in a sub-sentence pair align to the words inside it.

3.1.2 Frontier Node Assumption

Bilingual sentence segmentation leads to a great space reduction for constructing packed forests. However, even after sentence segmentation, the generation space of tree structures would be still very large, especially when the sub-sentence is very long. In order to further simplify the space, we take advantage of the following assumption during forest construction:

Frontier Node Assumption*: The more frontier nodes the tree structure has, the more reasonable it is for translation.*

[4] The heuristic values in the constraints are chosen by a series of survey and experiments on a well-aligned corpus.
[5] We use {。 ， ： ； ？ ！} and {. , : ; ? !} as split anchors for Chinese and English, respectively. We take the position before and after the punctuations as split point candidates.

Frontier nodes are utilized to factor a tree structure into several fragments for rule extraction (Galley et al., 2004). Formally, a frontier node is a node that meets the following constraint: the span of the node and its dominated span at the other side are consistent with word alignment[6]. For example, in FIGURE 2, node *PRP…RB*'s span is {we meet again} and it dominates span {*我们 再次 见面*} at the other side. These two spans are consistent with word alignment. Therefore, node *PRP…RB* is a frontier node.

Our *frontier node assumption* makes sense in tree-based translation model. This is because with the purpose of achieving better rule coverage, we tend to extract small minimal rules as many as possible and generate larger rules by composing them. Maximizing the number of frontier nodes supports this goal, while producing many interior (non-frontier) nodes hinders it (DeNero and Klein, 2007). Hence, in the forest constructor, we follow this assumption and only consider the tree structures with the largest number of frontier nodes.

Denero and Uszkoreit (2011) utilized a similar heuristic to construct their unlabeled trees. They required that all spans in their trees must align continuously to the other side. Unlike their heuristic, our *frontier node assumption* only maximizes the number of frontier nodes. The interior nodes are also permitted in the tree structure, which is more flexible and appropriate for constructing forests.

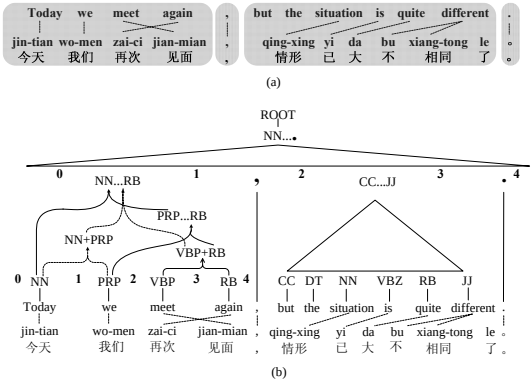


FIGURE 3 – (a) An example of bilingual sentence segmentation. (b) The ultimate packed forest of the example sentence pair in (a).

3.2 Node Labeling

To create packed forests for sentences, a problem that must be resolved is how to label the forest nodes without any syntactic knowledge. Xiong et al. (2006) showed that the boundary word of a phrase is a very effective indicator for phrase reordering. Zollmann and Vogel (2011) labeled hierarchical rules with word classes of boundary words and achieved better translation

[6] A node's dominated span at the other side refers to the minimum continuous span covering all the words that are reachable from the node via word alignment. Two spans are consistent with word alignment means that words in one span only align to words in the other span via word alignment, and vice versa.

performance. Inspired by their work, we combine word classes of boundary words to label forest nodes. We divide the non-leaf forest nodes into three groups: one-word node, dominating only one word in the sentence, and accordingly, two-word node, and multi-word node. Naturally, a one-word node is labeled by the class of its dominating word; a two-word node is labeled by combining the classes of the two words, such as "*C*1+*C*2"; a multi-word node, whose leftmost word's class is *C*1 and rightmost word's class is *C*n, is labeled with "*C*1…*C*n". In this paper, POS tags are employed to serve as the word classes[7]. For example, in FIGURE 3(b), the forest node covering phrase "we meet again" is a multi-word node and is labeled with "*PRP…RB*". Our labeling strategy is similar to (Zollmann and Vogel 2011). The difference is that we are labeling our forest nodes, while they labeled hierarchical rules to substitute the original single non-terminal *X*.

3.3 Forest Constructor

In tree-based translation models, the binary structure has shown its efficiency on improving translation quality (Wang et al., 2007; Zhang et al., 2011a). Inspired by this, we take the binary structure as the basic unit of our forest.

After sentence segmentation, we first build a forest for each sub-sentence pair. Initially, we create a POS node for each word and then perform a CKY-style algorithm to construct forests. FIGURE 4 illustrates the main process of building forest for a sub-sentence pair in FIGURE 3(a). From FIGURE 4 we can see that, the algorithm continuously inspects each span[8] in a bottom-up manner and creates forest nodes to represent the spans.

During the above process, we check every split point in each span and generate an edge[9] for that split point. To comply with the frontier node assumption, we only preserve the edges maximizing $F[i, j]$:

$$F[i, j] = \arg\max_k \{F[i,k] + F[k, j] + Fron[i, j]\} \quad (1)$$

where $F[i, j]$ denotes the number of frontier nodes in the sub-tree whose root node is span $[i, j]$. $k \in (i, j)$ refers to the split point of span $[i, j]$. $Fron[i, j]$ is an indicator function whose value is 1 if the node for span $[i, j]$ is a frontier node and 0 otherwise. Obviously, Equation (1) guarantees that the sub-tree rooted at span $[i, j]$ carries the largest number of fronter nodes. Consequently, in a bottom-up manner, when we arrive at the node covering the whole sentence, we can achieve all the tree structures with the larget number of frontier nodes.

For example, in FIGURE 4(c), span [0,3] (length L=3) has two split points and thus can be composed of span [0,1] and span [1,3], or span [0,2] and span [2,3]. However, just as FIGURE 4(c) shows, there are only 3 frontier nodes in the former case ($F[0,1] + F[1,3] + Fron[0,3] = 3$, here $Fron[0,3] = 0$ because node *NN…VBP[0,3]* is not a frontier node), while there are 4 frontier

[7] Practically, we need a supervised POS tagger, which impairs the unsupervised property of our tree structure to some extent. Actually, the POS tags can be substituted by any unsupervised word classes here. In future, we also plan to design an efficient algorithm to learn the node label automatically, rather than using a heuristic like here.

[8] Here the "span" is based on the POS nodes. For example in FIGURE 4, span 0-2 refers to the span of node sequence "NN PRP".

[9] As each split point corresponds to two adjacent smaller spans, we generate an edge to link these two spans. Therefore, each edge we create here contains a head node and only two tail nodes.

nodes in the latter one ($F[0,2]+F[2,3]+Fron[0,3]=4$). Therefore, we only preserve the edge maximizing $F[0,3]$ for node *NN…VBP[0,3]*, i.e., the edge composed of span [0,2] and span [2,3].

In most cases, a forest node could emit more than one edge with the same largest number of frontier nodes. For example, in FIGURE 4(d), node *NN…RB [0,4]* (length *L*=4) emits two edges with 7 frontier nodes. One links node *NN[0,1]* and *PRP…RB[1,4]*. The other one connects node *NN+PRP[0,2]* and *VPB+RB[2,4]*. We preserve both of these two edges for node *NN…RB [0,4]*. Finally, we achieve the packed forest for the example sub-sentence pair in FIGURE 4(d). In addition, during the forest construction process, the lower nodes and edges not chosen to create upper level nodes will be discarded, such as node *NN…VBP[0,3]* in FIGURE 4(d).

After we create a forest for each sub-sentence pair (named as sub-forest), we combine these sub-forests together to generate a final binary forest for the whole sentence pair. The combination is also performed by a similar CKY-style algorithm. The only difference is that the span in this CKY algorithm is based on the root node of sub-forests. Then we add a goal root node labeled "*ROOT*" to the forest which will be used in the EM algorithm. As an example, FIGURE 3(b) illustrates the final packed forest of the example sentence pair in FIGURE 3(a).

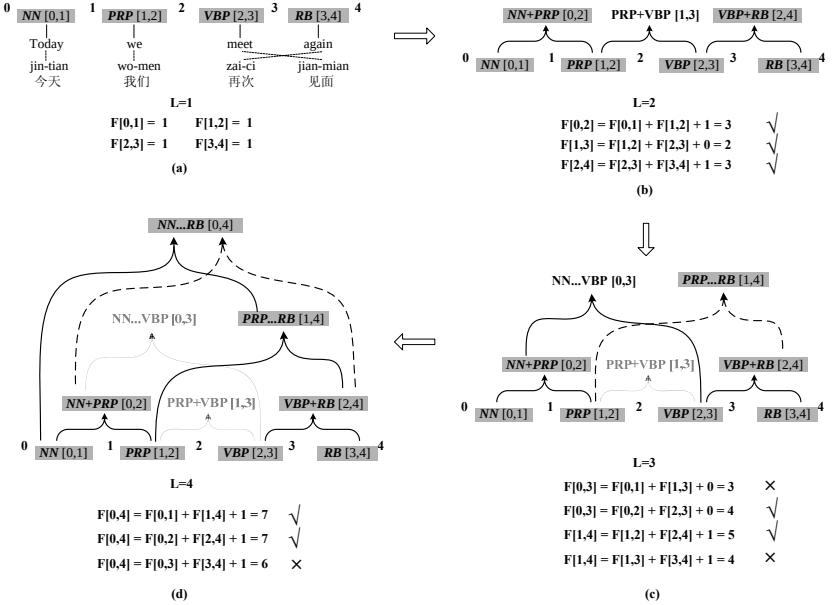


FIGURE 4 – The main process of building forest with the CKY-style algorithm. Here, shaded nodes denote frontier nodes. *L* refers to the length of span. F[i,j] denotes the number of frontier nodes in the sub-tree whose root node is span [i,j]. For example, the sub-tree that roots at span [1,4] in (c) contains 5 frontier nodes. This is because F[1,2] = 1 for node *PRP[1,2]*, and F[2,4] = 3 for node *VBP+RB[2,4]*, and node *PRP…RB [1,4]* is also a frontier node.

4 Learning Viterbi Tree via EM Algorithm

In this section, we design an EM algorithm to learn an effective synchronous tree substitution grammar (STSG) and then acquire Viterbi tree structures based on the STSG. Denero and Uszkoreit (2011) mentioned that their unlabeled tree structures can also be obtained by a similar method. However, their method is based on SCFG. Our work is different from theirs in that the EM algorithm is based on STSG. We use STSG here because most of the current tree-based translation models are based on STSG.

Given a parallel corpus with *n* sentence pairs, and the corresponding packed forest for each target sentence ***e***, we aim to search for a series of trees $(t_{e1}, t_{e2}, ..., t_{en})^*$ that maximize the likelihood of the whole corpus (t_e, f, a) [10], which is formulated as follows:

$$(t_{e1}, t_{e2}, ..., t_{en})^* = \arg\max_{(t_{e1}, t_{e2}, ..., t_{en})} \prod_{i=1}^{n} p(t_{ei}, f_i, a_i)$$

The probability of triple (t_{ei}, f_i, a_i) is further computed by aggregating the rule probabilities $p(r)$ in each derivation *d* in the set of all derivations *D*. That is

$$p(t_{ei}, f_i, a_i) = \sum_{D} \prod_{r \in d} p(r)$$

To get the derivation set *D*, we employ the algorithm of Mi et al., (2008a) to transform our induced packed forests into synchronous derivation forests. Practically, in order to reduce the complexity of the derivation forest, we only utilize the minimal STSG translation rules extracted by the method of Galley et al., (2004) and Mi et al., (2008b) to construct derivation forests[11].

Using the synchronous derivation forests, the rule probabilities are estimated by the inside-outside algorithm (Graehl and Knight, 2004). Here, leaf(***r***) and root(***r***) denote the leaf non-terminals and root node of rule ***r*** respectively. The inside and outside probabilities of forest node ***N*** are defined as follows,

$$p_{IN}(N) = \sum_{r \in R(N)} \left[p(r) \cdot \prod_{N_l \in leaf(r)} p_{IN}(N_l) \right]$$

$$p_{OUT}(N) = \sum_{r:\, N \in leaf(r)} \left[p(r) \cdot p_{OUT}(root(r)) \cdot \prod_{N_l \in leaf(r) - \{N\}} p_{IN}(N_l) \right]$$

where ***R(N)*** denotes the set of matched rules rooted at node ***N***. Therefore, the process of EM algorithm is shown as follows:

[10] In the triple, *te* refers to the target tree structures, *f* denotes the source language sentences, and *a* is the word alignment between them.

[11] We follow the highest attachment strategy in (Galley et al., 2006) to deal with unaligned words.

E-step: the expected count for each occurrence of rule ***r*** in a derivation forest is computed as:

$$p(r) \bullet p_{OUT}(root(r)) \bullet \prod_{N_l \in leaf(r)} p_{IN}(N_l)$$

M-step: the expected counts of rules, *c(r)*, are used to update the probabilities of rules:

$$p(r) = \frac{c(r)}{\sum_{r_a : root(r_a) = root(r)} c(r_a)}$$

After the EM algorithm, we traverse the derivation forest to obtain the Viterbi derivation d^* and its corresponding best tree structure. Then the acquired tree structures can be applied to any traditional tree-based translation system.

5 Experiments

5.1 Experimental Setup

In order to verify the effectiveness of our unsupervised tree structures, we compare them with linguistic parse trees based on string-to-tree translation. Here we experiment on Chinese-to-English translation, for which English parse trees can be easily obtained. Our training data is the FBIS corpus containing about 7.1 million Chinese words and 9.2 million English words. We generate the final symmetric word alignment using GIZA++ and the *grow-diag-final-and* balance strategy. We train a 5-gram language model on the target part of the training corpus and the Xinhua portion of English Gigaword corpus. We use the NIST MT 2003 evaluation data as the development set, and adopt NIST MT04 and MT05 as the test set. The final translation quality is evaluated in terms of case-insensitive BLEU-4 with shortest length penalty. The statistical significance test is performed using the re-sampling approach (Koehn, 2004).

Our baseline system is an in-house string-to-tree system (named ***s2t***) based on Galley et al. (2006) and Marcu et al. (2006). The English side of the training corpus is parsed with Berkeley parser (Petrov et al., 2006). We extract the minimal GHKM rules (Galley et al., 2004) and the rules of SPMT Model 1 (Marcu et al., 2006) with phrases up to length $L=5$ on the source side. Then we extract the composed rules by composing two or three adjacent minimal GHKM rules (Galley et al., 2006). The beam size of the decoder is set as 500. We further implement head binarization on the English parse trees and apply the achieved binary trees to another string-to-tree system (abbreviated as ***s2t-hb***) with the same settings of ***s2t***. In addition, we also run the state-of-the-art hierarchical phrase-based system Joshua (Li et al., 2009) for comparison.

For inducing our unsupervised tree structures, we use Urheen[12] to get the POS tags of the English corpus. Just as we described in section 3.1.1, we reuse GIZA++ and the *grow-diag-final-and* strategy to re-align words based on the sub-sentence pairs and then combine the alignment result together to get a new word alignment for the whole sentence pair. We perform the EM algorithm to capture the final tree structures by 20 iterations. Then we build a string-to-tree system using our induced unsupervised tree structures (abbreviated as ***s2t-IT***). Different from the above

[12] http://www.openpr.org.cn/index.php/NLP-Toolkit-for-Natural-Language-Processing/

baseline system, the beam size of our ***s2t-IT*** system is set as 300 to get a comparable translation speed to the baseline system.

Besides, using the new alignment, we also run the ***s2t*** and ***s2t-hb*** system with the same settings as the abovementioned ***s2t*** and ***s2t-hb*** systems (We mark the systems using the new word alignment as ***re-align*** systems).

5.2 Experimental Results

The translation results of different systems are shown in TABLE 1. As we can see, the performance of the baseline string-to-tree system significantly outperforms the hierarchical phrase-based system *Joshua*, which verifies the superiority of our baseline *s2t* system.

System		MT04	MT05	All
Joshua		30.71	27.86	29.59
s2t (baseline)		33.73*	30.25*	32.75*
s2t-hb		34.09	30.99*	32.92
re-align	***s2t***	33.53	29.30	32.29
	s2t-hb	33.88	30.49*	32.61
	s2t-IT	**34.71#**	**31.55#**	**33.53#**
Number of sentences		1788	1082	2870

TABLE 1 – Results (in case-insensitive BLEU-4 scores) of different systems. The "*" and "#" denote that the result are significantly better than the adjacent above system and all the other systems respectively ($p<0.01$).

TABLE 1 also demonstrates the effectiveness of binary structures. It can be clearly seen that whether we do re-alignment or not, the head binarization approach can always help to improve the *s2t* system (lines 2-5). Besides, as we can see from TABLE 1, the performances of the *re-align s2t* and *s2t-hb* system are slightly worse than the *s2t* and *s2t-hb* system. It indicates that the sentence segmentation method might be harmful to the traditional translation system. We will explore the reason in the next section.

The system using our induced unsupervised trees (*s2t-IT*) achieves the best performance among all the systems. It significantly outperforms the baseline *s2t* system by 0.98 and 1.3 BLEU points on MT04 and MT05 respectively. Furthermore, as shown in TABLE 1, even using the head binarization approach, the performance of the best *s2t-hb* system is still lower than that of our *s2t-IT* system by 0.61 BLEU points on the combined test set. Obviously, the above comparisons strongly demonstrate that our induced unsupervised trees are much more appropriate than parse trees for the string-to-tree translation model.

5.3 Analysis and Discussion

The improvement of translation performance has strongly verified the effectiveness of our induced unsupervised tree structures. We further conduct a series of deep analysis on the result.

We first adopt FIGURE 5, which depicts an example of our unsupervised tree structure and a traditional parse tree structure, to explain the superiority of our unsupervised trees. Comparing these two structures, we can see that our unsupervised tree structure carries more frontier nodes and thus can be factored into more small sub-structures. Consequently, the resulted minimal rules tend to be more general and smaller. For example, in FIGURE 5, rule (c) and (d) are the minimal

rules extracted from the unsupervised tree structure and the parse tree structure respectively to translate Chinese phrase "*有利于* (you-li-yu)". Obviously, rule (c) is much smaller and can be utilized without any limit while rule (d) cannot, since rule (d) requires that the translation after "*is conducive to*" must be dominated by an "***S***" node. Additionally, we can further acquire many big rules by composing several small minimal rules. On the above basis, our induced unsupervised trees are much more conducive to extracting both general enough rules and specific enough rules, which leads to a better rule coverage and translation quality.

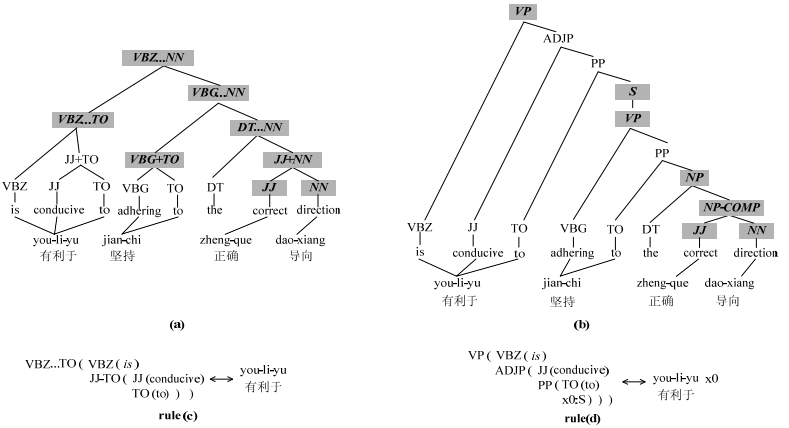


FIGURE 5 – Different tree structures and extracted example rules: (a) the unsupervised tree structures (b) the binary tree structures produced by berkeley parser (The node "NP-COMP" is created by the head binarization approach). The shaded nodes in the FIGURE denote frontier nodes. (c) and (d) are the minimal rules extracted from the structures in (a) and (b) respectively.

To further demonstrate the above analysis, TABLE 2 shows the average number of frontier nodes per tree structure (#Aver-Frontier-Nodes) and the grammar size (#RULES) of different systems. As we can see, the string-to-tree systems using parse trees benefit from the head binarization approach which helps to recall frontier nodes (from 33.9 to 40.4 for *s2t* system and 32.0 to 38.6 for the re-aligned *s2t* system).

System		#Aver-Frontier-Nodes	#RULES
s2t (baseline)		33.9	15.5M
s2t-hb		40.4	28.1M
re-align	***s2t***	32.0	13.8M
	s2t-hb	38.6	26.2M
	s2t-IT	**47.4**	**51.9M**

TABLE 2 – Average number of frontier nodes and grammar size for different string-to-tree systems.

Furthermore, using our induced unsupervised trees, it accounts for 47.4 frontier nodes on average while there are only 33.9 frontier nodes at most in the traditional linguistic parse trees. Obviously,

this comparison indicates that our unsupervised trees are more compatible with the word alignment and are beneficial to extracting more useful translation rules. Just as column "#RULES" shows, our *s2t-IT* system obtains a total of 51.9M rules while the baseline *s2t* system only gets 15.5M rules at most.

We have found that the sentence segmentation method might do harm to the traditional translation system in TABLE 1. TABLE 2 gives a faithful explanation on this phenomenon. As indicated by TABLE 2, after we do re-alignment, the number of frontier nodes decreases (from 33.9 to 32.0 for *s2t*) and the grammar size is reduced at the same time. We believe that the reduced grammar leads to the worse performance of the *re-align s2t* and *s2t-hb* system. Intuitively, the deterioration caused by sentence segmentation would also affect our *s2t-IT* system. However, our *s2t-IT* system still significantly outperforms the baseline *s2t* system. More work would be devoted to alleviate the influence of sentence segmentation.

We further investigate the used tags of tree nodes in our unsupervised trees. According to the statistics, there are a total of 2,862 tags for the non-leaf nodes in the final corpus of our unsupervised trees. With such many tags, a natural question is that does the grammar extracted from these tree structures suffer from a data sparseness problem? TABLE 3 answers this question in detail. In the TABLE, for example, line 2 denotes that the most frequent 143 tags (5% of all tags) account for 76.5% of all frontier nodes and 82.4% of all tree nodes. As illustrated in TABLE 3, 87.0% frontier nodes and 90.3% tree nodes are labeled with the most frequent 286 tags (10% of all tags), indicating that the vast majority of our translation rules are composed of these tags. Compared with the 70 tags[13] used in the linguistic parse trees, we believe our employed tags are both specific enough for distinguishing different rules and general enough for avoiding the data sparseness problem.

#tag num	#percentage of frontier nodes	#percentage of tree nodes
85(3%)	68.0%	75.8%
143(5%)	76.5%	82.4%
228(8%)	83.6%	87.7%
286(10%)	**87.0%**	**90.3%**
429(15%)	92.3%	94.2%
572(20%)	95.2%	96.4%
…	…	…

TABLE 3 – The proportion of frequently appearing tags in our induced tree structures.

Conclusion and Perspectives

In this paper, we propose effective unsupervised trees to substitute parse trees for tree-based translation models. Since current tree-based translation models are all driven by parse trees, this work creates a brand-new direction for them. We first roughly group the words into several sub-sentence pairs by a bilingual sentence segmentation method. After that, we compress all the reasonable tree structures of sentence pairs into packed forests under *frontier node assumption*. Finally, we design an EM algorithm to learn an effective STSG and then select a best tree structure for each sentence pair.

The unsupervised tree structures are constructed depending on the word alignment. Therefore, they are naturally compatible with word alignment and lead to a better rule coverage.

[13] There are 44 POS tags, 5 clausal tags and 21 phrasal tags for labeling the linguistic parse trees.

Experiments on string-to-tree translation system show that our unsupervised trees significantly outperform the parse trees. We believe that our method is quite beneficial for the translation between resource-poor languages.

In the future, we plan to conduct more experiments on other tree-based models, such as tree-to-string model and tree-to-tree model. Furthermore, we also plan to develop unsupervised methods to jointly induce the tree structure and word alignment for tree-based translation models. This issue is more difficult since the search space is much larger and we plan to employ Bayesian methods with sampling approach to fulfil this task.

Acknowledgments

The research work has been funded by the Natural Science Foundation of China under Grant No. 6097 5053 and the Hi-Tech Research and Development Program ("863" Program) of China under Grant No. 2011AA01A207. This research is also supported by the Singapore National Research Foundation under its International Research Centre @ Singapore Funding Initiative and administered by the IDM Programme Office. We also thank the anonymous reviewers for their valuable suggestions.

References

Blunsom, P. and Cohn, T. (2010). Inducing synchronous grammars with slice sampling. In *Proceedings of Human Language Technologies: The 11th Annual Conference of the North American Chapter of the Association for Computational Linguistics (HLT-NAACL)*, pages 238–241, Los Angeles, California.

Blunsom, P., Cohn, T., Dyer, C., and Osborne, M. (2009). A gibbs sampler for phrasal synchronous grammar induction. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP (ACL-IJCNLP)*, pages 782–790, Suntec, Singapore.

Blunsom, P., Cohn, T., and Osborne, M. (2008). Bayesian synchronous grammar induction. In Koller, D., Schuurmans, D., Bengio, Y., and Bottou, L., editors, *Advances in Neural Information Processing Systems (NIPS) 21*, pages 161–168.

Burkett, D., Blitzer, J., and Klein, D. (2010). Joint parsing and alignment with weakly synchronized grammars. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, HLT '10, pages 127–135, Stroudsburg, PA, USA. Association for Computational Linguistics.

Burkett, D. and Klein, D. (2008). Two languages are better than one (for syntactic parsing). In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '08, pages 877–886, Stroudsburg, PA, USA. Association for Computational Linguistics.

Burkett, D. and Klein, D. (2012). Transforming trees to improve syntactic convergence. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 863–872, Jeju Island, Korea. Association for Computational Linguistics.

Chiang, D. (2005). A hierarchical phrase-based model for statistical machine translation. In

Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics, ACL '05, pages 263–270, Stroudsburg, PA, USA. Association for Computational Linguistics.

Chiang, D. (2007). Hierarchical phrase-based translation. *Computational Linguistics*, 33 (2).

Chiang, D., Knight, K., and Wang, W. (2009). 11,001 new features for statistical machine translation. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, NAACL '09, pages 218–226, Stroudsburg, PA, USA. Association for Computational Linguistics.

Cohn, T. and Blunsom, P. (2009). A Bayesian model of syntax-directed tree to string grammar induction. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 352–361, Singapore. Association for Computational Linguistics.

Cowan, B., Ku˘cerová, I., and Collins, M. (2006). A discriminative model for tree-to-tree translation. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, pages 232–241, Sydney, Australia. Association for Computational Linguistics.

DeNeefe, S., Knight, K., Wang, W., and Marcu, D. (2007). What can syntax-based MT learn from phrase-based MT? In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 755–763, Prague, Czech Republic. Association for Computational Linguistics.

DeNero, J. and Klein, D. (2007). Tailoring word alignments to syntactic machine translation. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 17–24, Prague, Czech Republic. Association for Computational Linguistics.

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

Ding, Y. and Palmer, M. (2005). Machine translation using probabilistic synchronous dependency insertion grammars. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 541–548, Ann Arbor, Michigan. Association for Computational Linguistics.

Eisner, J. (2003). Learning non-isomorphic tree mappings for machine translation. In *The Companion Volume to the Proceedings of 41st Annual Meeting of the Association for Computational Linguistics*, pages 205–208, Sapporo, Japan. Association for Computational Linguistics.

Galley, M., Graehl, J., Knight, K., Marcu, D., DeNeefe, S., Wang, W., and Thayer, I. (2006). Scalable inference and training of context-rich syntactic translation models. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 961–968, Sydney, Australia. Association for Computational Linguistics.

Galley, M., Hopkins, M., Knight, K., and Marcu, D. (2004). What's in a translation rule? In Susan Dumais, D. M. and Roukos, S., editors, *HLT-NAACL 2004: Main Proceedings*, pages 273–280, Boston, Massachusetts, USA. Association for Computational Linguistics.

Graehl, J. and Knight, K. (2004). Training tree transducers. In Susan Dumais, D. M. and

Roukos, S., editors, *HLT-NAACL 2004: Main Proceedings*, pages 105–112, Boston, Massachusetts, USA. Association for Computational Linguistics.

Huang, L. and Chiang, D. (2007). Forest rescoring: Faster decoding with integrated language models. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 144–151, Prague, Czech Republic. Association for Computational Linguistics.

Huang, L., Knight, K., and Joshi, A. (2006). A syntax-directed translator with extended domain of locality. In *Proceedings of the Workshop on Computationally Hard Problems and Joint Inference in Speech and Language Processing*, CHSLP '06, pages 1–8, Stroudsburg, PA, USA. Association for Computational Linguistics.

Koehn, P., Och, F. J., and Marcu, D. (2003). Statistical Phrase-Based Translation, In *Proc. of HLT/NAACL 2003*.

Koehn, P. (2004). Statistical significance tests for machine translation evaluation. In Lin, D. and Wu, D., editors, *Proceedings of EMNLP 2004*, pages 388–395, Barcelona, Spain. Association for Computational Linguistics.

Koehn, P., Hoang, H., Birch, A., Callison-Burch, C., Federico, M., Bertoldi, N., Cowan, B., Shen, W., Moran, C., Zens, R., Dyer, C., Bojar, O., Constantin, A., and Herbst, E. (2007). Moses: Open source toolkit for statistical machine translation. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics Companion Volume Proceedings of the Demo and Poster Sessions*, pages 177–180, Prague, Czech Republic. Association for Computational Linguistics.

Li, Z., Callison-Burch, C., Dyery, C., Ganitkevitch, J., Khudanpur, S., Schwartz, L., Thornton, W. N. G., Weese, J., and Zaidan, O. F. (2009). Demonstration of joshua: An open source toolkit for parsing-based machine translation. In *Proceedings of the ACL-IJCNLP 2009 Software Demonstrations*, pages 25–28, Suntec, Singapore. Association for Computational Linguistics.

Liu, S., Li, C.-H., Li, M., and Zhou, M. (2012). Re-training monolingual parser bilingually for syntactic smt. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 854–862, Jeju Island, Korea. Association for Computational Linguistics.

Liu, Y., Liu, Q., and Lin, S. (2006). Tree-to-string alignment template for statistical machine translation. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 609–616, Sydney, Australia. Association for Computational Linguistics.

Liu, Y., Lü, Y., and Liu, Q. (2009). Improving tree-to-tree translation with packed forests. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 558–566, Suntec, Singapore. Association for Computational Linguistics.

Marcu, D., Wang, W., Echihabi, A., and Knight, K. (2006). Spmt: Statistical machine translation with syntactified target language phrases. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, pages 44–52, Sydney, Australia. Association for Computational Linguistics.

Mi, H. and Huang, L. (2008). Forest-based translation rule extraction. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 206–214, Honolulu, Hawaii. Association for Computational Linguistics.

Mi, H., Huang, L., and Liu, Q. (2008). Forest-based translation. In *Proceedings of ACL-08: HLT*, pages 192–199, Columbus, Ohio. Association for Computational Linguistics.

Och, F. J. (2003). Minimum error rate training in statistical machine translation. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, pages 160–167, Sapporo, Japan. Association for Computational Linguistics.

Papineni, K., Roukos, S., Ward, T., and Zhu, W.-J. (2002). Bleu: a method for automatic evaluation of machine translation. In *Proceedings of 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

Petrov, S., Barrett, L., Thibaux, R., and Klein, D. (2006). Learning accurate, compact, and interpretable tree annotation. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 433–440, Sydney, Australia. Association for Computational Linguistics.

Quirk, C., Menezes, A., and Cherry, C. (2005). Dependency treelet translation: Syntactically informed phrasal SMT. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 271–279, Ann Arbor, Michigan. Association for Computational Linguistics.

Shen, L., Xu, J., and Weischedel, R. (2008). A new string-to-dependency machine translation algorithm with a target dependency language model. In *Proceedings of ACL-08: HLT*, pages 577–585, Columbus, Ohio. Association for Computational Linguistics.

Snyder, B., Naseem, T., and Barzilay, R. (2009). Unsupervised multilingual grammar induction. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 73–81, Suntec, Singapore. Association for Computational Linguistics.

Wang, W., Knight, K., and Marcu, D. (2007). Binarizing syntax trees to improve syntax-based machine translation accuracy. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 746–754, Prague, Czech Republic. Association for Computational Linguistics.

Wang, W., May J., Knight, K., and Marcu, D. (2010). Re-structuring, re-labeling, and re-aligning for syntax-based machine translation. *Computational Linguistics*, 2010.

Xiong, D., Liu, Q., and Lin, S. (2006). Maximum entropy based phrase reordering model for statistical machine translation. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 521–528, Sydney, Australia. Association for Computational Linguistics.

Xiong, D., Zhang, M., and Li, H. (2010). Learning translation boundaries for phrase-based decoding. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 136–144, Los

Angeles, California. Association for Computational Linguistics.

Zhai, F., Zhang, J., Zhou, Y., and Zong, C. (2011). Simple but Effective Approaches to Improving Tree-to-Tree Model. *MT-Summit-11*. pages 261-268.

Zhang, H., Fang, L., Xu, P., and Wu, X. (2011a). Binarized forest to string translation. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 835–845, Portland, Oregon, USA. Association for Computational Linguistics.

Zhang, H., Huang, L., Gildea, D., and Knight, K. (2006). Synchronous binarization for machine translation. In *Proceedings of the Human Language Technology Conference of the NAACL, Main Conference*, pages 256–263, New York City, USA. Association for Computational Linguistics.

Zhang, H., Zhang, M., Li, H., Aw, A., and Tan, C. L. (2009). Forest-based tree sequence to string translation model. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 172–180, Suntec, Singapore. Association for Computational Linguistics.

Zhang, J., Zhai, F., and Zong, C. (2011b). Augmenting string-to-tree translation models with fuzzy use of source-side syntax. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 204–215, Edinburgh, Scotland, UK. Association for Computational Linguistics.

Zhang, J. and Zong, C. (2009). A framework for effectively integrating hard and soft syntactic rules into phrase based translation. In *Proceedings of the 23rd Pacific Asia Conference on Language, Information and Computation*, pages 579–588, Hong Kong. City University of Hong Kong.

Zhang, M., Jiang, H., Aw, A., Li, H., Tan, C. L., and Li, S. (2007). A Tree-to-Tree Alignment-based model for statistical Machine translation. *MT-Summit-07*. pages 535-542.

Zhang, M., Jiang, H., Aw, A., Li, H., Tan, C. L., and Li, S. (2008). A tree sequence alignment-based tree-to-tree translation model. In *Proceedings of ACL-08: HLT*, pages 559–567, Columbus, Ohio. Association for Computational Linguistics.

Zollmann, A. and Venugopal, A. (2006). Syntax augmented machine translation via chart parsing. In *Proceedings of the Workshop on Statistical Machine Translation*, StatMT '06, pages 138–141, Stroudsburg, PA, USA. Association for Computational Linguistics.

Zollmann, A. and Vogel, S. (2011). A word-class approach to labeling pscfg rules for machine translation. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 1–11, Portland, Oregon, USA. Association for Computational Linguistics.

Constructing Chinese Abbreviation Dictionary: A Stacked Approach

Longkai Zhang *Sujian Li* *Houfeng Wang** *Ni Sun* *Xinfan Meng*
Key Laboratory of Computational Linguistics (Peking University) Ministry of Education, China
zhlongk@qq.com, lisujian@pku.edu.cn, wanghf@pku.edu.cn,
sunny.forwork@gmail.com, mxf@pku.edu.cn

ABSTRACT
Abbreviation is a common linguistic phenomenon with wide popularity and high rate of growth. Correctly linking full forms to their abbreviations will be helpful in many applications. For example, it can improve the recall of information retrieval systems. An intuition to solve this is to build an abbreviation dictionary in advance. This paper investigates an automatic abbreviation generation method, which uses a stacked approach for Chinese abbreviation generation. We tackle this problem in two stages. First we use a sequence labeling method to generate a list of candidate abbreviations. Then, we try to use search engine to incorporate web data to re-rank the candidates, and finally get the best candidate. We use a Chinese abbreviation corpus which contains 8015 abbreviation pairs to evaluate the performance. Experiments revealed that our method gave better performance than the baseline methods.

KEYWORDS: Chinese Abbreviation Generation, Abbreviation Mining.

*Corresponding author

Proceedings of COLING 2012: Technical Papers, pages 3055–3070,
COLING 2012, Mumbai, December 2012.

1 INTRODUCTION

Abbreviation is defined as a short description of the original long phrase. For example, "ACL" is the abbreviation for the full form "Association for Computational Linguistics". While abbreviation is a common linguistic phenomenon, it causes many problems like spelling variation (Nenadić et al., 2002).The different writing manners make it difficult to identify the terms conveying the same concept, which will hurt the performance of many applications, such as information retrieval (IR) systems.

In IR applications, one simple solution is to expand the original query by adding corresponding abbreviations to a search engine. For example, when using a search engine with an original query of "United States of America", a user will get more relevant results by expanding the query to include the abbreviation "USA." To achieve this we need to have an abbreviation dictionary, which is laborious to manually maintain because the number of abbreviations increases rapidly (Chang and Schutze, 2006). Therefore, it is helpful to automatically generate abbreviation from full forms. This leads to the idea of "abbreviation generation", i.e., finding the correct abbreviation for a full form.

The generation of abbreviations in Chinese differs from that for English. The reason is that Chinese itself lacks many commonly considered features in English abbreviation generation methods (Pakhomov, 2002; Yu et al., 2006; HaCohen-Kerner et al., 2008; Ao and Takagi, 2005). Detailed differences between English abbreviation generation and Chinese abbreviation features are listed in TABLE 1. Due to these differences, specific attention should be paid to Chinese abbreviation generation.

Feature	English	Chinese
Word boundary	YES	NO
Case sensitivity	YES	NO

Table 1: Comparison between Chinese and English abbreviation generation with regards to features.

Most of Chinese abbreviations are generated by selecting representative characters from the full forms[1]. For example, the abbreviation of "北京大学" (Peking University) is "北大" which is generated by selecting the first and third characters, see TABLE 2. This can be tackled from the sequence labeling point of view.

Original	北	京	大	学
Keep/Skip	Keep	Skip	Keep	Skip
Result	北		大	

Table 2: The abbreviation "北大" of "北京大学" (Peking University)

Meanwhile, full forms and abbreviations show linguistic links like co-occurrence in large text materials. If we can find candidate abbreviations and rank them properly, the performance of abbreviation generation can be improved. Web pages can just serve as a large corpus to provide this information. While it is impractical to retrieve and analyze each web page individually, search engine provides an interface to this vast information. When querying a term in a search

[1]A small portion of Chinese abbreviations are not generated from the full form. For example, the abbreviation of "河北省"(He Bei Province) is "冀". However, we can use a look-up table to get this kind of abbreviations.

engine, titles and snippets of pages containing the query terms are returned, which provides a natural corpus for further analysis.

In this paper, we propose a stacked approach to automatically generate Chinese abbreviations. This method consists of a candidate generation phase and a ranking phase. First, we generate a list of candidates for the given full form using sequence labeling method. Then a supervised re-ranking method based on Support Vector Machine (SVM) using web data is applied to find the exact abbreviation.

We evaluate on a Chinese abbreviation corpus and compare it with previous methods. A pure sequence labeling approach by (Sun et al., 2009) and a state-of-art method to incorporate web data by (Jain et al., 2007) are chosen as baseline methods.

The contribution of this paper is that we integrate sequence labeling and web data to create a robust and automatic abbreviation generator. Experiments show that this combination gets better result than existing methods. Using this method we build a Chinese abbreviation dictionary, which later can be used in other NLP applications to help improve performance.

The paper is structured as follows. We first describe our approach. In section 2 we describe the sequence labeling procedure and in section 3 the re-ranking procedure. Experiments are described in section 4. In section 5 we give a detailed analysis of the results. In section 6 related works are introduced, and the paper is concluded in the last section.

2 Candidate Generation

2.1 Sequence Labeling

As mentioned in section 1, the generation of Chinese abbreviations can be formalized as a task of selecting characters from the full form, which can be solved by sequence labeling models. Previous works proved that Conditional Random Fields (CRFs) can outperform other sequence labeling models like MEMMs in abbreviation generation tasks (Sun et al., 2009; Tsuruoka et al., 2005). For this reason we choose CRFs model in the candidate generation stage.

A CRFs model is a type of discriminative probabilistic model most often used for the labeling or parsing of sequential data. Detailed definition of CRF model can be found in (Lafferty et al., 2001; McCallum, 2002; Pinto et al., 2003).

2.2 Labeling strategy

Considering both training efficiency and modeling ability, we use a labeling method which uses four tags, "BIEP". "B" stands for "Beginning character of skipped characters", "I" stands for "Internal character of skipped characters,"E" stands for "End character of skipped characters", and Label "P" means the current character to be preserved in abbreviation. An example is shown in TABLE 3.

2.3 Feature templates

The feature templates we use are as follows. See TABLE 4.

Templates 1 and 2 express uni-grams and bi-grams, which is widely used in abbreviation generation tasks. Template 3 is used to encode the ability of numbers in the generation of Chinese abbreviations. Templates 4 and 5 are designed to detect character duplication, because duplicated characters are often kept only once.

"国家语言文字工作委员会" (National Linguistics Work Committee) The abbreviation is "国家语委" (the 1st, 2nd, 3rd, 9th characters of the full form)	
BIEP	国/P 家/P 语/P 言/B 文/I 字/I 工/I 作/E 委/P 员/B 会/E

Table 3: The abbreviation "国家语委" of " 国家语言文字工作会" (National Linguistics Work Committee)

1. Uni-gram X_i
2. Bigrams (X_i, X_{i+1})
3. Whether X_i is a number
4. Whether character i equals character $i+1$
5. Whether character i equals character $i+2$

Table 4: Feature templates used in our algorithm.

3 Re-ranking

3.1 Re-rank with web data

Many abbreviations simply generated by the CRF model do not actually match the reference abbreviation. The reason is that as a sequence labeling model, CRF gives a most probable abbreviation character sequence by analyzing local information for each character. However, for Chinese abbreviations, local information alone is not adequate.

The full form and its abbreviation naturally co-occur in a large text corpus. This information contributes to the retrieval of an abbreviation given its full form. However, we cannot incorporate this information directly in traditional statistical learning models, because to get this information we first need a list of candidate abbreviations of the full form, which should be obtained in advance. We also observe that although the top-ranked output of the CRF model is not always correct, the true abbreviation very often appears in the top few outputs of the CRF. Therefore we choose to use the output of CRF model as the list of candidates. The remaining job is to find an effective method to re-rank the candidates using some additional information.

The additional information mentioned above can be obtained from search engines. Search engines index huge amount of web pages, providing an efficient interface to such vast information. The results returned by search engines typically contain the total number of related pages, title and snippet for each page. All the above text materials are useful for us to extract "implicit connections" between the full forms and abbreviations to re-rank the candidates generated in the previous phase.

An example of these "implicit connections" is shown in FIGURE 1. In this case we investigate on the search results of the full form "国家语言文字工作委员会" (National Linguistics Work Committee) and its abbreviation "国家语委". From FIGURE 1(a) we can find that the abbreviation "国家语委" appears in the title of the 3rd result when searching the full form. From FIGURE 1(b) we can also find that the full form "国家语言文字工作委员会" co-occurs with the abbreviation in the snippet of the 3rd result when searching the abbreviation. Furthermore, we see that the two queries share the same top-ranked result, which can be inferred from the same URL of the first search result. All of these evidences imply that "国家语委" seems to be the abbreviation for

"国家语言文字工作委员会". Note that the highlighted key words also indicate that the search

(a) Search result of the full form "国家语言文字工作委员会"

(b) Search result of the abbreviation "国家语委"

Figure 1: An example of search results. We can see clearly that the full form and the abbreviation do have implicit connections in search results.

engine itself does not know the correspondence of the two words. In FIGURE 1(a) we can see that in the result containing the abbreviation, the abbreviation itself is not highlighted as a keyword. Instead, it only matches the keyword "国家" (National). Therefore our method just learns to make use of the implicit connections, rather than exploits what the search engine has already learnt.

Besides search results, another appealing source of text corpus that we should mention is Chinese Wikipedia. Wikipedia seems to be more structured, however, we choose not to use Wikipedia in our context because many Chinese abbreviations like coordinate phrases are not collected in wiki-texts. Besides, new abbreviations spring out almost every day, while manually maintained Wikipedia is updated slowly. These shortcomings of Wikipedia make it less competitive than search engines.

For the re-ranking phase, we generate lists of candidates for the training data and label reference abbreviations as positive instances, and the incorrect candidates as negative instances. Then a SVM classifier is trained for its advantage in processing continuous values. The original SVM model itself does not calculate probability, while there are various ways to estimate the probability (Platt et al., 1999). What we use in our approach is the probability a candidate to be labeled as positive. We re-rank these candidates by these probabilities in decreasing order, and choose the first one as the final result.

3.2 Features for re-ranking

The results returned by search engines mainly contain the total number of related pages, title and snippet of each page. Search engines usually automatically highlight the key words in title and snippet by bolding (or red coloring) the keywords. We once considered using the highlighted keywords as counting criterion in our algorithm, but soon we found that this criterion has many deficiencies. Take "清华大学" (Tsinghua University) as an example, one of its false candidate is "清华大", which happens to be the first 3 characters of the full form. When

searching "清华大", many "清华大"s are highlighted, but they are all appears as part of the full form "清华大学". So it will be biased if we choose highlighting as our criterion. All things considered, we use the direct matching schema in our algorithm, instead of only considering the highlighted words given by search engines.

The following are the features we choose.

Factor 1: how often the full form appears in the title when searching for a candidate
We score this factor by taking the first 20 results of searching for the candidate abbreviation form, and counting the number of results for which the title contains the full form. The text containing the abbreviation usually also contains its full form. To avoid misjudge, if the candidate itself does not appear in the search results, its score will be set 0.

Factor 2: how often the candidate form appears in the title when searching for itself
We score this factor by searching for the candidate abbreviation, and counting the number of results whose the title contains the candidate. The popularity of the candidate form to some extent reflects how common it is in daily life. We find that misspellings may have impact on this factor. Therefore, we require the full form to appear in the title of all search results at least once, or the score will be set 0.

Factor 3: how often the full form appears in snippet when searching for a candidate
This factor considers the occurrence of the full form in search result snippets, which is similar to factor 1. The only difference here is that we consider snippets, instead of titles.

Factor 4: how often the candidate form appears in snippet when searching itself
Similar to factor 2, this factor considers the occurrence in search result snippets instead of titles, which serves as a validation for whether the candidate is a legal phrase.

Factor 5 and 6: how often the candidate appears in title and snippet when searching its full form
These factors are represented as factor 5 and 6, corresponding to title and snippet respectively. The two factors are complementary to factor 1 and 3, differing in whether one searches the candidate or the full form. These factors serve as verification for the candidate form in searching full form results, testing whether the candidate is a legal term.

Factor 7: comparing similarity between searching candidate and full form
We first use factor 7 to denote similarity between the titles of the first 20 results of searching a candidate and same amount of titles from searching its full form. For two titles, we say they are same only if they fully match with each other, which indeed is the case in search results.

Factor 8: search results count
This factor is scored by the total number of results returned by searching "full-form AND candidate". As far as we can see, more results when searching the full form and a candidate together indicate a stronger link between these two terms.

Factor 9: the co-occurrence of a candidate and its full form
This factor considers how often a candidate and its full form co-occur in results of searching "full-form candidate". The co-occurrence of the full form and a candidate will increase the probability for this candidate to be the true abbreviation.

Factor 10: matching forward syntactic patterns
We first define syntactic patterns such as "X简称Y" ("Y is short for X"). Then we score this factor by counting how many times the results of searching "full-form candidate" match these patterns.

The word "forward" means the full form appears ahead of the candidate.
Our pattern extraction algorithm is illustrated in TABLE 5. The GetSnippets function returns a list of snippets for the given joint query "full-form + candidate" for each pair (A, B) in S. For each snippet found by GetSnippets function, we replace the full form and the abbreviation with wildcards "X" and "Y". Then we use function GetNgrams to extract character n-grams for n = 2, 3, 4, 5, 6 and 7. The n-grams are guaranteed to contain exactly one X and one Y. We sort the n-grams by their frequency and select the top patterns. We then use these patterns to score candidates in the re-ranking phase. Some of the patterns we use are shown in TABLE 6

Algorithm 1 : ExtractPatterms()
• Initialize: Let S be a "Full form"-"Abbreviation" set. • Begin: • For each full-abbreviation $pairs(A,B) \in S$ Do $D \leftarrow GetSnippets(A,B)$ • For each snippet $d \in D$ Do $N \leftarrow N \cup GetNgrams(A,B,d)$ • $Patterns \leftarrow SortByFreq(N)$ • Return Patterns

Table 5: Algorithm for extract patterns.

• X（Y） • X（简称Y） • X(Y) • X（以下简称Y） • X简称Y，

Table 6: Forward patterns used.

Factor 11: matching backward syntactic patterns
The term "backward", in contrast to the previous "forward", means that the abbreviation appears in front of the full form. The algorithm to extract patterns is the same as factor 10. Some of the patterns we use are shown in TABLE 7.

• Y X • Y-X • Y（X） • Y和X是同义词 • Y是X的简称

Table 7: Backward patterns used.

4 Experiments

We use the abbreviation corpus provided by Institute of Computational Linguistics (ICL) of Peking University in out experiments. The corpus is homogeneous to the corpus used in (Sun et al., 2008, 2009). It contains 8,015 Chinese abbreviations. Various kinds of abbreviation pairs can be found in this corpus, including noun phrases, organization names and some other types. Some examples are presented in TABLE 8. The length distributions of full form and references are shown in FIGURE 2.

Type	Full form	Abbreviation
Noun Phrase	优秀稿件(Excellent articles)	优稿
Organization	作家协会(Writers' Association)	作协
Coordinate phrase	受伤死亡(Injuries and deaths)	伤亡
Proper noun	传播媒介(Media)	传媒

Table 8: Examples of the corpus (Noun Phrase, Organization, Coordinate Phrase, Proper Noun)

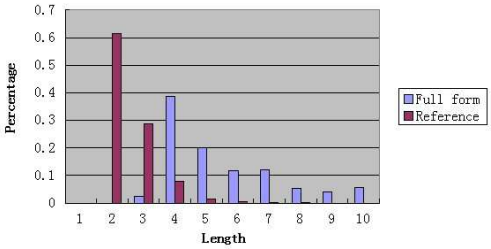


Figure 2: Length distribution of training set.

In some cases a long phrase may contain more than one abbreviation. For these cases, the corpus just keeps their most commonly used abbreviation one for each. Meanwhile to accurately get the results from the search engine that we need in our algorithm, we only keep the pairs with abbreviation containing more than 1 character, because the search results of a single Chinese character are usually ambiguous.

To improve the reliability of the experiment, we use 10 fold cross-validations. The evaluation metric used in our experiment is the top-k accuracy, which is also used by (Tsuruoka et al., 2005) and (Sun et al., 2009). The top-k accuracy measures what percentage of the reference abbreviations are found if we take the top N candidate abbreviations from all the results. In our experiment, top-10 candidates are considered in re-ranking phrase and the measurement used is top-1 accuracy because the final aim of the algorithm is to detect the exact abbreviation, rather than a list of candidates.

CRF++[2] and libsvm[3] , two open source tools, are used, with parameters are kept as default. The kernel function we use in our experiment is RBF kernel. All numeric values in SVM are scaled between 0 and 1. The generation of training examples for re-ranking considers the fact that a full form corresponds to a few candidate abbreviation forms, while only one of them is its reference. During the SVM process, we treat the reference as a positive instance, and treat the other false candidates as negative instances. Take the full form "北京大学" (Peking University) as an example. It corresponds to many candidate abbreviations like "北大", "京学". Only the reference "北大" is regarded as positive instance while the rest are negative. We then normalize the factors described in section 3 and use them together with the CRF score as features for each positive and negative instance in the re-ranking procedure.

The trained SVM classifier is then used in testing to give each candidate a label. For a given

[2]http://crfpp.sourceforge.net/
[3]http://www.csie.ntu.edu.tw/ cjlin/libsvm/

candidate, what we are interested in is not the label, but the probability it will be labeled as positive. In libsvm the probability is calculated based on the vertical distance to the hyper-plane[4] . We follow this schema. This probability is then used as the re-ranking standard and we select the top-ranked candidate as the final result.

For search engine, we use the search engine Baidu[5] in the re-ranking phase, which is the biggest Chinese search engine.

5 Results and Discussion

5.1 Comparison of re-ranking

TABLE 9 shows the top-10 accuracy of the candidate generation stage, which is the first stage of our method. We can see the top-10 candidates include the reference abbreviation for most full forms. The top 10 candidates already cover 92% of the reference abbreviations using BIEP labels. In theory if we can find web data to re-rank the candidates, as high as 92% accuracy can be achieved compared to the original 58% accuracy.

Top-K	1	2	3	5	10
Accuracy	0.5812	0.7293	0.7975	0.8652	0.9240

Table 9: Top-10 Accuracy of CRF-BIEP.

We then use search results to re-rank the top-10 candidates. After re-ranking we select the top-ranked candidate as the final abbreviation of each instance. TABLE 10 shows the results. We can see that the accuracy of our method is 64.25%, which improved by +6% compared to using sequence labeling models alone.

Method	Without re-rank	With re-rank
Top-1 accuracy	0.5812	0.6425

Table 10: Results of Chinese abbreviation generation after re-ranking.

We also compare our method with previous methods. The first two are $CRF + GI$ and $DPLVM + GI$ in (Sun et al., 2009). We compare our approach with another web-based method used in (Jain et al., 2007), which is slightly different from ours. The work in (Jain et al., 2007) focuses on extracting full-abbreviation pairs, rather than generating abbreviations from full forms. However, we think it is meaningful to compare because in both cases the web data is used only to extract the useful information lie between the full form and abbreviation, which is independent of the problem settings. This method is denoted as "$CRF + AEPW$" used point wise mutual information (PMI), popularity of the abbreviation and the pagerank of the URLs in search results as features and integrate these features by multiplying them all. We also compare with another approach denoted as "$CRF + MUL$" which also multiplies all the features described in section 3. We add this comparison to see whether the difference is made by the feature set, not the re-ranking model itself.

TABLE 11 shows the results of the comparisons.

[4]http://www.csie.ntu.edu.tw/ cjlin/libsvm/

[5]http://www.baidu.com

[6]DPLVM is a model that needs multiple random initializations to get closer to the global optimal point. So we did not apply cross-validation for DPLVM+GI.

Method	$CRF + AEPW$	$CRF + MUL$	$CRF + GI$	$DPLVM + GI$[6]	Our method
Top-1 Accuracy	0.5698	0.6039	0.5850	0.5990	0.6425

Table 11: Performance of different method.

While our method outperforms other methods, we surprisingly find that the CRF+AEPW slightly decreases performance compared to the pure CRF approach. The reason is that CRF+AEPW tries to extract information between well-formed full forms and also well-formed abbreviations. However, in the current Chinese abbreviation generation process, some ill-formed candidates may be generated, like include illegal terms and common phrases which are in fact substrings of the full form.

From CRF+MUL we can also find that simply multiplying the scores of each of the features does improve performance, however, the improvement is not as much as our approach. This indicates that our approach can better model the information extracted from search results than the simply treating the features equally. We measure what extent each feature contributes to the re-ranking process by adding one/two feature alone each time. For results see FIGURE3.

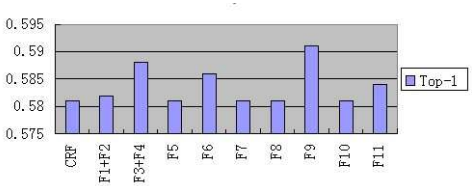


Figure 3: Contribution of each feature. The first column is the original sequence labeling score, which we use as a comparison.

FIGURE3 shows that feature 3+4, and 9 are the top contributing features. From feature 9 we can see that co-occurrence is indeed the most important factor. From feature 3 and 4 we can see that if the full form appears in the search results of a candidate, the candidate tends to be correct. This agrees to our intuition. If a candidate appears in the search results of the full form, it may happen to be a popular word as well as a substring of the full form. However, if the full form appears in the search results of a candidate, it means the full form does have strong link to the candidate.
We find that the re-ranking phase do play an important role in selecting the reference. Some reference abbreviations with low CRF scores can be reordered to the front after re-ranking. TABLE 12 shows the example of the organization name "阿拉伯国家联盟" (Arab League). The CRF score of its reference "阿盟" is low compared to other candidates, while after re-ranking, "阿盟" becomes the top-ranked candidate among all candidates.

TABLE 13, TABLE 14 and TABLE 15 show 3 more examples, which belong to different phrase types: noun phrase, coordinate phrase and proper noun. In all these cases, the references of the full form are picked out from the top 10 candidates. The results indicate that the re-ranking phrase can improve the performance of abbreviation generation.

Candidate	CRF Score	Re-rank Score
阿国联	0.427687	0.115977
阿联	0.182765	0.109433
国联	0.085203	0.0736369
阿盟(Reference)	0.053718	0.973178
阿国盟	0.043342	0.0225648
阿国联盟	0.032406	0.0361468
阿伯国联	0.021623	0.0213784
阿拉联	0.015541	0.0211315
阿联盟	0.013848	0.028979
阿拉伯国联	0.008748	0.0207346

Table 12: Generated abbreviations for Organization Name "阿拉伯国家联盟" (Arab League) and the correct re-ranking results.

Candidate	CRF Score	Re-rank Score
公共关	0.119895	0.0482089
公关系	0.099415	0.0365399
公共系	0.095923	0.0296014
公共	0.069083	0.036555
公系	0.058653	0.0250627
共关系	0.0545110	0.0604979
公关(Reference)	0.027417	0.96299
共关	0.015033	0.0450968
共系	0.012027	0.0284318
关系	0.001589	0.044727

Table 13: Generated abbreviations for Noun Phrase "公共关系" (Public Relation) and the correct re-ranking results.

Candidate	CRF Score	Re-rank Score
体医	0.522066	0.280765
体医疗	0.318698	0.0788524
体疗(Reference)	0.1497140	0.850495
体育医	0.003325	0.0244886
体育疗	0.001119	0.0303553
育医	0.001106	0.0234508
育医疗	$6.75E-4$	0.0270482
育疗	$3.72E-4$	0.0304597
医疗	$2.28E-4$	0.0430604
体育	$2.0E-5$	0.0394013

Table 14: Generated abbreviations for Coordinate Phrase "体育医疗" (Sports and Health) and the correct re-ranking results.

Candidate	CRF Score	Re-rank Score
物疗	0.344123	0.146159
物法	0.121928	0.0462008
物理疗	0.084886	0.0426191
物疗法	0.081885	0.0287058
物理法	0.073357	0.027162
理疗(Reference)	0.055539	0.906018
物理	0.050708	0.0491229
理法	0.047949	0.037015
理疗法	0.013221	0.054479
疗法	0.002413	0.0440872

Table 15: Generated abbreviations for Proper Noun "物理疗法" (Physiotherapy) and the correct re-ranking results.

5.2 Performance considering length

Long terms contain more characters, which is much easier to make mistakes during the sequence labeling phase. FIGURE 4 shows the top-1 accuracy respect to the term length using BIEP labeling method. The x-axis represents the length of the full form. The y-axis represents top-1 accuracy. We find that the search result based re-ranking method works especially well than pure CRF approach when the full form is long. By re-ranking using web data, additional information is incorporated. Therefore many of these errors can be eliminated. Meanwhile, if the reference

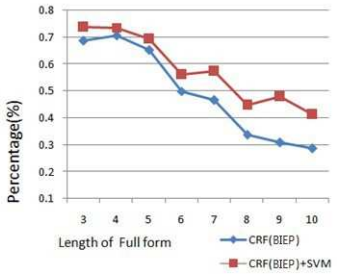


Figure 4: Accuracy grouped by length of full form.

itself is commonly used, the search results tend to contain more information of the relation between the candidate and the full form. With more information at hand, the re-ranking phase can make correct decisions with more confidence.

5.3 Error analysis

Though the accuracy is improved by +6% after we re-rank using search. There are still false candidates generated by the current method. We categorize the remaining errors as follows:

1. Candidates produced in the sequence labeling phase are only a portion of all possible combinations. Considering a full form with length 10, there are $2^{10} - 2 = 1022$ potential

candidates (the original term and empty string are omitted). Note that in our method we only take the top 10 candidates into consideration. If the references do not appear in the top 10 candidates, it is impossible for us to find the reference during the re-ranking phase. This kind of errors usually appears when the full form is very long. We find that this kind of error flourishes when the reference contains a Chinese word as its component. For example, "中央人民政府交通部" (Communication Department of the Central Government) is the full form for "中央交通部". The front "中央" (Central) and end "交通部" (Communication Department) are both Chinese words and appear continuously in the full form. Statistics show that this error makes up 13% of all errors.

2. Besides, character based candidate generation cannot use word level features, like word position. For example, if word "医院"(Hospital) is used in the middle of a full form, it is often abbreviated as "医", while in the end of a full form it will be abbreviated as "院". This is the shortcoming of al character based methods. However, we do not incorporate word information into our framework. As far as we investigate, previous works also seldom involve word information. The reason is that Chinese lacks natural word boundary, which cannot be segmented automatically with perfect accuracy. Current state-of-art Chinese word segmentation tools have at least 5% error rate, which can hurt consequent generation of abbreviations.
3. Search engines may provide biased information when handling location sensitive phrases. Take "香港民主同盟" (Democracy league of Hong Kong) as an example. Its correct abbreviation is "港同盟". Our method choose "民盟" as its abbreviation, which is the abbreviation of "中国民主同盟" (Democracy League of mainland China). Because the search engine we choose is Baidu.com, which is the most prevalent search engine in mainland China. Thus the number of search results related to "民盟" overwhelms that of "港同盟", with "民盟" 5200000 results compared with "港同盟" 13700 results. Besides when the web pages mentioning "民盟" (most of the pages are news pages), the "香港民主同盟" is always mentioned as well because there are homogeneous. Thus it is hard for the algorithm to exclude these interferences using localized search results. However, this kind of errors can be eliminated by using location-independent search engines.
4. Although some false candidates are not the standard reference, they are indeed used colloquially, only not as formally as the reference abbreviations. The reason for this phenomenon lies in the fact that the verification data we use is web search results. Web search results are sometimes colloquial, compared with official documents or other formal materials. Take "丁型病毒性肝炎"(Viral Hepatitis D) as an example, our method generates "丁肝", while the reference is "丁型肝炎". Both of these results are acceptable, while the reference is more formal.

Interestingly, we find that in this kind of errors, the "false" abbreviations are always shorter in length than the standard abbreviations, which is identical to the intuition that these abbreviations are more widely used orally.

6 Related work

Previous research on abbreviations mainly focuses on "abbreviation disambiguation", and machine learning approaches are commonly used (Park and Byrd, 2001; HaCohen-Kerner et al., 2008; Yu et al., 2006; Ao and Takagi, 2005). These ways of linking abbreviation pairs are effective, however, they cannot solve our problem directly because the full form is not always ambiguous. In many cases the full form is definite while we don’ t know the corresponding abbreviation.

To solve this problem, some approaches maintain a database of abbreviations and their corresponding "full form" pairs. The major problem of pure database-building approach is obvious. It is impossible to cover all abbreviations, and the building process is quit laborious. To find these pairs automatically, a powerful approach is to find the reference for a full form given the context, which is referred to as "abbreviation generation".

There is research on heuristic rules for generating abbreviations (Barrett and Grems, 1960; Bourne and Ford, 1961; Taghva and Gilbreth, 1999; Park and Byrd, 2001; Wren et al., 2002; HEARST, 2002). Most of them achieved high performance. However, hand-crafted rules are time consuming to create, and it is not easy to transfer the knowledge of rules from one language to another.

Recent studies of abbreviation generation have focused on the use of machine learning techniques. (Sun et al., 2008) proposed a supervised learning approach by using SVM model. (Tsuruoka et al., 2005; Sun et al., 2009) formalized the process of abbreviation generation as a sequence labeling problem. In (Tsuruoka et al., 2005) each character in the full form is associated with a binary value label y, which takes the value S (Skip) if the character is not in the abbreviation, and value P (Preserve) if the character is in the abbreviation. Then a MEMM model is used to model the generating process. (Sun et al., 2009) followed this schema but used DPLVM model to incorporate both local and global information, which yields better results.

While there are many statistical approaches, there are few approaches using Web as a corpus in machine learning approaches for generating abbreviations. Early examples like (Adar, 2004) proposed methods to detect such pairs from biomedical documents. Related work using web data includes (Liu et al., 2009; Jain et al., 2007). For example (Jain et al., 2007) used web search results as well as search logs to find and rank abbreviates full pairs, which show good result. But in fact search log data is only available in a search engine backend. In contrast, ordinary approach does not have access to search engine internals. Besides, they all use web data to expand the abbreviations to their full form, which is the opposite process of ours.

Conclusion and future work

To build an abbreviation dictionary, we used a stacked method to generate abbreviations from the full forms. We used sequence labeling method with BIEP labels to generate candidates for each full form, and used a SVM classifier which utilizes search results to re-rank the candidates to generate the final result.

The results are promising and outperformed the baseline methods. The accuracy can still be improved. Potential future works may include using semi-supervised methods to incorporate unlabeled data, or use more powerful methods to extract the characters of abbreviations in web data.

Acknowledgments

This work was partially supported by National High Technology Research and Development Program of China (863 Program) (No. 2012AA011101), National Natural Science Foundation of China (No.91024009, No.60973053), and the Specialized Research Fund for the Doctoral Program of Higher Education of China (Grant No. 20090001110047)

References

Adar, E. (2004). Sarad: A simple and robust abbreviation dictionary. *Bioinformatics*, 20(4):527–533.

Ao, H. and Takagi, T. (2005). Alice: an algorithm to extract abbreviations from medline. *Journal of the American Medical Informatics Association*, 12(5):576–586.

Barrett, J. and Grems, M. (1960). Abbreviating words systematically. *Communications of the ACM*, 3(5):323–324.

Bourne, C. and Ford, D. (1961). A study of methods for systematically abbreviating english words and names. *Journal of the ACM (JACM)*, 8(4):538–552.

Chang, J. and Schutze, H. (2006). Abbreviations in biomedical text.

HaCohen-Kerner, Y., Kass, A., and Peretz, A. (2008). Combined one sense disambiguation of abbreviations. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics on Human Language Technologies: Short Papers*, pages 61–64. Association for Computational Linguistics.

HEARST, A. (2002). A simple algorithm for identifying abbreviation definitions in biomedical text. In *Pacific Symposium on Biocomputing 2003: Kauai, Hawaii, 3-7 January 2003*, page 451. World Scientific Pub Co Inc.

Jain, A., Cucerzan, S., and Azzam, S. (2007). Acronym-expansion recognition and ranking on the web. In *Information Reuse and Integration, 2007. IRI 2007. IEEE International Conference on*, pages 209–214. IEEE.

Lafferty, J., McCallum, A., and Pereira, F. (2001). Conditional random fields: Probabilistic models for segmenting and labeling sequence data.

Liu, H., Chen, Y., and Liu, L. (2009). Automatic expansion of chinese abbreviations by web mining. *Artificial Intelligence and Computational Intelligence*, pages 408–416.

McCallum, A. (2002). Efficiently inducing features of conditional random fields. In *Proceedings of the Nineteenth Conference on Uncertainty in Artificial Intelligence*, pages 403–410. Morgan Kaufmann Publishers Inc.

Nenadić, G., Spasić, I., and Ananiadou, S. (2002). Automatic acronym acquisition and term variation management within domain-specific texts. In *Third International Conference on Language Resources and Evaluation (LREC2002)*, pages 2155–2162.

Pakhomov, S. (2002). Semi-supervised maximum entropy based approach to acronym and abbreviation normalization in medical texts. In *Proceedings of the 40th annual meeting on association for computational linguistics*, pages 160–167. Association for Computational Linguistics.

Park, Y. and Byrd, R. (2001). Hybrid text mining for finding abbreviations and their definitions. In *Proceedings of the 2001 conference on empirical methods in natural language processing*, pages 126–133.

Pinto, D., McCallum, A., Wei, X., and Croft, W. (2003). Table extraction using conditional random fields. In *Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval*, pages 235–242. ACM.

Platt, J. et al. (1999). Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. *Advances in large margin classifiers*, 10(3):61–74.

Sun, X., Okazaki, N., and Tsujii, J. (2009). Robust approach to abbreviating terms: A discriminative latent variable model with global information. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 2-Volume 2*, pages 905–913. Association for Computational Linguistics.

Sun, X., Wang, H., and Wang, B. (2008). Predicting chinese abbreviations from definitions: An empirical learning approach using support vector regression. *Journal of Computer Science and Technology*, 23(4):602–611.

Taghva, K. and Gilbreth, J. (1999). Recognizing acronyms and their definitions. *International Journal on Document Analysis and Recognition*, 1(4):191–198.

Tsuruoka, Y., Ananiadou, S., and Tsujii, J. (2005). A machine learning approach to acronym generation. In *Proceedings of the ACL-ISMB Workshop on Linking Biological Literature, Ontologies and Databases: Mining Biological Semantics*, pages 25–31. Association for Computational Linguistics.

Wren, J., Garner, H., et al. (2002). Heuristics for identification of acronym-definition patterns within text: towards an automated construction of comprehensive acronym-definition dictionaries. *Methods of information in medicine*, 41(5):426–434.

Yu, H., Kim, W., Hatzivassiloglou, V., and Wilbur, J. (2006). A large scale, corpus-based approach for automatically disambiguating biomedical abbreviations. *ACM Transactions on Information Systems (TOIS)*, 24(3):380–404.

Stacking Heterogeneous Joint Models of Chinese POS Tagging and Dependency Parsing

Meishan Zhang Wanxiang Che Ting Liu[*] *Zhenghua Li*
School of Computer Science and Technology, Harbin Institute of Technology, Harbin, China
{mszhang, car, tliu, lzh}@ir.hit.edu.cn

ABSTRACT
Previous joint models of Chinese part-of-speech (POS) tagging and dependency parsing are extended from either graph- or transition-based dependency models. Our analysis shows that the two models have different error distributions. In addition, integration of graph- and transition-based dependency parsers by *stacked learning* (stacking) has achieved significant improvements. These motivate us to study the problem of stacking graph- and transition-based joint models. We conduct experiments on Chinese Penn Treebank 5.1 (CTB5.1). The results demonstrate that the guided transition-based joint model obtains better performance than the guided graph-based joint model. Further, we introduce a constituent-based joint model which derives the POS tag sequence and dependency tree from the output of PCFG parsers, and then integrate it into the guided transition-based joint model. Finally, we achieve the best performance on CTB5.1, 94.95% in tagging accuracy and 83.98% in parsing accuracy respectively.

TITLE AND ABSTRACT IN CHINESE

采用堆方法融合异种的中文词性和依存句法联合模型

过去的中文词性和依存句法联合模型基本上都根据基于图的依存句法分析模型或者基于转移的依存句法分析模型进行拓展而形成的。我们的分析结果表明这两种不同的模型错误分布并不一样，而且在依存句法中，将基于图的模型和基于转移的模型使用堆方法融合之后，能够显著的提升依存句法的性能，这些促使我们进一步研究采用堆方法去融合基于图的和基于转移的词性依存句法联合模型。我们在中文宾州树库5.1版本（CTB5.1）上进行试验，实验结果表明，相比使用基于图的联合模型为被指导模型，采用转移的联合模型为被指导模型能取得较好的性能。更进一步，我们介绍了基于短语句法结构的联合模型，它从一个句子的概率短语文法分析器输出结果中提取句子的词性序列以及依存树结果，然后我们采用基于短语句法结构的联合模型更进一步指导基于转移的联合模型，最终我们在CTB5.1的数据上取得了最好结果，词性标注准确率达到94.95%，同时，依存句法准确率达到83.98%。

KEYWORDS: Chinese POS Tagging, Dependency Parsing, Joint Model, Stacked Learning.

KEYWORDS IN CHINESE: 中文词性标注,依存分析,联合模型,堆方法融合学习.

[*]Corresponding author

Proceedings of COLING 2012: Technical Papers, pages 3071–3088,
COLING 2012, Mumbai, December 2012.

1 Introduction

Part-of-speech (POS) tagging and dependency parsing are two fundamental natural language processing (NLP) tasks. Typically, POS tagging is a preprocessing step for dependency parsing, especially in a pipeline architecture. There are two main problems in a pipeline system: (1) Dependency parsing suffers the problem of error propagation; (2) POS tagging cannot exploit useful, important syntactic information for disambiguation.

For Chinese POS tagging and dependency parsing, a pipeline system seriously suffers these two problems. The study presented in (Li et al., 2011b) demonstrates the error propagation factor. The authors develop a graph-based joint model for Chinese POS tagging and dependency parsing. The most interesting thing they found is that even with lower tagging accuracy, a joint model could achieve higher parsing accuracy. The work presented in (Hatori et al., 2011) also demonstrates a joint model can largely improve the performance of dependency parsing further. They propose a transition-based joint model for Chinese POS tagging and dependency parsing.

Recently, ensemble models have been gained a lot of interests in NLP community. Stacked learning (stacking) (Wolpert, 1992; Breiman, 1996), which is a typical method for ensemble models, has been applied to a number of NLP tasks for its elegance and conciseness, such as Chinese Word segmentation (Sun, 2011), POS tagging (Li et al., 2011a), named entity recognition (Dekai Wu and Carpuat, 2003) and dependency parsing (McDonald, 2006; Nivre and McDonald, 2008; Martins et al., 2008; Søgaard and Rishøj, 2010; McDonald and Nivre, 2011). Especially, (Nivre and McDonald, 2008) demonstrate that the performance of dependency parsing can be largely improved by stacking a graph-based dependency parser and a transition-based dependency parser. Thus it is interesting to investigate the effect of stacked learning when it is applied to joint models.

Graph- and transition-based joint models are extended from graph- and transition-based models of dependency parsing respectively. They are the two mainstream approaches for dependency parsing. It is noteworthy that, the probabilistic context-free grammar (PCFG) parsers, such as Brown parser (Charniak and Johnson, 2005) and Berkeley parser (Petrov et al., 2006; Petrov and Klein, 2007), which are traditionally used for constituent parsing, have also been suggested for dependency parsing (Yamada and Matsumoto, 2003; McDonald, 2006; Sun, 2012; Che et al., 2012). We denote these methods by constituent-based models. The precondition of constituent-based models is that the output constituent structure of the PCFG parsers can be transformed into dependency structure by rules adequately. This is satisfied for Chinese. Moreover, the PCFG models can process POS tagging simultaneously in constituent parsing. They treat POS tagging as a submodule of constituent parsing. Thus we can also adopt a PCFG model for joint Chinese POS tagging and dependency parsing. We denote it by constituent-based joint model. (Sun, 2012) proposed to improve the Chinese parsing accuracy by a PCFG parser. Similarly, (Sun and Uszkoreit, 2012) exploited a PCFG parser to enhance Chinese POS tagging. Thus it is reasonable to investigate the performance of constituent-based joint models and to improve the performance of joint Chinese POS tagging and dependency parsing by a constituent-based joint model.

In this paper, first we study the integration of a graph-based joint model (JGraph) and a transition-based joint model (JTrans) by stacked learning. The stacked learning is implemented using a two-level architecture, where the level-0 consists of one or more predictors of which the results are exploited as input to enhance the level-1 predictor. Thus either JGraph or JTrans can be chosen as the level-1 model. We call the stacking model using JGraph as level-1 model

by the guided graph-based joint model and the stacking model using JTrans as level-1 model by the guided transition-based joint model. Further we introduce a constituent-based joint model (JConst) and then integrate this model into the previous better stacking model by further stacking.

We conduct the experiments on Chinese Penn Treebank 5.1 (CTB5.1) data set (Xue et al., 2005). First, we evaluate the performance of stacking models. The guided transition-based joint model gets better performance than the guided graph-based joint model, achieving 94.76% in tagging accuracy and 82.22% in parsing accuracy. Then we evaluate the performance of our constituent-based joint model. The reported accuracies are 93.45% in POS tagging and 81.03% in dependency parsing. Finally, we integrate the constituent-based joint model into the guided transition-based joint model by a further stacking. Our final results are 94.95% in tagging accuracy and 83.98% in parsing accuracy, resulting in further improvements of 0.19% in tagging accuracy and 1.76% in parsing accuracy.

Finally, detailed error analysis is carried out by two aspects: (1) the different error distributions of JGraph, JTrans and JConst are shown in detail to interpret the improvements in stacking and (2) the comparisons between joint models and pipeline approaches are conducted to understand the interaction between Chinese POS tagging and dependency parsing. Through the analysis, we can find several interesting phenomenons. For example, JConst can do better for long distance dependencies, the tagging accuracies of joint models are more fragile facing to wrong dependencies, dependencies with the head on the right are more easily recognized however the corresponding POS tags of the modifiers in these dependencies are more difficult to be handled.

The rest of the paper is organized as follows. Section 2 reviews the related works of our paper. Section 3 describes the graph- and transition-based joint models . Section 4 describes the stacking models including the guided graph-based joint model and the guided transition-based joint model. Section 5 describes our constituent-based model and the further stacking model. Section 6 reports the experimental results. Section 7 gives the systematic analysis of the joint models. Finally, in Section 8 we conclude this paper and point out our future works.

2 Related Works

Related Works on Joint Models of Chinese POS Tagging and Dependency Parsing (Li et al., 2011b) present the first joint model for Chinese POS tagging and dependency parsing. They extend models of graph-based dependency parsing (McDonald et al., 2005; McDonald and Pereira, 2006; Carreras, 2007; Koo and Collins, 2010), making them enable to handle POS tagging simultaneously. They conclude that joint models can achieve better performance in dependency parsing and can do much better some certain POS tagging error patterns.

Secondly, (Hatori et al., 2011) and (Bohnet and Nivre, 2012) propose joint models based on transition-based dependency parsing (Nivre, 2008; Huang and Sagae, 2010; Zhang and Nivre, 2011). (Hatori et al., 2011) also examine the results of their joint model carefully, demonstrating similar conclusions to that of (Li et al., 2011b). However, their certain error patterns are slightly different with that of (Li et al., 2011b). The differences may be induced by the different manners of modeling the joint task.

Thirdly, the constituent-based joint model processes joint Chinese POS tagging and dependency parsing in an indirectly way. It performs the POS tagging and dependency parsing by a conversion from the initial output of a PCFG parser. In Chinese POS tagging, (Sun and Uszkoreit, 2012) suggest to enhance Chinese POS tagging guided by the output POS tags of

Berkeley parser (Petrov et al., 2006; Petrov and Klein, 2007). In Chinese dependency parsing, (Sun, 2012) has proposed to improve the performance of dependency parsing by Berkeley parser. (Che et al., 2012) have compared the performance of several PCFG parsers on Stanford dependencies. This method has also been employed for English dependency parsing (Yamada and Matsumoto, 2003; McDonald, 2006).

In this paper, we study the integration of the three joint models by stacking. Then we discuss these different joint models including the stacking models comprehensively, aiming for two purposes: (1) figure out the benefits from stacking and (2) understand the interaction between Chinese POS tagging and dependency parsing.

Related Works on Stacked Learning *Stacked generalization* is a meta-learning algorithm that is first proposed by (Wolpert, 1992) and (Breiman, 1996). It has been exploited in a number of NLP tasks for integration. We mainly concern the works of stacked learning applied on POS tagging and dependency parsing. The work of (Li et al., 2011a) presented a mostly recent work for stacking POS taggers. They exploit the output of a CRF POS tagger to help a perceptron-based POS tagger with syntactic features. (McDonald, 2006) proposed the first stacking work of dependency parsing. The author incorporated parse decisions of two constituent-based parsers, Collins parser (Collins, 1999; Bikel, 2004) and Charniak parser (Charniak, 2000), into the second-order MST parser. Then (Nivre and McDonald, 2008) suggested integrating graph- and transition-based models by stacking, and more detailed analysis was given in (McDonald and Nivre, 2011). (Martins et al., 2008) also demonstrated that stacking transition- and graph-based parsers can improve parsing performance significantly and meanwhile offer theoretical interpretations for stacking. In our paper, stacked leaning is applied on the joint tasks of Chinese POS tagging and dependency parsing.

3 Two Models for Joint Chinese POS Tagging and Dependency parsing

A dependency tree for an input sentence $\mathbf{x} = w_0 w_1 \cdots w_n$ (where $w_0 = \text{ROOT}$) can be denoted by $\mathbf{d} = \{(h,m) \mid 0 \leq h \leq n, 0 < m \leq n\}$, where (h,m) represents a dependency $w_h \rightarrow w_m$ whose *head* word (or father) is w_h and *modifier* word (or child) is w_m. The task of dependency parsing is to find an optimum dependency tree $\mathbf{d}$ for the input sentence $\mathbf{x}$. Generally, the POS tag sequence of the sentence $\mathbf{t} = t_1 \cdots t_n$ (where $t_i \in T, 1 \leq i \leq n$, T is the POS tag set) is taken as an input for dependency parsing, which is determined by the task of POS tagging, thus forming a pipeline model of the two tasks. POS tagging is a typical sequence labeling problems which can be resolved by algorithms such as maximum-entropy (Ratnaparkhi, 1996), conditional random fields (CRF) (Lafferty et al., 2001) and averaged perceptron (Collins, 2002). The goal of joint models of the two tasks is to find an optimum dependency tree and an optimum POS tag sequence $(\hat{\mathbf{t}}, \hat{\mathbf{d}})$ for $\mathbf{x}$ concurrently.

3.1 Graph-based Joint Model

The graph-based joint model is first proposed by (Li et al., 2011b). Such a model is extended from a graph-based model for dependency parsing (McDonald et al., 2005; McDonald and Pereira, 2006; Carreras, 2007; Koo and Collins, 2010). In the model, the score of a dependency tree along with POS tags on each node is factored into scores of small parts.

(Li et al., 2011b) have introduced several different graph-based joint models in term of small parts employed in the models. According to the results they reported, we employ the model Li-11(v1,2nd) in terms of both accuracies and decoding speed. The scoring parts of the model

include dependencies, siblings and grandchilds, as is shown in Figure 1. The score function of Li-11(v1,2nd) can be represented by Equation 1,

$$\begin{aligned}\text{Score}_{\text{joint}}(\mathbf{x},\mathbf{t},\mathbf{d}) = &\sum_{\{(h,m)\}\subseteq\mathbf{d}} \mathbf{w}_{\text{pos}}\cdot\mathbf{f}_{\text{pos}}(\mathbf{x},\mathbf{t},m) + \mathbf{w}_{\text{dep}}\cdot\mathbf{f}_{\text{dep}}(\mathbf{x},\mathbf{t},h,m) + \sum_{\{(h,s)(h,m)\}\subseteq\mathbf{d}} \mathbf{w}_{\text{sib}}\cdot\mathbf{f}_{\text{sib}}(\mathbf{x},\mathbf{t},h,s,m) \\ &+ \sum_{\{(g,h)(h,m)\}\subseteq\mathbf{d}} \mathbf{w}_{\text{grd}}\cdot\mathbf{f}_{\text{grd}}(\mathbf{x},\mathbf{t},g,h,m)\end{aligned} \tag{1}$$

where $\mathbf{w}$ denotes the model parameters and $\mathbf{f}_{\text{pos}}(\cdot), \mathbf{f}_{\text{dep}}(\cdot), \mathbf{f}_{\text{sib}}(\cdot), \mathbf{f}_{\text{grd}}(\cdot)$ denote the features of POS tagging, dependency part, sibling part and grandchild part.

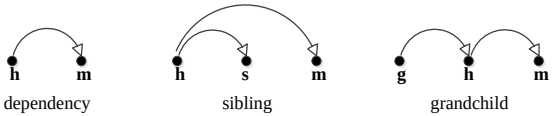


Figure 1: The scoring parts used in our graph-based joint model. Each node in the scoring parts has been tagged with POS already.

For more detailed description, we refer to the original paper. We call this graph-based joint model as JGraph for brevity.

3.2 Transition-based Joint Model

(Hatori et al., 2011) propose the first joint model of Chinese POS tagging and transition-based dependency parsing (Nivre, 2008; Huang and Sagae, 2010; Zhang and Nivre, 2011). In a transition-based system, we learn a joint model for scoring the transition A_i from one state ST_i to the next ST_j. As shown in Figure 2, the state ST of the transition system is composed by a stack S and a queue Q, where $S = (\cdots, s_1, s_0)$ is a stack of dependency trees along with POS tags and $Q = (q_0, q_1, \cdots, q_{n-j}) = (w_j, w_{j+1}, \cdots, w_n)$ is the remaining words which have not been processed at the current state.

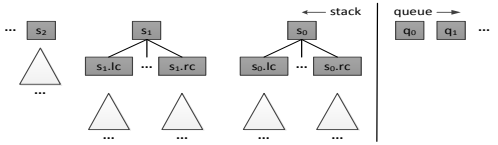


Figure 2: A state in the transition-based joint model. The figure is borrowed from (Huang and Sagae, 2010).

The candidate transition action A at each step is defined as follows:

- SHIFT(t) (SH(t)): move the head word w_j of queue Q into the stack S, assigning the word with the POS tag t.
- REDUCE-RIGHT (RR): merge the top two trees s_0, s_1 into a new subtree $s_0 \overset{\frown}{} s_1$.
- REDUCE-LEFT (RL): merge the top two trees s_0, s_1 into a new subtree $s_0 \overset{\frown}{} s_1$.

Equation 2 describes the score function of the transition-based joint model.

$$\text{Score}_{\text{joint}}(\mathbf{x},\mathbf{t},\mathbf{d}) = \sum_{A_i=\text{SHIFT(t)}} \mathbf{w}_{\text{pos}} \cdot \mathbf{f}_{\text{pos}}(\text{ST}_i, A_i, t) + \sum \mathbf{w}_{\text{syn}} \cdot \mathbf{f}_{\text{syn}}(\text{ST}_i, A_i) \quad (2)$$

where $\mathbf{f}_{\text{pos}}(\cdot)$ refers to the general features exploited in POS tagging and $\mathbf{f}_{\text{syn}}(\cdot)$ is all other features which are syntax related.

In this work, we employ the joint model of Joint-ZN^{-} in (Hatori et al., 2011) after considering the performance. We refer to their paper for more detailed descriptions. We thank the authors for sharing their code with us. We make some changes to make the basic POS tagging features the same with that in JGraph. We call this transition-based joint model as JTrans for brevity.

4 Stacking Joint Models

Stacked learning is a typical approach for model integration. The idea of stacked learning is to include two "level" of predictors : the level-0 includes one or more predictors $g_1, \cdots g_K (K \geq 1) : R^d \rightarrow R$ and the level-1 consists of one single predictor $h : R^{d+k} \rightarrow R$. Each predictor g_k of level 0 receives input $\mathbf{x} \in R^d$ and outputs a prediction$g_k(\mathbf{x})$. The level-1 predictor takes as input $\langle \mathbf{x}, g_1(\mathbf{x}), \cdots, g_K(\mathbf{x}) \rangle$ and outputs a final prediction $h(\mathbf{x}, g_1(\mathbf{x}), \cdots, g_K(\mathbf{x}))$.

In our work, we have two strategies for stacking JGraph and JTrans. The guided graph-based joint model exploits JGraph as the level-1 model and the guided transition-based joint model exploits JTrans as the level-1 model. We will describe them in the following respectively.

4.1 The Guided Graph-based Joint Model

The guided graph-based joint model, which we call JGraph(JTrans), exploits JGraph as the level-1 model and JTrans as level-0 model. As shown in Equation 1, the graph-based joint model computes a dependency tree along with POS tags by factored it into small parts including dependencies, siblings and grandchilds. Correspondingly, assuming the output of JTrans is $(\hat{\mathbf{t}}^{\text{JTrans}} = t_1^{\text{JTrans}} \cdots t_n^{\text{JTrans}}, \hat{\mathbf{d}}^{\text{JTrans}})$, the score function of JGraph(JTrans) becomes:

$$\begin{aligned}\text{Score}_{\text{joint}}(\mathbf{x},\mathbf{t},\mathbf{d}) = &\sum_{\{(h,m)\}\subseteq\mathbf{d}} \mathbf{w}_{\text{pos}} \cdot \mathbf{f}_{\text{pos}}(\mathbf{x},\mathbf{t},\hat{\mathbf{t}}^{\text{JTrans}},m) + \mathbf{w}_{\text{dep}} \cdot \mathbf{f}_{\text{dep}}(\mathbf{x},\mathbf{t},h,m,\hat{\mathbf{d}}^{\text{JTrans}}) \\ &+ \sum_{\{(h,s)(h,m)\}\subseteq\mathbf{d}} \mathbf{w}_{\text{sib}} \cdot \mathbf{f}_{\text{sib}}(\mathbf{x},\mathbf{t},h,s,m,\hat{\mathbf{d}}^{\text{JTrans}}) + \sum_{\{(g,h)(h,m)\}\subseteq\mathbf{d}} \mathbf{w}_{\text{grd}} \cdot \mathbf{f}_{\text{grd}}(\mathbf{x},\mathbf{t},g,h,m,\hat{\mathbf{d}}^{\text{JTrans}})\end{aligned} \quad (3)$$

We can see that the feature functions are modified to include additional arguments $\hat{\mathbf{t}}^{\text{JTrans}}$ and $\hat{\mathbf{d}}^{\text{JTrans}}$. Thus new features related with the additional arguments will be produced. These new features account for the guided features over the output of JTrans. The specific features used by JGraph(JTrans) are given in Table 1.

4.2 The Guided Transition-based Joint Model

The guided transition-based joint model, which we call JTrans(JGraph), exploits JTrans as the level-1 model and JGraph as level-0 model. As shown in Equation 1, the transition-based joint model computes a dependency tree along with POS tags by the sequence of transition actions of shaping it. Correspondingly, assuming the output of JTrans is $(\hat{\mathbf{t}}^{\text{JGraph}} = t_1^{\text{JGraph}} \cdots t_n^{\text{JoinG}}, \hat{\mathbf{d}}^{\text{JGraph}})$, the score function of JTrans(JGraph) becomes:

$$\text{Score}_{\text{joint}}(\mathbf{x},\mathbf{t},\mathbf{d}) = \sum_{A_i=\text{SHIFT(t)}} \mathbf{w}_{\text{pos}} \cdot \mathbf{f}_{\text{pos}}(\text{ST}_i, A_i, t, \hat{\mathbf{t}}^{\text{JGraph}}) + \sum \mathbf{w}_{\text{syn}} \cdot \mathbf{f}_{\text{syn}}(\text{ST}_i, A_i, \hat{\mathbf{d}}^{\text{JGraph}}) \quad (4)$$

The Guided Graph-based Joint Model: JGraph(JTrans)	
pos	$\{\hat{t}_m^{\text{JTrans}}, \hat{t}_m^{\text{JTrans}} \circ \hat{t}_{m-1}^{\text{JTrans}}, \hat{t}_m^{\text{JTrans}} \circ \hat{t}_{m+1}^{\text{JTrans}}, \hat{t}_{m-1}^{\text{JTrans}} \circ \hat{t}_{m+1}^{\text{JTrans}}\} \otimes \{t_m, w_m \circ t_m\}$
dep	{Whether $h \curvearrowright m$ is in $\hat{\mathbf{d}}^{\text{JTrans}}$?} $\otimes \{t_h, t_m, t_h \circ t_m\}$
sib	{Whether $h \curvearrowright m$ and $h \curvearrowright s$ are in $\hat{\mathbf{d}}^{\text{JTrans}}$?}$\otimes \{t_h, t_m, t_h \circ t_m\}$
grd	{Whether $g \curvearrowright h \curvearrowright m$ is in $\hat{\mathbf{d}}^{\text{JTrans}}$?}$\otimes \{t_h, t_m, t_h \circ t_m\}$
The Guided Transition-based Joint Model: JTrans(JGraph)	
pos	$\{t_m, w_m \circ t_m\} \otimes \{\hat{t}_m^{\text{JGraph}}, \hat{t}_m^{\text{JGraph}} \circ \hat{t}_{m-1}^{\text{JGraph}}, \hat{t}_m^{\text{JGraph}} \circ \hat{t}_{m+1}^{\text{JGraph}}, \hat{t}_{m-1}^{\text{JGraph}} \circ \hat{t}_{m+1}^{\text{JGraph}}\}$
syn	{Whether $s_0 \curvearrowright s_1$ is in $\hat{\mathbf{d}}^{\text{JGraph}}$?, Whether $s_0 \curvearrowleft s_1$ is in $\hat{\mathbf{d}}^{\text{JGraph}}$?} $\otimes \{s_0.t, s_1.t, s_0.t \circ s_1.t\}$, {Whether $s_0 \curvearrowright s_1$ is in $\hat{\mathbf{d}}^{\text{JGraph}}$?, Whether $s_0 \curvearrowleft s_1$ is in $\hat{\mathbf{d}}^{\text{JGraph}}$?} $\otimes$ {Whether $s_0 \curvearrowleft (s_0.\text{lc})$ is in $\hat{\mathbf{d}}^{\text{JGraph}}$?, Whether $s_0 \curvearrowright (s_0.\text{rc})$ is in $\hat{\mathbf{d}}^{\text{JGraph}}$?, Whether $s_1 \curvearrowleft (s_1.\text{lc})$ is in $\hat{\mathbf{d}}^{\text{JGraph}}$?, Whether $s_1 \curvearrowleft (s_1.\text{rc})$ is in $\hat{\mathbf{d}}^{\text{JGraph}}$?} $\otimes \{s_0.t, s_1.t, s_0.t \circ s_1.t\}$

Table 1: Guided features for JGraph(JTrans) and JTrans(JGraph). These features are defined by referring to (Li et al., 2011a) for POS related features and (Nivre and McDonald, 2008) for syntax related features. The symbol $\otimes$ denotes a cross join operation, and the symbol $\circ$ denotes a conjoin operation. In features of JTrans(JGraph), the index of m refers to the involved word for tagging, $x.t$ denotes the POS tag of word x, and x.lc and x.rc denote x's leftmost and rightmost child.

We can see that the feature functions are modified to include additional arguments $\hat{\mathbf{t}}^{\text{JGraph}}$ and $\hat{\mathbf{d}}^{\text{JGraph}}$. Thus new features related with the additional arguments will be produced. These new features account for the guided features over the output of JGraph. The specific features used by JTrans(JGraph) are given in Table 1.

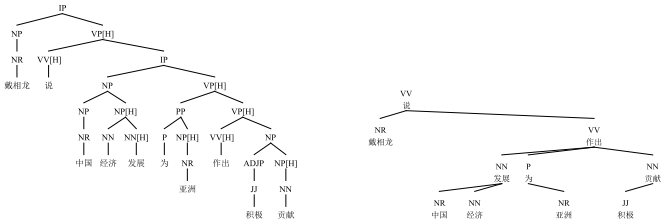
5 Constituent-based Joint Model and Further Stacking

5.1 Constituent-based Joint Model

Constituent-based joint models rely on certain PCFG parsers which exploit context-free grammar (CFG) to shape the search space for possible syntactic analysis. POS tagging and syntax analysis are usually processed simultaneously in PCFG parsers. These models have significant differences with the above two models. They model the joint POS tagging and dependency parsing in an indirect way. These models resolve the joint task by a two-step process: (1) constituent parsing and (2) rule-based transformation from constituent structures (CS) to dependency structures (DS). One advantage of constituent-based joint models is that all well-studied PCFG parsers can be used for the joint task. Figure 3 shows an example for this method.

In this work, we choose Berkeley parser[1] (Petrov et al., 2006; Petrov and Klein, 2007) to perform constituent parsing for its high performance in Chinese. We call this constituent-based joint model as JConst for brevity. Berkeley parser is an unlexicalized PCFG parser with latent variables. The observed constituent trees are automatically modeled with fined-grained unobserved constituent trees using latent variables. In Chinese POS tagging, (Sun and Uszkoreit, 2012) have proposed to use the output of Berkeley parser to enhance Chinese POS tagging. In Chinese dependency parsing, (Sun, 2012) has suggested to use Berkeley parser to improve the

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



(a) Constituent structure with head nodes marked already.

(b) Dependency structure.

Figure 3: An example of constituent-based joint model: 戴相龙说中国经济发展为亚洲作出积极贡献(Dai Xianglong said that Chinese economic development made positive contributions for Asia). The left constituent structure can be converted to the right dependency structure, and the head nodes (marked by [H]) are specified by CS-to-DS rules.

performance of parsing accuracy, and (Che et al., 2012) have demonstrate that Berkeley parser outperforms several other PCFG parsers for Chinese Stanford dependencies.

5.2 Stacking Constituent-based Joint Model as the Second Level-0 Model

As mentioned in Section 4, we have introduced the integration of JGraph and JTrans by stacked learning. Here we concern a more complex case to integrate JConst into the ensemble model of JGraph and JTrans. Actually, it can be carried out very simply by adding the features of JConst similar to the other level-0 model. For example, if we exploit the guided graph-based joint model to integrate JConst, the guided features related with JConst are similar to that related with JTrans, which are produced by replacing the output of JTrans with JConst.

By adding JConst for further stacking, there are two level-0 models in stacking. Besides the independent guided features of the two level-0 models, we suppose that the consistency of the two level-0 models is also an effective indicator. For example, in dependency parsing, if $w_h \frown w_m$ exists in both level-0 models, then the probability of adding such a dependency should be much higher, and if $w_h \frown w_m$ exists in only one of the level-0 models, the probability should be lower but also a positive contribution, and further if $w_h \frown w_m$ doesn't exist in anyone of the level-0 models, the probability should be much lower with a negative impact. We name the features related with the consistency of the two level-0 models as *guided consistent features*. Table 2 lists the guided consistent features used in this work. Both the guided graph-based joint model and the guided transition-based joint model are considered.

6 Experiments

6.1 Experimental Settings

We use CTB5.1 to conduct our experiments. Following the works of (Li et al., 2011b) and (Hatori et al., 2011), we use the standard split of CTB5.1 described in (Duan et al., 2007) and the conversion rules of CS-to-DS in (Zhang and Clark, 2008).

We use the standard tagging accuracy to evaluate POS tagging. For dependency parsing, we use word accuracy (also known as dependency accuracy or UAS), root accuracy and complete

The Guided Graph-based Joint Model: JGraph(JTrans, JConst)	
pos	{Whether $\hat{t}_m^{\text{JTrans}}$ is identical to $\hat{t}_m^{\text{JConst}}$?} $\otimes$ {$\hat{t}_m^{\text{JTrans}} \circ t_m, \hat{t}_m^{\text{JTrans}} \circ w_m \circ t_m$}
dep	{Whether the heads of m are identical in $\hat{\mathbf{d}}^{\text{JTrans}}$ and $\hat{\mathbf{d}}^{\text{JConst}}$?} $\otimes$ {Whether $h \curvearrowright m$ is in $\hat{\mathbf{d}}^{\text{JTrans}}$?} $\otimes$ {$t_h, t_m, t_h \circ t_m$}
The Guided Transition-based Joint Model: JTrans(JGraph, JConst)	
pos	{Whether $\hat{t}_m^{\text{JGraph}}$ is identical to $\hat{t}_m^{\text{JConst}}$?} $\otimes$ {$\hat{t}_m^{\text{JGraph}} \circ t_m, \hat{t}_m^{\text{JGraph}} \circ w_m \circ t_m$}
syn	{Whether the heads of s_0 are identical in $\hat{\mathbf{d}}^{\text{JGraph}}$ and $\hat{\mathbf{d}}^{\text{JConst}}$?, Whether the heads of s_1 are identical in $\hat{\mathbf{d}}^{\text{JGraph}}$ and $\hat{\mathbf{d}}^{\text{JConst}}$?} $\otimes$ {Whether $s_0 \curvearrowright s_1$ is in $\hat{\mathbf{d}}^{\text{JGraph}}$?, Whether $s_0 \curvearrowright s_1$ is in $\hat{\mathbf{d}}^{\text{JGraph}}$?} $\otimes$ {$s_0.t, s_1.t, s_0.t \circ s_1.t$}

Table 2: Guided consistent features.

match rate (all excluding punctuation) to evaluate the performance.

The models are trained iteratively and the best one is chosen for final evaluation for each joint model in terms of tagging accuracy and dependency accuracy on development set. The iterative number for graph-based joint models and transition-based joint models are 15 and 50 respectively. The beam size of our transition-based joint models is set to 64. In stacked learning, we split the train data into five folds to get augmented train data for level-1 model.

6.2 Stacking Graph- and Transition-based Joint Models

6.2.1 Baseline Performance

At first, we evaluate the performance of our baseline joint models: JGraph and JTrans. The performance of the pipeline models of JGraph and JTrans are also reported. Table 3 shows the results. The POS tagger of pipeline models is trained using averaged perceptron (Collins, 2002), exploiting the features which are only related with POS tagging. Our POS tagger achieves a tagging accuracy of 94.34% on development set, and 94.11% on test set, which is higher than the pipeline models of (Li et al., 2011b) and (Hatori et al., 2011). PGraph and PTrans denote the pipeline models of JGraph and JTrans respectively.

As shown in Table 3, both JGraph and JTrans achieve higher parsing accuracies than the pipeline models, where JGraph achieves increases of 1.36% and JTrans achieves increases of 1.61%. In tagging accuracy, JGraph achieve increases of 0.4% compared to PGraph, whereas JTrans suffers very little loss compared to PTrans. JGraph has a significant improvement in POS tagging compared to the reported results in their paper as they have enhanced the model before sharing the code for us. We use the code shared by the authors of (Hatori et al., 2011) to train JTrans, meanwhile some modifications have been done to make the basic POS tagging features identical with JGraph. However our parsing accuracy is slightly lower than the reported results in their paper. The work of (Bohnet and Nivre, 2012) is the only one other related work for joint Chinese POS tagging and dependency parsing. They also reported their results on CTB5.1 data set. However we didn't make a comparison as they employed a different conversion method for CS-to-DS.

		Syntactic Metrics			Tagging
		word	root	compl.	Accuracy
Stacking	JGraph(JTrans)	82.04	**78.17**	30.21	94.52
	JTrans(JGraph)	**82.22**	78.03	**30.58**	**94.76**
Graph	JGraph	**80.88**	**75.55**	**28.83**	**94.51**
	PGraph	79.52	75.34	26.70	94.11
	(Li et al., 2011b) (v1,2nd)	80.74	75.80	28.24	93.08
Transition	JTrans	**80.91**	**76.91**	**29.32**	94.07
	PTrans	79.30	75.73	27.80	**94.11**
	(Hatori et al., 2011)(ZN)	81.33	77.93	29.90	93.94

Table 3: The results of baseline models on test corpus. Results of significant test show that the p-value of tagging accuracy is lower than 10^{-3} and the p-value of parsing accuracy is lower than 10^{-5} for both JGraph(JTrans) and JTrans(JGraph).

6.2.2 Stacking Results

Table 3 shows the results of integration JGraph and JConst by stacked learning. We can see both the guided graph-based joint model and the guided transition-based achieve significantly improvements compared to the baseline models. The guided transition-based joint model obtains slightly better performance than the guided graph-based joint model, achieving the tagging accuracy of 94.76% and the parsing accuracy of 82.22%. (Nivre and McDonald, 2008) have done a similar work on dependency parsing. Their result demonstrate that the the guided graph-based model is better than the guided transition-based model in Chinese. Our results are different. This is perhaps caused by our baseline models are more complex, especially the transition-based joint model.

6.3 Stacking Constituent-based Joint Model as the Second Level-0 Model

We have shown that the guided transition-based joint model achieves better performance. Thus we integrate the constituent-based joint model into this guided model in our work.

6.3.1 Performance of the Constituent-based Joint Model

Table 5 shows the performance of JConst. The parsing accuracy of JConst is slightly higher than JGraph and JTrans. This demonstrates that JConst is also an effective method for joint Chinese POS tagging and dependency parsing. However, the tagging accuracy is not good. This may be caused by that Berkeley parser is a generative model, and thus relatively poor POS related features can be used in it.

6.3.2 Effectiveness of Guided Consistent Features

For further stacking JConst, we have suggested the guided consistent features. To test the effect of guided consistent features, we conduct feature ablation experiments. Table 4 shows the results on development set, where the mark {***-GC***} denotes the model without guided consistent features. The ablation of the guided consistent features resulted in 0.2% decreases in tagging accuracy and 0.28% decreases of parsing accuracy for JTrans(JGraph, JConst), showing the effectiveness of these features.

	Syntactic Metrics			Tagging
	word	root	compl.	Accuracy
JTrans(JGraph, JConst)	**84.68**	**80.38**	**34.37**	**95.31**
JTrans(JGraph, JConst){**-GC**}	84.40	79.82	34.12	95.11

Table 4: Feature ablation for guided consistent features.

6.3.3 Final Results

Table 5 shows the final results of stacking JGraph, JTrans and JConst together. The final stacking model JTrans(JGraph, JConst) achieves a tagging accuracy of 94.95% and a dependency accuracy of 83.98%, obtaining further increases of 0.19% in tagging accuracy and 1.76% in parsing accuracy compared to JTrans(JGraph). The improvements of JTrans(JGraph, JConst) compared to JTrans(JGraph) are larger than that of JTrans(JGraph) compared to JGraph or JTrans. In both tagging and parsing performance, JTrans(JGraph, JConst) is better than JTrans(JGraph), and meanwhile JTrans(JGraph) is better than anyone of {JGraph, JTrans, JConst} . It demonstrates that each individual joint model has a positive impact in the stacking.

	Syntactic Metrics			Tagging
	word	root	compl.	Accuracy
JTrans(JGraph, JConst)	**83.98**	**81.29**	**32.15**	**94.95**
JConst	81.03	78.12	28.01	93.45

Table 5: Final results of further stacking for constituent-based joint model.

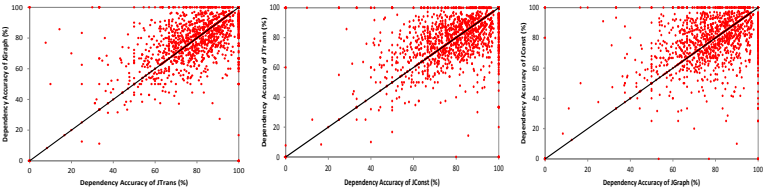
7 Analysis

The analysis is conducted on test corpus of CTB5.1. It includes two aspects. First we carefully examine the error distributions of the joint models, which can help us to figure out how stacked learning helps Chinese POS tagging and dependency parsing. Then we compare the joint models to pipeline models, which can help us to understand the interaction between Chinese POS tagging and dependency parsing.

7.1 Comparisons between Heterogeneous Joint Models

In this section, we mainly concern the performance of dependency parsing. Generally, stacking can perform effectively when the differences between baseline models are very large. For the final stacking model JTrans(JGraph, JConst), the baseline models include JGraph, JTrans, JConst.

First, we display the scatter plots of the parsing accuracies of JGraph against JTrans, JTrans against JConst and JConst against JGraph respectively. Each point (x, y) in the scatter plots denotes a sentence's parsing accuracy in the two joint models. If the point is upper the line $y = x$, it denotes that the joint model represented by vertical axis achieve higher dependency accuracy for the sentence. As is shown in Figure 4, the points seem to be random distributed in the three plots, and the number of points divided by line $y = x$ seems equal. These demonstrates that the error distributions of the three baseline models are rather different. Each model can process better on some sentences, and also may perform worse on some other sentences. By stacking, we can take advantage of the strengths of each model, thus resulting in a better performance.



(a) x:JTrans,y:JGraph (b) x:JConst,y:JTrans (c) x:JGraph,y:JConst

Figure 4: Scatter plots of dependency accuracies of every two baseline models.

Figure 5 shows the performance of dependency parsing in term of dependency length. Dependency length denotes the distance between the head and modifier in a dependency. Generally, dependency accuracies decrease when the distance between the head and modifier become longer. JConst can do better than the other two baseline models. JGraph and JTrans model dependency trees directly, and dependency length is a very important factor when building a dependency to shape the tree, thus JGraph and JTrans can be influenced much by it. However, JConst builds a dependency tree indirectly, the syntax analysis is done under the grammar of CFG, which needn't consider the attributes of a dependency tree. Thus dependency length has smaller influence to JConst. During the stacking, the model can learn that JConst is good at the dependencies with long dependency length. Thus the final stacking model can perform well for long dependency length, which is shown in Figure 5.

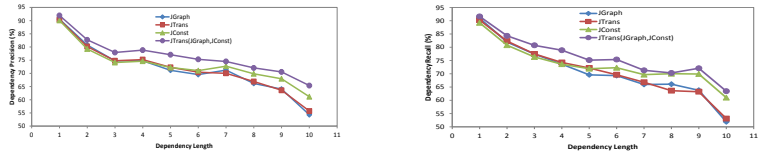


Figure 5: Dependency arc precision/recall relative to predicted/gold dependency length.

7.2 Comparisons between Joint Models and Pipeline Models

Impact on POS tagging : Intuitively, the tagging accuracy should be much higher if the dependency structures are correctly predicted. Table 6 shows the tagging accuracies of the different models with correct dependency heads and wrong heads respectively. The accuracies of words with correct heads are 10% higher than that with wrong heads on average. Joint models are more easily affected by the correctness of dependency head compared to PGraph[2]. This is caused by that we have exploited the syntactic features for POS tagging in joint models.

Next, we investigate which POS tagging error patterns can be influenced greatly in joint models. Figure 6 shows the related high frequency error patterns. The error patterns with rectangles upper the horizontal line are positive error patterns which joint models can do better, and the error patterns with rectangles below are negative error patterns which joint models can

[2]We only choose PGraph for comparison since it achieves better performance than PTrans.

	PGraph	JGraph	JTrans	JConst	JTrans(JGraph, JConst)
Correctly Headed	96.23	96.92↑	96.64↑	96.36↑	97.03↑
Wrongly Headed	86.65	85.34↓	84.16↓	81.79↓	84.7↓

Table 6: POS tagging accuracies with correct dependency heads and wrong dependency heads.

do worse. We can see that the joint models can do significantly better on such error patterns like NN → VV and DEC → DEG, and in contrast joint models can do worse on error patterns for example NR → NN and NN → JJ. Actually, the positive error patterns are important for dependency parsing and the positive error patterns usually needn't to be distinguished in dependency parsing, we will demonstrate it later.

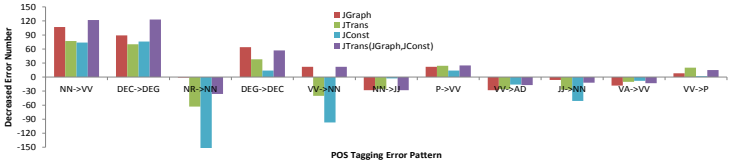


Figure 6: POS tagging error patterns influenced significantly by joint models.

Finally, the arc direction of dependency is also an important factor to influence the POS tagging accuracy. Table 7 shows the POS tagging accuracy of modifier words in term of gold arc direction $w_i \frown w_j$, $w_i \frown w_j$ $(i < j)$[3]. In Chinese, the POS tags such as NN, JJ or AD, which are difficult to be distinguished by dependency parsing, are usually tagged for modifier words in $w_i \frown w_j$, thus we can see that joint models gain little improvements for $w_i \frown w_j$.

	PGraph	JGraph	JTrans	JConst	JTrans(JGraph, JConst)
$w_i \frown w_j$	94.12	94.24	93.78	92.86	94.51
$w_i \frown w_j$	93.95	94.92	94.54	94.54	95.62

Table 7: POS tagging accuracies of modifier words in term of the gold arc direction.

Impact on dependency parsing : We mainly concern how POS tagging errors influence the performance of dependency parsing. In the previous discussion, we have demonstrated that dependency structures can influence some certain POS tagging error patterns significantly. We expect that the parsing accuracies should decrease sharply for the POS error patterns who demonstrate positive influences, because we suppose that these error patterns are harmful for dependency parsing, thus they can be distinguished by dependency parsing. Figure 7 shows the dependency accuracies of these POS tagging error patterns. We can see that the parsing accuracies of joint models for positive error patterns such as NN → VV, DEC → DEG and DEG → DEC drop drastically compared to PGraph. This conforms to our expectation. It demonstrates that joint models can do much better on certain POS tagging errors which are harmful for dependency parsing.

We also look into the impact of dependency arc direction on dependency parsing. Table 8 shows the dependency accuracies in term of gold arc direction. Generally, joint models do much better than PGraph in both $w_i \frown w_j$ and $w_i \frown w_j$. The accuracy of $w_i \frown w_j$ is significantly higher than $w_i \frown w_j$. One interpretation is that the words with POS tags such as NN, JJ or AD are usually

[3]We neglect the ROOT when considering the arc direction.

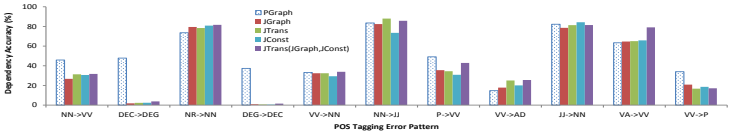


Figure 7: Dependency accuracies of different POS tagging error patterns.

easier to find their dependency head, and the head of such words is always on the right. Further, the accuracy gap of JConst between $w_i \frown w_j$ and $w_i \frown w_j$ is apparently smaller than the JGraph and JConst, which is similar to the phenomenon that dependency length has smaller influence to JConst, as dependency length and arc direction are important factors in JTrans and JConst which process dependency parsing directly.

	PGraph	JGraph	JTrans	JConst	JTrans(JGraph, JConst)
$w_i \frown w_j$	81.03	82.01	82.44	81.75	84.81
$w_i \frown w_j$	76.45	78.92	77.73	79.7	82.35

Table 8: Dependency accuracies in term of the gold arc direction.

Conclusions

In this paper, we present a study to investigate the integration of joint models of Chinese POS tagging and dependency parsing by stacked learning. First we integrate two joint models: JGraph and JTrans. We find that the stacking can improve the performance of joint models significantly. Meanwhile our experimental results demonstrate that the guided transition-based joint model can do better than the guided graph-based joint model. Next, we introduce a constituent-based joint model JConst, which employ a PCFG parser to get the result of constituent structure and then extract the results of POS tagging and dependency parsing from it. To further improve the performance of joint models, we integrate the JConst into the guided transition-based joint model by further stacking. The final stacking joint model achieves a POS tagging accuracy of 94.95% and a parsing accuracy of 83.98% in Chinese, resulting in total error reductions of 8% and 16% respectively compared to the best model of JGraph, JTrans and JConst.

Further, we conduct a detailed analysis aiming to figure out how stacked learning helps dependency parsing and POS tagging, and meanwhile aiming to understand the relationship between Chinese POS tagging and dependency parsing. We can find that JGraph, JTrans and JConst are very different in error distribution, joint models can do much better on certain POS tagging errors which are harmful for dependency parsing.

Acknowledgments

We especially thank Weiwei Sun for her suggestion of stacking constituent-based models in this work. This work was supported by National Natural Science Foundation of China (NSFC) via grant 61133012, the National “863” Major Projects via grant 2011AA01A207, and the National “863” Leading Technology Research Project via grant 2012AA011102.

References

Bikel, D. M. (2004). Intricacies of collins parsing model. *Computational Linguistics*, 30(4):479–511.

Bohnet, B. and Nivre, J. (2012). A transition-based system for joint part-of-speech tagging and labeled non-projective dependency parsing. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 1455–1465, Jeju Island, Korea. Association for Computational Linguistics.

Breiman, L. (1996). Stacked regressions. *Machine Learning*, 24:49–64.

Carreras, X. (2007). Experiments with a higher-order projective dependency parser. In *Proceedings of the CoNLL Shared Task Session of EMNLP-CoNLL*, pages 957–961.

Charniak, E. (2000). A maximum-entropy-inspired parser. In *Proceedings of the Second Meeting of North American Chapter of Association for Computational Linguistics (NAACL2000)*.

Charniak, E. and Johnson, M. (2005). Coarse-to-fine n-best parsing and maxent discriminative reranking. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 173–180, Ann Arbor, Michigan. Association for Computational Linguistics.

Che, W., Spitkovsky, V., and Liu, T. (2012). A comparison of chinese parsers for stanford dependencies. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 11–16, Jeju Island, Korea. Association for Computational Linguistics.

Collins, M. (1999). *Head-Driven Statistical Models for Natural Language Parsing*. PhD thesis, Pennsylvania University.

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

Dekai Wu, G. N. and Carpuat, M. (2003). A stacked, voted, stacked model for named entity recognition. In *Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003*.

Duan, X., Zhao, J., and Xu, B. (2007). Probabilistic models for action-based chinese dependency parsing. In Kok, J. N., Koronacki, J., de Mántaras, R. L., Matwin, S., Mladenic, D., and Skowron, A., editors, *Proceedings of ECML/ECPPKDD*, volume 4701 of *Lecture Notes in Computer Science*, pages 559–566. Springer.

Hatori, J., Matsuzaki, T., Miyao, Y., and Tsujii, J. (2011). Incremental joint pos tagging and dependency parsing in chinese. In *Proceedings of 5th International Joint Conference on Natural Language Processing*, pages 1216–1224, Chiang Mai, Thailand. Asian Federation of Natural Language Processing.

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

Koo, T. and Collins, M. (2010). Efficient third-order dependency parsers. In *Proceedings of the 48th Annual Meeting of the ACL*, number July, pages 1–11.

Lafferty, J. D., McCallum, A., and Pereira, F. C. N. (2001). Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *ICML*, pages 282–289.

Li, Z., Che, W., and Liu, T. (2011a). Improving chinese pos tagging with dependency parsing. In *Proceedings of 5th International Joint Conference on Natural Language Processing*, pages 1447–1451, Chiang Mai, Thailand. Asian Federation of Natural Language Processing.

Li, Z., Zhang, M., Che, W., Liu, T., Chen, W., and Li, H. (2011b). Joint models for chinese pos tagging and dependency parsing. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 1180–1191, Edinburgh, Scotland, UK. Association for Computational Linguistics.

Martins, A. F. T., Das, D., Smith, N. A., and Xing, E. P. (2008). Stacking dependency parsers. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 157–166, Honolulu, Hawaii. Association for Computational Linguistics.

McDonald, R. (2006). *Discriminative learning and spanning tree algorithms for dependency parsing*. PhD thesis, University of Pennsylvania.

McDonald, R., Crammer, K., and Pereira, F. (2005). Online large-margin training of dependency parsers. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics - ACL '05*, number June, pages 91–98, Morristown, NJ, USA.

McDonald, R. and Nivre, J. (2011). Analyzing and integrating dependency parsers. *Computational Linguistics*, 37(1):197–230.

McDonald, R. and Pereira, F. (2006). Online learning of approximate dependency parsing algorithms. In *Proceedings of EACL*.

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

Nivre, J. and McDonald, R. (2008). Integrating graph-based and transition-based dependency parsers. In *Proceedings of ACL-08: HLT*, pages 950–958, Columbus, Ohio. Association for Computational Linguistics.

Petrov, S., Barrett, L., Thibaux, R., and Klein, D. (2006). Learning accurate, compact, and interpretable tree annotation. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 433–440, Sydney, Australia. Association for Computational Linguistics.

Petrov, S. and Klein, D. (2007). Improved inference for unlexicalized parsing. In *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; Proceedings of the Main Conference*, pages 404–411, Rochester, New York. Association for Computational Linguistics.

Ratnaparkhi, A. (1996). A maximum entropy part-of-speech tagger. In *Proceedings of the Empirical Methods in Natural Language Processing*.

Søgaard, A. and Rishøj, C. (2010). Semi-supervised dependency parsing using generalized tri-training. In *Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010)*, pages 1065–1073, Beijing, China. Coling 2010 Organizing Committee.

Sun, W. (2011). A stacked sub-word model for joint chinese word segmentation and part-of-speech tagging. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 1385–1394, Portland, Oregon, USA. Association for Computational Linguistics.

Sun, W. (2012). *Learning Chinese Language Structures with Multiple Views*. PhD thesis, Saarland University.

Sun, W. and Uszkoreit, H. (2012). Capturing paradigmatic and syntagmatic lexical relations: Towards accurate chinese part-of-speech tagging. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 242–252, Jeju Island, Korea. Association for Computational Linguistics.

Wolpert, D. H. (1992). Stacked generalization. *Neural Networks*, 5:241–259.

Xue, N., Xia, F., Chiou, F.-D., and Palmer, M. (2005). The penn chinese treebank: Phrase structure annotation of a large corpus. *Natural Language Engineering*, 11(2):207–238.

Yamada, H. and Matsumoto, Y. (2003). Statistical dependency analysis with support vector machines. In *Proceedings of 8th International Workshop on Parsing Technologies (IWPT2003)*.

Zhang, Y. and Clark, S. (2008). A tale of two parsers: Investigating and combining graph-based and transition-based dependency parsing. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 562–571, Honolulu, Hawaii. Association for Computational Linguistics.

Zhang, Y. and Nivre, J. (2011). Transition-based dependency parsing with rich non-local features. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 188–193, Portland, Oregon, USA. Association for Computational Linguistics.

A Lazy Learning Model for Entity Linking Using Query-Specific Information

WeiZhang[1] *JianSu*[2]
ChewLimTan[1] *YunboCao*[3] *ChinYewLin*[3]
(1) National University of Singapore
(2) Institute for Infocomm Research, Singapore
(3) Microsoft Research Asia
{z-wei,tancl}@comp.nus.edu.sg, sujian@i2r.a-star.edu.sg,
{Yunbo.Cao,cyl}@microsoft.com

Abstract

Entity linking disambiguates a mention of an entity in text to a Knowledge Base (KB). Most previous studies disambiguate a mention of a name (e.g.*"AZ"*) based on the distribution knowledge learned from labeled instances, which are related to other names (e.g.*"Hoffman"*,*"Chad Johnson"*, etc.). The gaps among the distributions of the instances related to different names hinder the further improvement of the previous approaches. This paper proposes a lazy learning model, which allows us to improve the learning process with the distribution information specific to the queried name (e.g.*"AZ"*). To obtain this distribution information, we automatically label some relevant instances for the queried name leveraging its unambiguous synonyms. Besides, another advantage is that our approach still can benefit from the labeled data related to other names (e.g.*"Hoffman"*,*"Chad Johnson"*, etc.), because our model is trained on both the labeled data sets of queried and other names by mining their shared predictive structure.

Keywords: Entity Linking, Lazy Learning, Query-Specific Information.

Proceedings of COLING 2012: Technical Papers, pages 3089–3104,
COLING 2012, Mumbai, December 2012.

1 Introduction

Recently, more and more knowledge bases (KB) which contain rich knowledge about the world's entities such as Wikipedia [1], OpenCyc [2] and KIM [3] (Popov et al., 2004) have become available. These knowledge bases have been shown to form a valuable component for many natural language processing tasks such as knowledge base population (Ji and Grishman, 2011), text classification (Wang and Domeniconi, 2008), and cross-document coreference (Finin et al., 2009). However, to be able to utilize or enrich these KB resources, the applications usually require linking the mentions of entities in text to their corresponding entries in the knowledge bases, which is called entity linking task and has been proposed and studied in Text Analysis Conference (*TAC*) since 2009 (McNamee and Dang, 2009).

Given a mention of an entity in text and a KB, entity linking is to link the mention to its corresponding entry in KB. The major challenges of this task are name variation and name ambiguity. Name variation refers to the case that more than one name variation such as *alias, misspelling* and *acronym* refers to the same entity. For example, both *"48th State"* and *"The Grand Canyon State"* refer to *state of Arizona, U.S.*. Name ambiguity refers to the case that more than one entity shares the same name. For example, *"AZ"* may refer to *state of Arizona*, the Italian airline *Alitalia*, the country *Azerbaijan*, or other entries in KB that have the same name.

Most previous studies on entity linking used annotated data to learn a classifier or ranker (Dredze et al., 2010; Lehmann et al., 2010; Zheng et al., 2010; Zhang et al., 2010; Ploch, 2011; Ratinov et al., 2011) or to estimate parameters (Gottipati and Jiang, 2011; Han and Sun, 2011). Besides, from the analysis by Ji et al. (2011), all of the top systems from the participants in the shared task of *TAC-11*[4] used supervised learning approaches to solve this disambiguation problem.

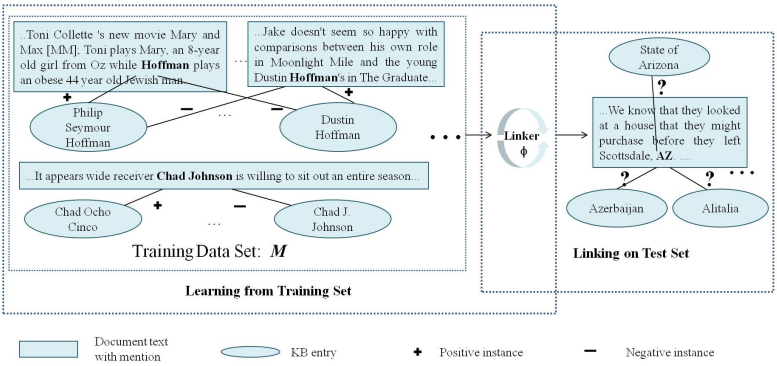


Figure 1: The System Architecture for Traditional Approaches. (*M* contains a certain number of names. *"Hoffman"* and *"Chad Johnson"* are two examples of them.)

However, as there are infinite number of entity names, it is impossible to manually create the labeled

[1] http://www.wikipedia.org/
[2] http://www.opencyc.org/
[3] http://www.ontotext.com/kim
[4] http://nlp.cs.qc.cuny.edu/kbp/2011/

data set for each name. The available labeled data for entity linking is only for a certain number of names. Thus, as shown in Figure 1, the previous approaches disambiguate a mention of the name (e.g.*"AZ"*) based on the distribution knowledge learned from the labeled mention-KB_entry pairs in the training set ***M*** related to other names (e.g.*"Hoffman"*,*"Chad Johnson"*, etc.). Given the query, the previous systems are static at evaluation time and their training does not depend on the input query. However, Figure 2 illustrates the distribution of the labeled instances related to the three names in a feature space (*Bag of Words*, *Named Entities* and *Edit Distance*, the popular features used in previous work). We can see that the width and location of the gap to separate positive and negative instances for different names vary widely. Moreover, the positive-negative instance ratio of each name is also very different from others. Thus, the entity linker generalized beyond labeled names (*"Hoffman"*, *"Chad Johnson"*, etc.) without considering the knowledge of the queried name (*"AZ"*) suffers from this distribution problem.

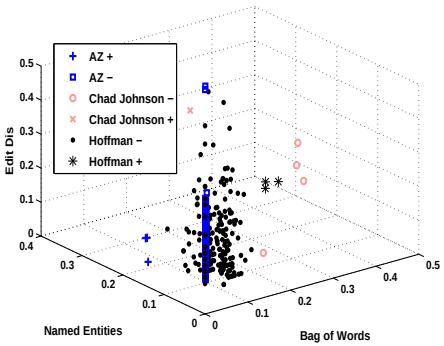


Figure 2: Instances Illustration in 3D Feature Space (Feature detail is in Table 2)

To narrow down the gap between the instance distributions related to labeled and queried names, this paper proposes a lazy learning model, in which generalization on the labeled data is delayed until a query is made. This allows the distribution information specific to queried name to be incorporated into the learning process. To obtain this distribution information, our lazy learning model automatically labels some relevant instances for the queried name leveraging its unambiguous synonyms.

In addition to the new notion of benefiting from the auto-generated instances related with the queried name, our approach further benefits from the manually labeled data related to other names. Specifically, the learned linker generalizes on the labeled data sets related to both queried and other names by exploiting the inherent predictive structure shared by these two data sets. We conduct evaluation on *TAC* data. Our experiments show that our proposed lazy learning model significantly improves entity linking over the state-of-the-art systems.

The remaining of this paper is organized as follows. In Section 2, we review the related work for entity linking. Section 3 elaborates the pre-processing stage to retrieve the possible KB entries

for a given mention. Section 4 presents our lazy learning for entity linking with query-specific information. Section 5 discusses a special case - NIL mentions . The experiments are shown in Section 6. Section 7 concludes the paper.

2 Related Work

As we have discussed, most of the previous entity linking work (Dredze et al., 2010; Lehmann et al., 2010; Zheng et al., 2010; Zhang et al., 2010; Ploch, 2011; Gottipati and Jiang, 2011; Han and Sun, 2011) fall into the traditional entity linking framework shown in Figure 1. Besides, the collaborative approach (Chen and Ji, 2011) tried to search similar queries as their query collaboration group by clustering texts. This differs from our method where we use the selective knowledge from unlabeled data.

In some other work, entity linking is also called *named entity disambiguation using Wikipedia* (Bunescu and Pasca, 2006; Cucerzan, 2007) or *Wikification* (Mihalcea and Csomai, 2007; Milne and Witten, 2008; Ratinov et al., 2011). These two similar tasks link expressions in text to their referent Wikipedia pages. However, since Bunescu and Pasca (2006) used Wikipedia hyperlinks to train the SVM kernel, Cucerzan (2007) used Wikipedia collection and news stories as the development data, and all of the three *Wikification* work (Mihalcea and Csomai, 2007; Milne and Witten, 2008; Ratinov et al., 2011) generalized their ranker on the data generated from Wikipedia without considering the knowledge of the queried name, they also fall into the traditional entity linking framework and suffer from the distribution problem in the previous entity linking systems. Thus, we believe that our proposed lazy learning approach also can benefit these two tasks.

In WePS-3 [5], a task of *Online Reputation Management* was proposed (Amigo et al., 2010; Spina et al., 2011), which is the same with entity linking when KB only has one entry. Given a set of Twitter entries containing an ambiguous company name, and given the home page of the company, the task is to filter out Twitter entries that do not refer the company. Amigo et al. (2010) concluded that it was not viable to train separate system for each of the companies, as the system must immediately react to any imaginable company name. Thus, in this benchmark, the set of company names in the training and test corpora are different. However, the lazy learning approach proposed in this paper demonstrates that it is feasible to train separate system for each company, and the system can immediately react to any company name without manually labeling new corpora.

More generally, resolving ambiguous names in Web People Search (WePS) (Artiles et al., 2007) and Cross-document Coreference (Bagga and Baldwin, 1998) disambiguates names by clustering the articles according to the entity mentioned. This differs significantly from entity linking, which has a given entity list (i.e. the KB) to which we disambiguate the mentions.

3 Candidate Generation

Because the knowledge base usually contains millions of entries, it is time-consuming to apply the disambiguation algorithm to the entire knowledge base. Thus, the following pre-processing process is conducted to filter out irrelevant KB entries and select only a set of candidates that are potentially the correct match to the given query (a query consists of a name mention and its associated document text).

Because of name variation problem, it is ineffective to retrieve entity candidates by comparing the name strings of mention and KB entry. Thus, we need to use external world knowledge to build the name variation list for each KB entry. Since our experiment used a KB derived from Wikipedia, and

[5] http://nlp.uned.es/weps/

other KBs such as *KIM* and *OpenCyc* usually can be mapped to Wikipedia (Nguyen and Cao, 2008), we find the variations for the entries in KB by leveraging name variation sources in Wikipedia: *"titles of entity pages"*, *"disambiguation pages* [6]*"*, *"redirect pages* [7]*"* and *"anchor texts"*. Then, the candidates can be selected by comparing the name string of the mention with the name strings in the variation list of each KB entry. The KB entry with a name string which matches the name of the mention is considered as a candidate. In addition, as a pre-processing step, we prefer a candidate set with high recall. Thus, to increase the recall, we find more candidates by selecting the KB entry if its name string contains the name string of the mention (e.g. *"Cambridge, Massachusetts"* contains *"Cambridge"*).

4 Lazy Learning for Linking

We formalize the disambiguation task as follows. We are given a query q (i.e. a document d_q with a mention m_q) and its associated KB candidates C_q={c_1,..,c_N} generated in Section 3, and our goal is to select the correct KB entry e from the set C_q. Specifically, let $\phi_q(q,c_i)$ be a score function reflecting the likelihood that the candidate c_i is the correct KB entry for q. Then, a disambiguation model is to solve the following optimization problem:

$$e = \arg\max_{c_i \in C_q} \phi_q(q, c_i) \tag{1}$$

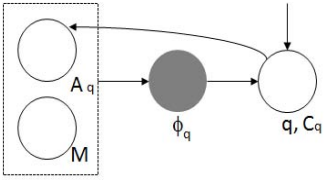


Figure 3: Lazy Learning Framework

In this section, we present our lazy learning model to solve this optimization problem. As shown in Figure 3, our process of the resolution is as follows:

1: Automatically label an instance set A_q based on the query q and candidates C_q.

2: Generalize a function ϕ_q on the data set A_q related with the queried name and a manually labeled data set M related with other names.

3: Select the correct KB entry e from the candidates using the function ϕ_q.

As shown in Figure 1, previous approaches generalize a universal linker ϕ for all of the queries on a labeled data set $\boldsymbol{M}$ related to irrelevant names (*"Hoffman"*, *"Chad Johnson"*, etc.), and they suffer from the distribution problem shown in Figure 2. In contrast, our lazy learning approach delays the model generalization until receiving the query. It can generalize a separate function ϕ_q for each query leveraging the distribution knowledge learned from the instances in A_q. As A_q is

[6] http://en.wikipedia.org/wiki/Wikipedia:Disambiguation
[7] http://en.wikipedia.org/wiki/Wikipedia:Redirect

automatically labeled for the queried name, it can be used to narrow down the gap of the instance distributions related to different names shown in Figure 2. Besides, ϕ_q also benefits from $\boldsymbol{M}$ in our model by mining its predictive information shared with A_q. Now, let us elaborate the method for generating A_q, and the generalization of ϕ_q, respectively.

4.1 Distribution Information A_q for Queried Name

In this section, we propose to obtain the distribution information for the queried name by automatically labeling some instances A_q for it. Following our work Zhang et al. (2010), which automatically generates training data for entity linking, we automatically label the instances related to the queried name based on its unambiguous synonyms.

Given a document d_q with a mention m_q and its associated KB candidates C_q, for example,

$\boldsymbol{d_q}$ *(*$\boldsymbol{m_q}$***="AZ")****: ...We know that they looked at a house that they might purchase before they left Scottsdale,* ***AZ****. ...*;

$\boldsymbol{C_q}$*: {*c_1*: state of Arizona,* c_2*: Azerbaijan, ...,* c_N*: Alitalia}*,

automatically creating the labeled set A_q for the name *"AZ"* requires automatically linking some mentions of *"AZ"* in text with the KB candidates in C_q. Our approach performs this linking based on two facts: (a) the title of the KB entry is unambiguous (e.g. *"state of Arizona"*). (b) The name variations of KB entry derived from *"redirect pages"* of Wikipedia in Section 3 are unambiguous (e.g. *"The Grand Canyon State"*). Then, we can generate the unambiguous name variation list for each candidate in C_q (see Table 1).

c_1	*state of Arizona; The Grand Canyon State; US-AZ; 48th State; AZ (U.S. state); The Copper State; Arizona, United States; ...*
c_2	*Azerbaijan; Azerbaidzhan; Republic of Azerbaijan; Azerbaijan Republic; Azerbaijani independence; Azerbaijan (Republic); ...*
...	
c_N	*Alitalia; Alitalia Airlines; Alitalia airways; Alitalia.it; Alitalia S.p.A.; ...*

Table 1: Unambiguous Variations for the Candidates of *"AZ"*

Because the unambiguous name only refers to one KB entry, we can link unambiguous name appearing in a document with the correct KB entry directly without human labor. Thus, we search the documents with these unambiguous name variations from a large document collection. Two examples of the retrieved documents are as below:

d_1 *(*m_1*="The Grand Canyon State"): ...* ***The Grand Canyon State*** *will get its shot to host the big game a year from now, ...*

d_2 *(*m_2*="Azerbaijan Republic"): ... It is located 30 km east of Ardebil and on the borderline with* ***Azerbaijan Republic****. ...*

We denote the labeled instance as a 4-tuple *(d,m,e,+1/-1)*, which means mention *m* in document *d* can/cannot be linked with KB entry *e*. Then, the two unambiguous examples above can be labeled as *(*d_1*,* m_1*,* c_1*, +1)* and *(*d_2*,* m_2*,* c_2*, +1)* automatically.

As we need to label the instances related to the name *"AZ"*, we further replace the unambiguous names in the documents with their ambiguous synonyms *"AZ"*. Then d_1 and d_2 are converted to:

d_1' *(m_q="AZ"): ...**AZ** will get its shot to host the big game a year from now, ...*

d_2' *(m_q="AZ"): ...It is located 30 km east of Ardebil and on the borderline with **AZ**. ...*

Finally, the labeled data set A_q for the queried name *"AZ"* is generated, where A_q=*{(d_1', m_q, c_1, +1),(d_1', m_q, c_2, -1), ...,(d_1', m_q, c_N, -1),(d_2', m_q, c_1, -1),(d_2', m_q, c_2, +1), ...,(d_2', m_q, c_N, -1) ...}*.

The data set A_q contains the query-specific information and such kind of information is not available in the training data M, such as the query-specific context features: words "*Airlines*" and "*State*" learned from A_q will be helpful for the disambiguation of "*AZ*"

4.2 Linear Function ϕ_q

In this section, we formulate the disambiguation function ϕ_q in Eq. 1 as follows,

$$\phi_q(q, c_i) = u^T \mathbf{X}_i \tag{2}$$

where the document d_q with a mention m_q and the candidate c_i in C_q are represented as a feature vector $\boldsymbol{X}_i \in \chi$, and u is a weight vector.

Estimate u on A_q. A popular method for finding u is *empirical risk minimization* with *least square regularization*. In this work, given a training set A_q=*{(d_i, m_q, e_i, Y_i)}*$_{i=1,...,n^{(q)}}$ ($Y \in$ *{+1,-1}*) related to the queried name m_q, firstly we transfer the instance *(d_i, m_q, e_i)* to the feature vector $\boldsymbol{X}_i^q$. Then, A_q=*{($\boldsymbol{X}_i^q$, Y_i^q)}*$_{i=1,...,n^{(q)}}$, ($\boldsymbol{X} \in \chi$, $Y \in$ *{+1,-1}*). Finally, we aim to find the weigh vector u that minimizes the empirical loss on the training data,

$$\hat{u} = \arg\min_u \left(\frac{1}{n^{(q)}} \sum_{i=1}^{n^{(q)}} L(u^T \mathbf{X}_i^q, Y_i^q) + \lambda \|u\|^2 \right) \tag{3}$$

where L is a loss function. We use a modification of the Huber's robust loss function: $L(p, y) = (max(0, 1 - py))^2$, if $py \geq$ *-1*; and *-4py* otherwise. We fix the regularization parameter λ to 10^{-4}.

Transfer *(q,c_i)* to Feature Vector $\boldsymbol{X}$. The features we adopted to construct $\boldsymbol{X}_i$ from *(q,c_i)* include 13 typical feature types used in *TAC* (Lehmann et al., 2010; Ji and Grishman, 2011), and the features are divided into four groups, contextual features (**CF**), semantic features (**SeF**), surface features (**SuF**) and generation source (**GS**) (see Table 2).

4.3 Incorporate M to u Estimation

A practical issue that arises in estimating u only on A_q is the paucity of labeled instances for some queries. This is because we automatically label the instances A_q leveraging its unambiguous synonyms (see Section 4.1). However, for some queried names, it is hard to find a sufficient number of unambiguous synonyms or the related documents containing these synonyms. On the other hand, the total number of available manually labeled instances M for other irrelevant names is relatively large. To illustrate the role of M in learning, consider the disambiguation of the two mentions "CPC" and "NY" in two documents. If the first mention "CPC" refers to entity "Communist Party

Name	Description
*Contextual Features (**CF**)*	
Bag of Words	The cosine similarity (tf.idf weighting) between query document and text of the candidate.
Similarity Rank	The inverted cosine similarity rank of the candidate in the candidate set.
Co-occurring NEs	Number of the same named entities appearing in query document and the text of the candidate.
*Semantic Features (**SeF**)*	
NE type	True if NE type (i.e. Person, GPE, Organization) of the query and the candidate is consistent.
Topic Similarity	Topic similarity between query document and text of candidate obtained by LDA (Blei et al., 2003)
*Surface Features (**SuF**)*	
Surface Match	True if the query matches the title of the candidate
Substring Match	True if the title of the candidate begins with the query (e.g. *"Beijing China"* and *"Beijing"*)
Acronym	True if the query is an acronym for the title of the candidate (e.g. *"NY"* and *"New York"*)
Word Match	Number of the same words between the title of the candidate and the query
Word Miss	Number of the different words between the title of the candidate and the query
Edit Distance	Levenshtein distance between name strings of the query and the candidate's title
*Generation Source (**GS**)*	
Wikipedia Source	True for each Wikipedia source (i.e. "entity pages", "disambiguation pages", "redirect pages" and "anchor texts" (Section 3)) which generates the candidate
String Match	For the candidate not generated from Wikipedia source, true if it is generated from full match.

Table 2: Feature Set for Disambiguation.

of China" and the second mention "NY" refers to entity "the city of New York", they have similar surface features (e.g. feature "acronym" is true), certain surface features effective for linking to "Communist Party of China" may be also effective for disambiguating "NY", and vice versa

However, with the gap in other aspects between the distributions of A_q and M shown in Section 1, directly adding M to our training set will produce a lot of noise with respect to the queried name. Thus, instead of using all the distribution knowledge in M, we propose to only incorporate the shared knowledge with A_q from M into u estimation based on *structural learning*.

The Structural Learning Algorithm. *Structural learning* (Ando and Zhang, 2005b) is a multi-task learning algorithm that takes advantage of the low-dimensional predictive structure shared by multiple related problems. Let us assume that we have K prediction problems indexed by $l \in \{1,..,K\}$, each with $n^{(l)}$ instances $(\mathbf{X}_i^l, \mathrm{Y}_i^l)$. Each $\boldsymbol{X}_i^l$ is a feature vector of dimension p. Let Θ be an orthonormal $h \times p$ (h is a parameter) matrix, that captures the predictive structure shared by all the

K problems. Then, we decompose the weight vector $\boldsymbol{u}_l$ for problem l into two parts: one part that models the distribution knowledge specific to each problem l and one part that models the common predictive structure,

$$\mathbf{u}_l = \mathbf{w}_l + \Theta^T \mathbf{v}_l \tag{4}$$

where $\mathbf{w}_l$ and $\mathbf{v}_l$ are weight vectors specific to each prediction problem l. Then, the parameters Θ, $\mathbf{w}_l$ and $\mathbf{v}_l$ can be learned by *joint empirical risk minimization*, i.e., by minimizing the joint empirical loss of the predictors for the K problems on the training instances as Eq. 5,

$$\underset{\Theta,\mathbf{w}_l,\mathbf{v}_l}{\arg\min} \sum_{l=1}^{K} \left(\frac{1}{n^{(l)}} \sum_{i=1}^{n^{(l)}} L\left(\left(w_l + \Theta^T v_l \right)^T \mathbf{X}_i^{(l)}, Y_i^{(l)} \right) + \lambda \|w_l\|^2 \right) \tag{5}$$

It shows that $\mathbf{w}_l$ and $\mathbf{v}_l$ are estimated on $n^{(l)}$ training instances of problem l. In contrast, Θ is estimated on all the training instances of the K problems. This is the key reason why structural learning is effective for learning the predictive structure shared by multiple prediction problems.

Alternating Structure Optimization. The Θ optimization problem in Eq. 5 can be approximately solved by the following alternating structure optimization procedure (Ando and Zhang, 2005a),

> 1: Learn K weight vectors u_l' for all the K problems on their corresponding instances independently using *empirical risk minimization* (similar with Eq. 3).
>
> 2: Let $U' = [u'_1, \ldots u'_K]$ be the $p \times K$ matrix formed from the K weight vectors.
>
> 3: Perform Singular Value Decomposition on U':$U' = V_1 D V_2^T$. The first h column vectors of V_1 are stored as rows of $\hat{\Theta}$

Structural Learning for Entity Linking: Incorporate *M* to *u* Estimation. As previous entity linking systems do not consider the information of the queried name, they usually use all the instances in M without any difference to train the linker. However, in data set M, some instances related with some particular names may share more predictive information with the queried name than other instances. Thus, in this work, we group the instances in M based on the "name", and then learn the shared information from the "name" group instead of individual instance. As shown in Figure 1, the data set M for entity linking usually has a certain number of names (e.g. *"Hoffman"*, *"Chad Johnson"*, etc.), each with some labeled instances. Then, we treat each "name" and its associated instances in M as a prediction problem of *structural learning*. Besides, the queried name (e.g. *"AZ"* in Figure 1) with auto-labeled instances A_q is our target prediction problem, which is the problem we are aiming to solve.

According to the applications of *structural learning* in other tasks, such as WSD (Ando, 2006), *structural learning* assumes that there exists a predictive structure shared by multiple related problems. In order to learn the predictive structure Θ shared by M and A_q, we need to (a) select relevant prediction problems (i.e. relevant names) from M. That is, they should share a certain predictive structure with the target problem; (b) select useful features from the feature set shown in Table 2. The relevant prediction problems may only has shared structure with target problem over certain features. In this paper, we use a set of experiments including feature split and data set M

partitioning to perform these two selection processes. This empirical method for selection will be elaborated in Section 6.3.

Let us assume that we have selected relevant names from data set M, which together with the queried name can be used as the K related prediction problems in *structural learning*. Applying *structural learning* to the K problems, we can obtain the shared structure $\hat{\Theta}$ by *alternating structure optimization*. Then, the weight vector $\boldsymbol{u}$ for the queried name in Eq. 2 can be approximately solved by the following procedure:

1: Learn $\hat{w}$ and $\hat{v}$ for the queried name by minimizing the empirical risk on data set A_q:

$$\arg\min_{w,v} \left(\frac{1}{n^{(q)}} \sum_{i=1}^{n^{(q)}} L\left((w + \hat{\Theta}^T v)\mathbf{X}_i^q, Y_i^q \right) + \lambda \|w\|^2 \right)$$

2: The estimated weight vector u for the queried name is:

$$\hat{u} = \hat{w} + \hat{\Theta}^T \hat{v}$$

The $\hat{\Theta}^T \hat{v}$ part is learned from the selected names in M and all the instances in A_q, and therefore it can model the shared predictive structure between M and A_q, and remove the noises in M as we expected. The $\hat{w}$ part is learned from the data set A_q, which can tackle the distribution problem (see Figure 2) in the previous work only using M.

5 Predicting NIL Mentions

So far we have assumed that each mention has a correct KB entry; however, when we run over a large corpus, a significant number of entities will not appear in the KB. In this situation, the document d_q with mention m_q should be linked to NIL. Traditional approaches usually need an additional classification step to resolve this problem (Zheng et al., 2010; Lehmann et al., 2010). In contrast, our approach seamlessly takes into account the NIL prediction problem. As we define $Y \in \{+1,-1\}$ to denote whether the pair of the mention and KB entry can be linked together, the median 0 can be assigned to $\phi_q(q,NIL)$. Then Eq. 1 is extended to:

$$e = \arg\max_{c_i \in C_q \cup NIL} \phi_q(q, c_i) \tag{6}$$

6 Experiments and Discussions

6.1 Experimental Setup

In our study, we use *TAC-10*[8] KB and document collection to evaluate our approach for entity linking. The KB is derived from Wikipedia, which contains 818,741 different entries and the document collection contains 1.7 million documents from newswire and blog text. Each KB entry consists of the Wikipedia Infobox [9] and the corresponding Wikipedia page text.

[8] http://nlp.cs.qc.cuny.edu/kbp/2010/

[9] http://en.wikipedia.org/wiki/Template:Infobox

The test set of *TAC-10* has 2,250 mentions across three named entity (**NE**) types: Person (**PER**), Geo-Political Entity (**GPE**) and Organization (**ORG**). The documents containing these mentions are from the document collection above. The training set of *TAC-10* consists of 5,404 mentions. Among them, 3,404 mentions are used as the data set M in our approach and the remaining 2,000 mentions are used as development set in our experiments.

We adopt micro-averaged accuracy officially used in *TAC-10* evaluation for our experiments, i.e. the number of correct links (including *NIL*) divided by the total number of the mentions.

6.2 Statistics of Data Set A_q

To minimize the distribution gap discussed in Section 1, we incorporate the distribution knowledge learned from A_q to the learning process. Thus, one of the key factors for the success of our lazy learning model is whether we can obtain A_q for the queries.

Therefore, firstly we investigate the amount of the labeled instances created for each query. When our model runs over the test data set, we find that 359 queries are assigned empty candidate sets (i.e. $C_q = \emptyset$) by the process described in Section 3. For these queries, we can directly link them with *NIL* without disambiguation. Thus, we only need to create A_q for the remaining 1,891 queries.

Figure 4 compares the proportions of the queries in different A_q size ranges. It shows that we have successfully created non-empty A_q for 96% of the 1,891 queries. This proves that our approach learning the distribution knowledge for the queried name from the automatically labeled instances A_q is feasible in practice. This also supports our assumption about the existence of the document with unambiguous synonyms in the document collection.

We also note that 49% of the queries have 10 to 99 labeled instances in A_q and 37% have 100 to 999 instances for each linker . In contrast, previous approaches usually trained their model on thousands of labeled instances. Thus, it suggests that we need more labeled instances for some queries and it is necessary to still leverage the manually labeled data set M in our learning process.

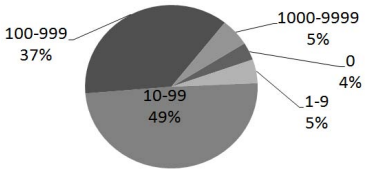


Figure 4: Proportions of the Queries Based on the Sizes of their Corresponding A_q

6.3 Exploring Θ Configuration

Because our lazy learning model generalizes on both the distribution knowledge learned from A_q and the predictive structure Θ shared by A_q and M, the effectiveness of such shared structure Θ is another key factor for the success of our lazy learning model. Thus, inspired by the work (Ando, 2006) for WSD, we design a set of experiments to investigate the configuration of Θ.

Consider the disambiguation of the two mentions *"CPC"* and *"NY"* in two documents again. They have similar surface features (e.g. feature *"acronym"* is true). The surface features effective for linking to *"Communist Party of China"* may be also effective for disambiguating *"NY"* to "the

city of New York", and vice versa. However, with respect to the semantic features, these two disambiguation problems may not have much in common. This is because *"Communist Party of China"* is likely related with the topic "politics", but *"the city of New York"* does not have such particular topic. That is, shared structure Θ between different names may depend on feature types, and in that case, seeking Θ for each of feature groups (*CF, SeF, SuF* and *GS* in Table 2) separately may be more effective. Hence, we experimented with both Θ configuration in Eq. 5 and Θ configuration, learning a Θ_j for each feature group j separately in Eq. 7.

$$\sum_{l=1}^{K}\left(\frac{1}{n^{(l)}}\sum_{i=1}^{n^{(l)}}L\left(w_l^T\mathbf{X}_i^{(l)}+\sum_{j\in F}v_l^{(j)\,T}\Theta_j\mathbf{X}_i^{(l,j)},Y_i^{(l)}\right)+\lambda\|w_l\|^2\right) \tag{7}$$

where F is a set of disjoint feature groups, and $\boldsymbol{X}^{(j)}$ (or $v^{(j)}$) is a portion of the feature vector $\boldsymbol{X}$ (or weight vector v) corresponding to feature group j, respectively.

The NE types of the instances in A_q and M are *PER, GPE* and *ORG*. Intuitively, the predictive structures of the names with the same NE type may be more similar than those of cross-NE-type names. Therefore, except for the feature split discussed above, we explore another two Θ configurations. One learns Θ from A_q and the whole M for each query. The other learns Θ from A_q and the subset of M, where the instances have the same NE type with the query.

Thus, we experimented on our development data set with the combinations of the two types of Θ configuration, i.e. configuration of feature split F and configuration for partitioning of data set M.

Figure 5 compares the performance using the various Θ configurations, and the results are in line with our expectation. *F={CF+SeF+SuF+GS}* treats the features of these four types as one group. It is equivalent to the Θ configuration without feature split in Eq. 5. Comparison of *F={CF, SeF, SuF, GS}* (learning Θ_j for these four feature groups separately by Eq. 7) and *F={CF+SeF+SuF+GS}* indicates that use of the feature split indeed improves disambiguation performance. We are also interested in whether all the feature groups are suitable for learning Θ_j. Thus, we further experimented with *F={SeF, SuF, GS}*, *F={CF, SuF, GS}*, *F={CF, SeF, GS}* and *F={CF, SeF, SuF}*. Figure 5 shows that these different subsets of feature groups do not improve the performance over using all the feature groups, and it proves that all the feature groups contribute to the learning of Θ. Besides, this figure also shows that learning Θ from A_q and the subset of M (i.e. instances have the same NE type with the query) usually performs better than learning it from A_q and the whole M. At last, as Θ has one parameter - its dimensionality h, the performance shown in this figure is the ceiling performance on the development set obtained at the best dimensionality (in {10, 50, 100,...}).

6.4 Evaluation Results for Lazy Learning

The experiments in this section evaluate our lazy learning model on the test data set of *TAC-10*. Our experiments used the best dimensionality $h = 150$ of Θ tuned on the development set in Section 6.3.

Table 3 shows the performances of three baseline methods and our approach with overall accuracy as well as accuracy on five subsets of the test set.

The second row (*M (Eq.3)*) used *empirical risk minimization* to estimate the weight vector u on the data set M (similar with Eq. 3). The third row (*M(SVM)*) used *SVM* classifier (Herbrich et al., 2000) to estimate the model on M. These two methods are similar with most of the previous work for disambiguation, because all of them disambiguate a mention of a name based on the distribution

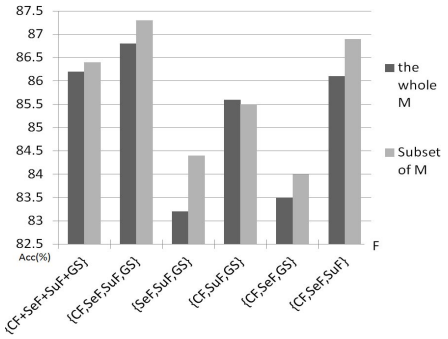


Figure 5: Accuracy on Development Set (As Θ has one parameter - its dimensionality h, the performance here is the ceiling performance obtained on the development set at the best dimensionality in {10, 50, 100,...})

knowledge learned from other labeled names. Row 5 (or 6) A_q+Θ *(or* Θ_j*)* shows the accuracy of our lazy learning model, which generalized the linker on both the distribution knowledge learned from A_q and the predictive structure Θ shared by A_q and M. Row 5 does not use feature split or data set M partitioning for learning Θ, but Row 6 uses them. Comparison of Row 6 and Row 2, 3 indicates our lazy learning model achieves significant improvements of 4.1% and 3.8%, respectively ($\rho < 0.05$, χ^2 statistical significance test). This significant improvement obtained by our approach is from solving the distribution problem (see Section 1) of previous methods.

Besides, Row 4 (M+A_q) used *empirical risk minimization* to estimate u on the data set M and A_q directly. Comparing it with our lazy learning model, the idea to learn the shared predictive information Θ achieves significant ($\rho < 0.05$) gain. This is because, rather than directly using M with a lot of noise, we only incorporate the useful information in M shared with A_q to our learning process.

	ALL	inKB	NIL	PER	ORG	GPE
M(Eq.3)	83.7	81.1	85.9	92.0	82.1	76.9
M(SVM)	84.0	78.5	88.6	92.1	84.0	76.0
M+A_q	84.5	81.4	87.1	92.7	82.7	78.1
A_q+Θ	86.6	84.5	88.3	94.8	85.2	79.7
A_q+Θ_j	**87.8**	85.5	90.0	96.1	86.3	80.9

Table 3: Micro-averaged Accuracy on Test Set

6.5 Comparison with State-of-the-Art Performance

We also compare our approach with the top systems in *TAC-10*. As shown in Figure 6, our lazy learning model achieves a 2% (or 5.9%) improvement over the best (or second best) system in *TAC-10*. The best system "lcc" used a state-of-the-art machine learning algorithm (i.e., logistic classifier) for disambiguation. However, same with other previous work, they only trained their model on data set M without considering the knowledge related to the queried name. Comparing it with our approach, it proves that our lazy learning model has effectively tackled the distribution

problem in the previous work and indeed improved the disambiguation systems. On the *TAC-11* data set, we obtain the similar result. We apply our method in this paper to our system in *TAC-11* (Zhang et al., 2011), which achieves 87.6% with a 1.3% improvement.

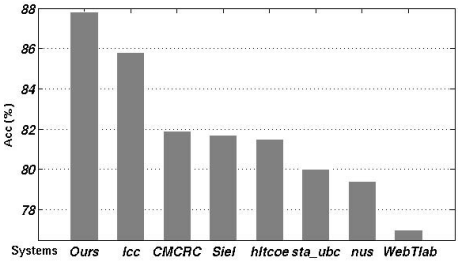


Figure 6: A Comparison with *TAC-10* Systems

6.6 Response Time

Our lazy learning delays the generalization on the labeled data until receiving the query. Hence, comparing with previous work, it increases the response time of the system for each query. However, many of the entity linking applications such as *knowledge base population* do not require real-time user interaction, and therefore they are time-insensitive applications. For those potential time-sensitive applications, we can calculate u_i' (i=1,...,K) in the optimization procedure of Eq. 5 before receiving the query and extract the features from the candidates in parallel. In our experiment, when using 8 *CPUs* (*1.86GHz*) for the multi-thread configuration and using Lemur/Indri[10] to index and search document for generating A_q, our approach can disambiguate the mentions in an article as fast as the baseline method (*M (Eq.3)*) using a single *CPU*.

7 Conclusions and Future Work

With the goal of achieving higher disambiguation performance, our focus was to solve the distribution problem in previous approaches. We have presented a lazy learning model, which can incorporate the distribution knowledge of the queried name to the learning process. To obtain this distribution knowledge, we proposed to automatically label relevant instances A_q for the queried name. Besides, instead of using or combining labeled data set M directly to train the linker, we proposed to use the predictive structure Θ shared by M and A_q. Our experiment showed that the best configuration of Θ was to use feature split over all the feature groups and use data set M partitioning according to NE type. Finally, our experiments also proved that previous approaches for entity linking can be significantly improved.

In the future, to further improve the disambiguation performance, we would like to explore more methods to learn the knowledge from M and A_q.

Acknowledgments

This work is partially supported by Microsoft Research Asia eHealth Theme Program.

[10] http://www.lemurproject.org/indri.php

References

Amigo, E., Artiles, J., Gonzalo, J., Spina, D., Liu, B., and Corujo, A. (2010). Weps3 evaluation campaign: Overview of the on-line reputation management task. In *CLEF (Notebook Papers/LABs/Workshops) 2010*.

Ando, R. K. (2006). Applying alternating structure optimization to word sense disambiguation. In *Conference on Natural Language Learning (CoNLL)*.

Ando, R. K. and Zhang, T. (2005a). A framework for learning predictive structures from multiple tasks and unlabeled data. In *Journal of Machine Learning Research, 6(Nov):1817–1853*.

Ando, R. K. and Zhang, T. (2005b). A high-performance semi-supervised learning method for text chunking. In *Annual Meeting of the Association for Computational Linguistics (ACL)*.

Artiles, J., Gonzalo, J., and Sekine, S. (2007). The semeval-2007 web evaluation: Establishing a benchmark for the web people search task. In *the Fourth International Workshop on Semantic Evaluations (SemEval-2007)*.

Bagga, A. and Baldwin, B. (1998). Entity-based cross-document coreferencing using the vector space model. In *joint conference of the International Committee on Computational Linguistics and the Association for Computational Linguistics (COLING-ACL)*.

Blei, D., Ng, A. Y., and Jordan, M. I. (2003). Latent dirichlet allocation. In *Journal of Machine Learning Research 3:993-1022, 2003*.

Bunescu, R. and Pasca, M. (2006). Using encyclopedic knowledge for named entity disambiguation. In *the 11th Conference of the European Chapter of the Association for Computational Linguistics*.

Chen, Z. and Ji, H. (2011). Collaborative ranking: A case study on entity linking. In *the Conference on Empirical Methods in Natural Language Processing*.

Cucerzan, S. (2007). Large-scale named entity disambiguation based on wikipedia data. In *the Conference on Empirical Methods in Natural Language Processing*.

Dredze, M., McNamee, P., Rao, D., Gerber, A., and Finin, T. (2010). Entity disambiguation for knowledge base population. In *23rd International Conference on Computational Linguistics*.

Finin, T., Syed, Z., Mayfield, J., McNamee, P., and Piatko, C. (2009). Using wikitology for cross-document entity coreference resolution. In *AAAI Conference on Artificial Intelligence*.

Gottipati, S. and Jiang, J. (2011). Linking entities to a knowledge base with query expansion. In *the Conference on Empirical Methods in Natural Language Processing*.

Han, X. and Sun, L. (2011). A generative entity-mention model for linking entities with knowledge base. In *the 49th Annual Meeting of the Association for Computational Linguistics (ACL)*.

Herbrich, R., Graepel, T., and Obermayer, K. (2000). Large margin rank boundaries for ordinal regression. In *Advances in Large Margin Classifiers (pp. 115-132)*.

Ji, H. and Grishman, R. (2011). Knowledge base population: Successful approaches and challenges. In *the 49th Annual Meeting of the Association for Computational Linguistics*.

Ji, H., Grishman, R., and Dang, H. T. (2011). An overview of the tac2011 knowledge base population track. In *Text Analytics Conference*.

Lehmann, J., Monahan, S., Nezda, L., Jung, A., and Shi, Y. (2010). Lcc approaches to knowledge base population at tac 2010. In *Text Analysis Conference 2010 Workshop*.

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

Mihalcea, R. and Csomai, A. (2007). Wikify!: linking documents to encyclopedic knowledge. In *the sixteenth ACM conference on Conference on information and knowledge management*.

Milne, D. and Witten, I. H. (2008). Learning to link with wikipedia. In *the ACM Conference on Information and Knowledge Management*.

Nguyen, H. T. and Cao, T. H. (2008). Named entity disambiguation on an ontology enriched by wikipedia. In *Research, Innovation and Vision for the Future. RIVF*.

Ploch, D. (2011). Exploring entity relations for named entity disambiguation. In *49th Annual Meeting of the Association for Computational Linguistics (ACL)*.

Popov, B., Kiryakov, A., Ognyanoff, D., Manov, D., and Kirilov, A. (2004). Kim - a semantic platform for information extraction and retrieval. In *Journal of Natural Language Engineering*.

Ratinov, L., Roth, D., Downey, D., and Anderson, M. (2011). Local and global algorithms for disambiguation to wikipedia. In *the 49th Annual Meeting of the Association for Computational Linguistics*.

Spina, D., Amigo, E., and Gonzalo, J. (2011). Filter keywords and majority class strategies for company name disambiguation in twitter. In *CLEF*.

Wang, P. and Domeniconi, C. (2008). Building semantic kernels for text classification using wikipedia. In *14th ACM SIGKDD international conference on Knowledge discovery and data mining*.

Zhang, W., Su, J., Chen, B., Wang, W., Toh, Z., Sim, Y., Cao, Y., Lin, C. Y., and Tan, C. L. (2011). I2r-nus-msra at tac 2011: Entity linking. In *Text Analysis Conference*.

Zhang, W., Su, J., Tan, C., and Wang, W. (2010). Entity linking leveraging automatically generated annotation. In *23rd International Conference on Computational Linguistics*.

Zheng, Z., Li, F., Huang, M., and Zhu, X. (2010). Learning to link entities with knowledge base. In *Annual Conference of the North American Chapter of the ACL*.

The Use of Dependency Relation Graph to Enhance the Term Weighting in Question Retrieval

Weinan Zhang[1*] *Zhao-yan Ming*[2†] *Yu Zhang*[1]
Liqiang Nie[2] *Ting Liu*[1] *Tat-Seng Chua*[2]
(1) School of Computer Science and Technology, Harbin Institute of Technology, Harbin, China
(2) School of Computing, National University of Singapore, Singapore
{wnzhang,zhangyu,tliu}@ir.hit.edu.cn
{mingzhaoyan,nieliq,dcscts}@nus.edu.sg

ABSTRACT
With the emergence of community-based question answering (cQA) services, question retrieval has become an integral part of information and knowledge acquisition. Though existing information retrieval (IR) technologies have been found to be successful for document retrieval, they are less effective for question retrieval due to the inherent characteristics of questions, which have shorter texts. One of the major common drawbacks for the term weighting-based question retrieval models is that they overlook the relations between term pairs when computing their weights. To tackle this problem, we propose a novel term weighting scheme by incorporating the dependency relation cues between term pairs. Given a question, we first construct a dependency graph and compute the relation strength between each term pairs. Next, based on the dependency relation scores, we refine the initial term weights estimated by conventional term weighting approaches. We demonstrate that the proposed term weighting scheme can be seamlessly integrated with popular question retrieval models. Comprehensive experiments well validate our proposed scheme and show that it achieves promising performance as compared to the state-of-the-art methods.

TITLE AND ABSTRACT IN ANOTHER LANGUAGE (CHINESE)

利用依存句法关系图改进问句检索中的词项赋权

随着社区型问答服务的出现，问句检索成为了信息及知识获取的重要途径。尽管已有的信息检索模型在文档检索方面取得成功，但是由于问句检索的短文本特性，使得已有检索模型很难适用于问句检索。对于已有的基于词项赋权的问句检索模型而言，一个主要的问题是在计算词项权重时忽略了词项之间的关系。为了解决这个问题，我们提出了一种新的利用词项间的依存句法关系作为线索的词项赋权机制。对于给定问句，我们首先构建依存关系图来计算每个词项对的关联强度，进而我们根据依存关联度来更新常规的词项权重。我们验证了所提出的词项赋权机制能够有效地整合到现有的问句检索模型中，且实验结果相比于当前最优问句检索模型有了较大提升。

KEYWORDS: cQA, Question Retrieval, Dependency Relations, Term Weighting.

KEYWORDS IN CHINESE: 社区型问答，问句检索，依存句法关系，词项赋权.

[*] This work was done when the first author was an intern in National University of Singapore.
[†] Corresponding author

Proceedings of COLING 2012: Technical Papers, pages 3105–3120,
COLING 2012, Mumbai, December 2012.

1 Introduction

With the proliferation and growth of Web 2.0, cQA services, such as Yahoo! Answers[1], Quora[2] and WikiAnswer[3], have emerged as extremely popular alternatives to acquire information online. They permit information seekers to post their specific questions on any topic and obtain answers provided by other participants. Meanwhile, the blooming social networking technologies quickly link the questions to the experts with first hand experiences and propagate well-answered questions among public who also have similar or relevant questions. Over times, a tremendous number of high quality QA pairs devoted by human intelligence has been accumulated as comprehensive knowledge bases, which greatly facilitate general users to seek information by querying in natural languages (Park and Croft, 2010; Ming et al., 2010; Park et al., 2011). As cQA services contain large scale question and answer (Q&A) archives, they offer an invaluable information resource on the Web, to provide answers to new questions posed by the users (Jeon et al., 2005b).

However, question retrieval is not a trivial task (Wang et al., 2009) due to the following problems. First, compared to other indexed documents, the archived questions in current cQA forums are usually very short, which are hard to be matched by lexicon and statistics based approaches such as okapi BM25 (Robertson et al., 1994) model etc. Similar situation happens to twitter search which also deals with short text. It was pointed out in (Teevan et al., 2011; Kwak et al., 2010) that traditional IR technologies can not be directly applied to such applications. Second, the queries are frequently depicted in natural language form that often includes various sophisticated syntactic and semantic features; they can not be easily handled by the simple key word matching models employed by current dominant web search engines.

It is worth mentioning that there already exist several efforts dedicated to research on question match. For example, Xue et al. (Xue et al., 2008) have exploited the translation-based language model (TLM) for question retrieval in large QA database and achieved significant retrieval effectiveness. A syntactic tree kernel approach to tackling the similar question matching problem was proposed by Wang et al. (Wang et al., 2009). Cui et al. (Cui et al., 2005) have tried to measure term dependencies by using different dependency parsing relation paths between the same term pairs. However, they didn't consider how the dependency parsing relation path similarities between term pairs influence the term weights. Despite their success, literature regarding question retrieval is still relatively sparse. Most of the existing work overlook the term relations by assuming that the terms in questions are independent. However, term relations, which reflect the semantic closeness between term pairs, have potentially great impacts on term weighting tasks. Table 1 shows the searching result by TLM which is the state-of-the-art question retrieval model. Though both questions are relevant to the search query, one is ranked at the top, while the other is ranked at 31st. This example demonstrates that the ability to capture term relevance among different dependency parsing relation paths is a key problem in question retrieval.

In this paper, we propose a novel term weighting scheme by exploiting the dependency relations between term pairs, which assumes that strongly dependent terms should be assigned closer weights. Given a question, we first construct a dependency graph and compute the corresponding dependency relevance matrix. Next, based on the dependency relations, a general approach

[1]http://answers.yahoo.com/
[2]http://www.quora.com/
[3]http://wiki.answers.com/

Query	How do you **charge** a farad **capacitor**?	Rank
Correct Position	How do you **charge** a 1 farad **capacitor**?	1
Wrong Position	5 farad **capacitor** for my audio system.. how to **charge** / install?.	31

Table 1: An example of question retrieval result which shows the relevant questions in both correct and wrong ranking position.

is employed to recover the "true weights" from the initial "basic" ones estimated using the traditional methods, such as maximum-likelihood (Xue et al., 2008). Finally, we integrate our term weighting scheme with classic IR model and the state-of-the-art TLM for question retrieval.

The contributions of this work are two-fold:

- First, we propose a novel term weighting scheme that models the closeness in term weights between word pairs in a sentence based on its overall grammatical dependency graph. To the best of our knowledge, this is the first work that tries to enhance term weighting based on dependency relation.
- Second, we seamlessly integrate the novel dependency graph based term weights as an orthogonal factor into the state-of-the-art retrieval models, and produce promising results on real-world data.

The remainder of this paper is organized as follows. Section 2 introduces our term weighting scheme. Sections 3 and 4 present the improved question retrieval model and our experimental results, respectively. Related works are briefly reviewed in Section 5, followed by the conclusions and future work in the last Section.

2 Proposed Term Weighting Scheme

As a key component in question retrieval models, we will first introduce our proposed term weighting scheme based on dependency relation modeling, before we proceed to integrate the model into a unified question retrieval.

2.1 Dependency Relation Detection

As mentioned earlier, dependency relations in the grammatical sense may exist between term pairs and may have certain effects on quantifying term importance. To further study the dependency strength given a question, we first perform dependency parsing utilizing the popular Stanford parser tool (de Marneffe et al., 2006). An illustrating example of parsing result for the question "How do you charge a farad capacitor?" is shown in Figure 1(a). The labels in red font represent dependency relations between term pairs. We note that dependency relations only exist between two terms which are syntactic related. It is also observed that the result of dependency parsing for a sentence is usually represented as a tree. We next remove the pseudo root node from the generated tree and ignore the directions of arcs as well as the labels, we then obtain the undirected dependency graph $G = (V, E)$, for $V = w_1, w_2, ..., w_n$, $E = e_1, e_2, ..., e_m$. where w_i represents the term in query, and e_j represents the undirect relation between terms.

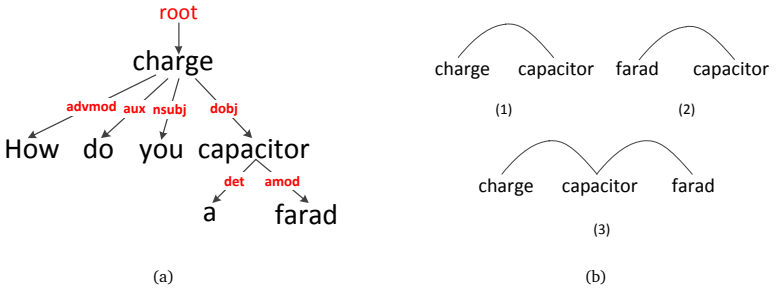


Figure 1: Illustration of the dependency relations for the question "How do you charge a farad capacitor?": (a) dependency parsing tree and (b) dependency relation path.

The undirected graph ensures that every term pair in a given question has a dependency relation path, with shorter paths reflect stronger relations. Figure 1(b) shows the representative length of the dependency relation paths from dependency relation graph. From Figure 1(b), it is obvious that the term "charge" is the direct neighbour of term "capacitor", hence, the dependency relation path length dr_path_len equals to 1 as shown in Figure 1(b) (1). However, "charge" is a bit farther away from the term "farad" as the dr_path_len between term "charge" and "farad" equals to 2 as shown in Figure 1(b) (3). This implies that "charge" should be weighted more closely with "capacitor" than with "farad".

2.2 Dependency based Closeness Estimation for Pairwise Terms

Several existing methods can be employed to compute the closeness between pairwise terms, such as pointwise mutual information (pmi), Chi and mutual information (Gao et al., 2004; Terra and Clarke, 2003). However, few of them take the syntactic dependency into consideration. Instead our approach estimates the dependency relevance of term pairs by linearly integrating multi-faceted cues, i.e., dependency relation path analysis as well as probabilistic analysis.

First, from the perspective of dependency relation path, we denote $dr\_path\_len(t_i, t_j)$ as the length of dependency relation path between term t_i and t_j. The dependency relevance can be defined as:

$$Dep(t_i, t_j) = \frac{1}{b^{dr\_path\_len(t_i,t_j)}} \quad (1)$$

where b is a constant larger than 1, which is selected based on a development set comprising 28 questions, which are randomly sampled from our querying collection. We tune b to the value that optimize the MAP. We name this metric as term dependency metric.

Second, we perform statistical analysis to capture the closeness of term pairs by pmi (Terra and Clarke, 2003) which directly capture the statistical relevance or independence between two terms, share many characteristics as mutual information. It can be formally formulated as:

$$Close_{pmi}(t_i, t_j) = \log \frac{p(t_i, t_j)}{p(t_i)p(t_j)} \quad (2)$$

where $p(t_i, t_j) = \frac{N_d(t_i,t_j)}{N_D}$ represents the probability of co-occurrence between terms t_i and t_j. $p(t_i) = \frac{N_d(t_i)}{N_D}$ and $p(t_j) = \frac{N_d(t_j)}{N_D}$ are respectively the probability of t_i and t_j occur in the whole data collection, where $N_d(t_i, t_j)$ represents the number of documents that contain both t_i and t_j. N_D represents the total number of documents. Meanwhile, $N_d(t_i)$ and $N_d(t_j)$ represent the number of documents that contain terms t_i and t_j respectively. This metric is referred to as term closeness metric.

Finally, we linearly combine the above two metrics by metric combination as:

$$w_{rel(i,j)} = \lambda Dep(t_i, t_j) + (1 - \lambda) Close_{pmi}(t_i, t_j) \tag{3}$$

where λ is a trade-off parameter.

2.3 Reallocation of Relation-based Term Weights

In the above section, we introduce the dependency relevance between two terms. Through capturing the strength of relevance, we expect to optimize the term weights. In this section, we will introduce the method which we use to reallocate term weights using dependency relevance.

First, for a given question q, we compute the dependency relevance among terms in q. If there are n terms in question q, we can construct a $n \times n$ matrix M, which we call the dependency relevance matrix. The element in M, m_{ij}, represents the dependency relevance between terms t_i and t_j computed using Equation 3. Note that as the dependency relevance graph is undirected, M is a symmetric matrix.

Second, we use orthogonal transformation to transform matrix M into a random matrix E so that we can ensure that there must be an analytical solution for the equations of which coefficient matrix equals E, where each elements in E is in range $[0, 1)$, and the sum of elements in the same row equals to 1. Hence, E has an eigenvalue that equals to 1. The solution vector of E exists and the vector with eigenvalue 1 corresponds to the solution vector. In addition, $E = D^{-1}M$, where D^{-1} is the orthogonal matrix which is used to transform the matrix M to E. Moreover, when matrix E and B are written as $E = D^{-\frac{1}{2}}BD^{\frac{1}{2}}$ and $B = D^{-\frac{1}{2}}MD^{-\frac{1}{2}}$, we can see that E is similar in structure with matrix B. Therefore, E and B will have the same eigenvalues. In fact, after solving the eigenvalues of matrix B, we get the eigenvalues of E.

Third, once we transform the dependency relevance matrix into a random matrix, we can obtain the analytical solution as term weights. However, we can also see that the analytical solution is not dependent on the initial term weight vector W_q^0. In our method, the initial term weight vector is estimated using the traditional IR models, such as VSM, BM25 and LM, and the translation-based language model (TLM) (Xue et al., 2008). Although, these models are term independent models, their term weighting schemes can also reflect the relevance between the query and documents. Hence, we linearly combine the initial term weight vector into our term weighting scheme and the analytical solution is the final optimized term weight vector W_q^*, which is derived as shown in Algorithm 1.

Term weight reassignment can be regarded as recovering the "true" weights from the initial one by using dependency relation information. The initial term weights provide a baseline for the "true" weights. Though noisy, they still reflect partial facts of the "true" weights and thus need to be preserved to some extent. Therefore, we introduce the trade-off parameter α. A small α means that the initial term weights play important role. When $\alpha = 0$, the new term weights will be the same as the initial weights.

Algorithm 1: The term weighting reallocation algorithm

Input: W_q^0, M
Output: W_q^*
Compute: $E = D^{-1}M$
Given: α
$W_q = \alpha E W_q + (1-\alpha)W_q^0$
Solution:
$W_q^* = (1-\alpha)(1-\alpha E)^{-1}W_q^0$

3 Unified Question Retrieval Model

To demonstrate that our term weighting scheme can be seamlessly integrated with the current popular question retrieval models without any underlying change, we first introduce the classic IR models and describe ways that our proposed term weighting scheme can be integrated.

3.1 Classic IR Models (VSM, BM25, LM)

The VSM model has been widely used in question retrieval. We consider a popular variation of this model, given query q, the ranking score S_{q,q^c} of the question q^c can be computed as follows:

$$S_{q,q^c} = \frac{\sum_{t \in q \bigcap q^c} w_{t,q} w_{t,q^c}}{\sqrt{\sum_t w_{t,q}^2}\sqrt{\sum_t w_{t,q^c}^2}}, \tag{4}$$
$$\text{where } w_{t,q} = \ln(1 + \frac{N}{f_t}), w_{t,q^c} = 1 + \ln(tf_{t,q^c}).$$

Here, given the query question q, S_{q,q^c} represents the ranking score of candidate question q^c. N is the number of questions in the collection, f_t is the number of questions that contain term t, and tf_{t,q^c} is the frequency of term t in q^c.

While the VSM model favors short questions, the BM25 model takes into account the question length to overcome this problem. Given a query q, the ranking score S_{q,q^c} of the question q^c can be computed as follows:

$$S_{q,q^c} = \sum_{t \in q \bigcap q^c} w_{t,q} w_{t,q^c}, \tag{5}$$
$$\text{where } w_{t,q} = \ln(\frac{N + f_t + 0.5}{f_t + 0.5}),$$
$$w_{t,q^c} = \frac{(k+1)tf_{t,q^c}}{k(1-\mathbb{b}) + \mathbb{b}\frac{W_{q^c}}{W_A} + tf_{t,q^c}}.$$

Here, k and $\mathbb{b}$ are two empirical parameters. W_{q^c} is the question length of q^c and W_A is the average question length in the whole question set.

The LM model is widely used in information retrieval, and also in question retrieval. The basic idea of the LM model is to estimate a language model for each question, and then rank questions by the likelihood of the query according to the estimated model for questions. Here, we use

Dirichlet smoothing for LM model. Given a query q, the ranking score S_{q,q^c} of the question q^c can be computed as follows:

$$\begin{aligned} S_{q,q^c} &= \prod_{t \in q} P(t|q^c) \\ &= \sum_{t \in q} P(t|M_q) \times \log P(t|M_{q^c}) \\ &= \sum_{t \in q \bigcap q^c} w_{t,q} w_{t,q^c}, \\ &\text{where } P(t|M_q) = tf_{t,q^c}, \\ P(t|M_{q^c}) &= \frac{|q^c|}{|q^c|+\delta} \times \frac{tf_{t,q^c}}{|q^c|} + \frac{\delta}{|q^c|+\delta} \times \frac{tf_{t,C}}{|C|} \end{aligned} \tag{6}$$

Here, C is the collection which contains about 20 millions question and answer pairs. $tf_{t,C}$ is the frequency of term t in C and δ is a smoothing parameter. Dirichlet smoothing is used in language model.

Integrating New Term Weights with Classic IR Models

From the aforementioned classic IR models, we find that they can be generalized to the following format:

$$S_{q,q^c} = \sum_{t \in q \bigcap q^c} w^0_{t,q} w_{t,q^c} \tag{7}$$

where t is term in question query q and $w^0_{t,q}$ represents the weight of term t in q. Note that language model can be transformed into the general form by logarithmic transformation. To replace the original term weights into dependency relevance term weight, we derive the updated form of IR models as follows:

$$S_{q,q^c} = \sum_{t \in q \bigcap q^c} w^{dr}_{t,q} w_{t,q^c} \tag{8}$$

where $w^{dr}_{t,q}$ is the updated weights explored by our proposed term weighting scheme.

3.2 Translation-based Language Model (TLM)

The TLM model, which is the state-of-the-art model in question retrieval, can be formally stated as:

$$p(q|q^c) = \prod_{w \in q} p(w|q^c) \tag{9}$$

where $p(w|q^c)$ is written as:

$$p(w|q^c) = \beta p_{ml}(w|q^c) + (1-\beta) \sum_{t \in q^c} p(w|t) p(t|q^c) \tag{10}$$

Here, given query question q, q^c indicates the candidate question for retrieval. $p(w|q^c)$ and $p(w|t)$ denote the language model and translation model, respectively; and β is the parameter to balance the two models.

Integrating New Term Weights with TLM

It is worth emphasizing that $p_{ml}(w|q^c)$ is the term weighting component in TLM. We further accomplish the unified question retrieval by simply replacing it with our term weighting scheme, and restate it as:

$$p(w|q^c) = \gamma p_{dr}(w|q^c) + (1-\gamma)\sum_{t \in q^c} p(w|t)p(t|q^c) \quad (11)$$

where dr indicates the dependency relevance.

4 Experiments

4.1 Experimental Settings

We collected a large real-world data set from Yahoo! Answers, that contains $1,123,034$ questions as our searching corpora, covering a wide range of topics, including health, internet, etc. From this dataset, we randomly selected 140 questions as our searching queries and 28 as the development set to tune all the involved parameters.

To obtain the relevance ground truth of each question query, we pool the top 20 results from various methods, such as vector space model, okapi BM25 model, language model and our proposed methods. We then asked two annotators, who are not involved in the design of the proposed methods, to independently annotate whether the candidate question is relevant (score 1) with the query question or not (score 0). When conflicts occur, a third annotator was involved in making the final decision.

For evaluation, we use precision at position n ($p@n$)($n = 1, 5, 10$), mean average precision (MAP) and mean reciprocal rank (MRR).

4.2 On Performance Comparison

We evaluate the effectiveness of our proposed term weighting scheme, with several question retrieval approaches.

First, we introduce several baselines, which includes three classic IR models, namely the Vector Space Model (VSM), okapi BM25 model (BM25) and Language model (LM).

- **VSM**: The Vector Space Model is used for question retrieval as baseline-1.
- **BM25**: The okapi BM25 model is used for question retrieval as baseline-2.
- **LM**: The language model based IR model is used for question retrieval as baseline-3.

The reason we use the classic IR models as baselines is that the classical IR models are easy to develop and tractable to operate. They capture the evidences from the whole corpus and perform well in tradition IR (Robertson et al., 1994) and cQA question and answer retrieval (Jeon et al., 2005b) tasks. The parameters in the above methods are tuned using development queries. The smoothing parameter δ of language model is set to 600; the k in BM25 model is set to 1.2 and $\mathbb{b}$ is set to 0.75 by following (Robertson et al., 1994).

Correspondingly, we derive three models which are integrated with our dependency relevance term weighting scheme, namely, **drVSM**, **drBM25** and **drLM**, respectively.

	p@1	p@5	p@10	MAP	MRR
VSM	0.1714	0.1691	0.1297	0.1980	0.1598
%chg	+18.4%	+18.2%	+17.5%	+4.5%	+16.9%
drVSM	**0.2029**	**0.1999***	**0.1523***	**0.2069***	**0.1868***
BM25	0.1857	0.1866	0.1418	0.2133	0.1716
%chg	+15.4%	+14.2%	+15.1%	+3.5%	+15.9%
drBM25	**0.2143**	**0.2131***	**0.1632***	**0.2208***	**0.1989***
LM	0.2071	0.2064	0.1603	0.2635	0.1929
%chg	+13.6%	+13.1%	+13.4%	+3.0%	+10.1%
drLM	**0.2353**	**0.2334***	**0.1818***	**0.2714***	**0.2124***

Table 2: Experiment results of classic IR models and the corresponding enhanced models that are integrated with the proposed dependency relevance term weights. $*$ indicates statistical significance over the respective baselines at 0.95 confidence interval using the t-test. %chg denotes the performance improvement in percent of dependency relevance based term weighting scheme enhanced model over the corresponding baseline.

Table 2 summarizes the experimental results in the five evaluation metrics using the three retrieval models and the three baselines. We can observe that the performances of all the three retrieval models are enhanced by dependency relevance-based term weighting scheme. Meanwhile, they obtain significant improvements over their baselines respectively. It indicates that, on the one hand, the proposed term weighting scheme is effective in question retrieval task. On the other hand, the proposed term weighting scheme provides orthogonal information about term weights when combined with the three classic IR models.

Through the experimental results, we can also see that the three classic IR models benefit differently from the dependency graph-based term weights. We conjecture the reason may be that, first, the LM has the collection smoothing scheme, so the original term weighting scheme is more rational than VSM and BM25. Second, the BM25 term weighting scheme considers question length feature so that it is unbiased in questions with different length, while the VSM favors short questions. Therefore, VSM benefits most from the term weighting scheme, followed by BM25 and LM.

Next, we introduce the state-of-the-art dependency relation-based question retrieval models as follow.

- **PRM**: Dependency relation-based passage retrieval model (PRM), which is proposed by Cui et al. (Cui et al., 2005) for question retrieval as baseline-4.

We use PRM[4] as baseline to check that whether our method is more effective than the previous dependency based IR model in question retrieval.

Next, we introduce the state-of-the-art question retrieval model TLM as another baseline, as we use two metrics to capture the term relevances, we introduce tcmTLM and tdmTLM to check the performances of each metric in question retrieval. Finally, we combine the above two metrics in

[4]We ran PRM under the setting of baseline-5 as described in (Cui et al., 2005)

form of Equation 3 and get the dependency relevance-based question retrieval model (**drTLM**). We describe the above four models as follows.

- **TLM**: Translation-based language model (TLM) which is proposed by Xue et al. (Xue et al., 2008), we implement it as baseline-5.
- **tcmTLM**: TLM integrated with our term weighting scheme where only term closeness metric is utilized to estimate the term relations.
- **tdmTLM**: TLM integrated with our term weighting scheme where only term dependency metric is utilized to estimate the term relations.
- **drTLM**: TLM integrated with our term weighting scheme where both term closeness metric and term dependency metric are combined to estimate the term relations.

For each method mentioned above, the involved parameters are carefully tuned, and the parameters with the best performances are used to report the final comparison results.

	p@1	p@5	p@10	MAP	MRR
PRM	0.2429	0.2397	0.1974	0.3595	0.2174
%chg	+14.1%	+10.6%	+7.4%	+16.0%	+18.8%
TLM	0.1928	0.1976	0.1759	0.2889	0.1889
%chg	+37.5%	+34.2%	+20.5%	+44.3%	+36.7%
tcmTLM	0.2084	0.2036	0.1903	0.3123	0.2145
%chg	+33.0%	+30.2%	+11.4%	+33.5%	+20.4%
tdmTLM	0.2675	0.2590	0.2086	0.4014	0.2495
%chg	+3.6%	+2.4%	+1.6%	+3.9%	+3.5%
drTLM	$\mathbf{0.2771}^{*}_{\dagger}$	$\mathbf{0.2651}^{*}_{\dagger}$	$\mathbf{0.2120}^{*}_{\dagger}$	$\mathbf{0.4170}^{*}_{\dagger}$	$\mathbf{0.2583}^{*}_{\dagger}$

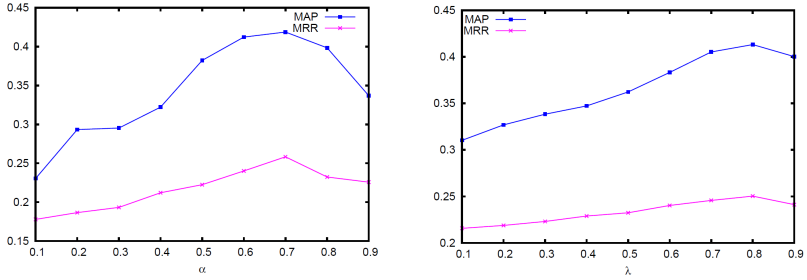
Table 3: Performance comparison among different question retrieval methods. $*$ and † indicate that the statistical significance over baseline and tcmTLM respectively is distributed within 0.95 confidence interval using the t-test. %chg denotes the boosted performance by **drTLM** in percentage. The results of our method are in bold font.

It can be observed that the performance of TLM can be enhanced by our term weighting scheme. This is due to the fact that our term weighting scheme captures the relations between term pairs and get better term weighting allocation. Compared with the existing dependency relation-based question answering passage retrieval model PRM, our method outperforms PRM in the above five evaluation methods. In particular, we only use dependency path length as a bridge to capture the relevance between two terms, which is a simple, stable and efficient way to use deep parsing, and get better performance. From this table, we can also observe that TLM integrated with tdm outperforms itself integrated with tcm. This is because tdm characterize more intrinsic dependency relations rather than the simple co-occurrences captured by tcm. Furthermore, the TLM incorporated with both tcm and tdm achieves the best performance. It worth noting that the performance of TLM is lower than that in the original paper, it is because that we use different data set and the answer evidence is not considered here.

4.3 On Parameter Sensitivity

In this section, we present how the parameters influence on question retrieval performance. In our experiments, grid search is performed to obtain the optimal values for parameters on the development set under the results of **drTLM** model.

As discussed before, smaller α in Algorithm 1 means initial term weighting scores dominate the term reassignment task, and ignore the dependency relations at all when α trends to zero. While a larger α means that our term weighting scheme will play a major role. The curve of MAP and MRR with different α value is presented in Figure 2(a) with other parameters fixed. We can see that the MAP and MRR increase with α growing and arrive at the peak when $\alpha = 0.7$ on our real dataset (development set); the performance then decrease sharply after that.



(a) (b)

Figure 2: The performance of **drTLM** with different α (a) and λ (b), when other parameters are fixed.

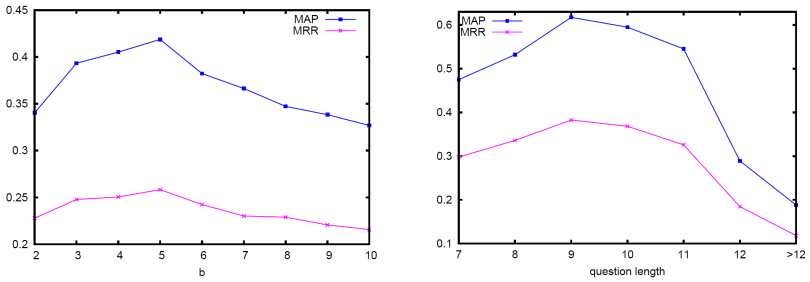
From Figure 2(a), we can infer that in our term weighting scheme, the improvement is mainly depended on term dependency relevance weighting scheme, meanwhile, the original term weights can somewhat influence the performance.

In our term weighting scheme, we introduce two relevance metrics between term pairs for term weighting reallocation as represents in Equation 3, the parameter λ is used to balance the two relevance metrics. Figure 2(b) shows the variation of MAP and MRR when λ is changing from 0.1 to 0.9.

From Figure 2(b), we can see that the MAP and MRR increase with λ growing and arrive at peak when $\lambda = 0.8$. It indicates that comparing with term closeness metric, the term dependency metric play the dominant role in the proposed term weighting scheme. Furthermore, it also illustrates that the term dependency relevance metric can effectively capture the strength of relations between two terms and influence the reallocation of term weights.

To further check the influence of parameter b in Equation 1 on the performance of question retrieval, Figure 3(a) presents the curve of MAP and MRR with different b value. From Figure 3(a), we can see that the performance arrive at the peak when $b = 5$.

In addition, we also consider the influence of question length, which indicates the number of words in one question, on the performance of question retrieval. Figure 3(b) shows the curve



(a) (b)

Figure 3: The performance of **drTLM** with different b (a) and question length (b), when other parameters are fixed.

of MAP and MRR with different question length. Through Figure 3(b), we actually check the ability of our proposed method on handling queries in different length as well as in different complexities. From Figure 3(b), we can see that the proposed method can adapt queries in wide range of length, which is from 7 to 11, and get well performance on question retrieval. Meanwhile, we also see that in question retrieval, neither shorter nor longer queries, can get better performance. It also reveals that natural language question queries can better represent users' searching intent than key words queries as they contain plentiful lexical information, as well they may introduce more noise. The parameter γ in Equation 11 equals 0.8, which also illustrates that in the dependency relevance-based TLM (**drTLM**) model, the proposed term weighting scheme contributes more in question retrieval. The improvement of searching results mainly depend on the reallocated term weights.

4.4 On Efficiency Analysis

For our proposed approach, we can see that the computational cost mainly comes from two parts: (1) question dependency parsing; (2) graph-based term weighting. Assume the question length is n, it can be analyzed that the computational cost scales as $O(n^3)$. In our data collection, n is averaged as 10.8, which leads to very low computational cost. In our experiments, we compare the time of process for each search round between the proposed **drTLM** model and PRM model which is also a dependency relation-based model that we use as baseline-5. The average search rounds that **drTLM** complete at one second is 14, while PRM is 17 (with a pc of 72G memory and Intel(R) Xeon(R) CPU E5620@2.40GHz). It means that the above two methods are comparable in efficiency. However, our proposed method doesn't need to training models, which leads to more efficient. Meanwhile, for further efficiency, we can also use iterative methods to get the numerical solution instead of analytical solution in graph-based term weighting.

4.5 Case Study

Table 4 representatively illustrates the top 5 search results for the query "How do you charge a farad capacitor?" by TLM and **drTLM** which is our dependency graph enhanced TLM. Clearly, our proposed model returns more relevant questions at top positions, mainly due to the adjusted weights for term "charge", "farad", and "capacitor".

Rank No.	TLM	**drTLM**
1	**How do you charge a 1 farad capacitor**?	**How to charge a farad capacitor**?
2	**How do you charge a 5 farad capacitor**?	**How do you charge a 1 farad capacitor**?
3	What resistor do you use to charge a 1 farad capacitor?	**How do you charge a 5 farad capacitor**?
4	How do you install a farad amp capacitor?	How do you install a 1 farad capacitor?
5	How do you hook up a 3 farad capacitor to two amps?	**5 farad capacitor for my audio system.. how to charge / install**?

Table 4: Search results comparison between TLM and **drTLM** for query "How do you charge a farad capacitor?".Questions in bold font are relevant ones.

5 Related Work

The existing IR technologies are frequently based on Bag-of-Words models and regard both the query and documents in collections as composition of individual and independent words. For example, Ponte et al. (Ponte and Croft, 1998) utilized unigram language model for information retrieval. Jones et al. (Jones et al., 2000) proposed the binary independent retrieval (BIR) model to capture the relevance between queries and documents. Duan et al. (Duan et al., 2008) proposed a new language model to capture the relation between question topic and focus. They may not be directly applicable in the question retrieval domain due to at least two reasons. First, compared to the simple keywords based search, the querying questions are usually represented in natural language and depict some concepts linked by intrinsic semantic relationships. Second, the to be searched documents are also questions, which are far shorter than the verbose documents in traditional search approaches.

Jeon et al. (Jeon et al., 2005a,b), moving forward one step, provided comparison of four different retrieval models, i.e., vector space model, okapi, language model and translation model for question retrieval in archived cQA data, experimental results revealed that the translation model outperforms the other models. Later, Xue et al. (Xue et al., 2008) combined the language model and translation model to a translation-based language model and observed better performance in question retrieval. Following that, Ming et al. (Ming et al., 2010) utilized three domain specific metrics to explore term weights and integrated them into existing IR models. However, most of these term weighting based retrieval models ignore the dependency relations between term pairs.

Researchers never stop to capture the term dependencies for IR models. For instance, (Song and Croft, 1999; Srikanth and Srihari, 2002) replaced the unigram to bigram and bi-term in language model. Gao et al. (Gao et al., 2004) proposed a dependency language model

to capture term dependencies through dependency parsing relations. Park et al. (Park and Croft, 2010) explore dependency features for term ranking in verbose query. Moreover, they proposed a quasi-synchronous IR model (Park et al., 2011) to integrate dependency information. Cui et al. (Cui et al., 2005) have tried to measure the terms dependencies by using different dependency parsing relation paths between same term pairs. Sun et al. (Sun et al., 2006, 2005) explored dependency relations for query expansion and answer extraction in question passage retrieval and answering retrieval. However, they only estimated term dependencies or syntactics between adjacent term and overlooked the nonadjacent cases. To tackle this issue, in this paper, we proposed a term weighting approach by incorporating global dependency relevance.

Conclusions

In this paper, we explored the dependency relations between question terms to enhance the question retrieval in cQA. Given a question, we first automatically constructed a dependency graph, and then estimated the relation strength between vertex pairs. Based on the quantified dependency relations, we proposed a novel term weighting scheme to refine the initial term weights estimated by traditional technologies. Further, we demonstrated that our term weighting approach can be unified with the state-of-the-art question retrieval models. By conducting experiments on real-world data, we demonstrated that our proposed scheme yields significant gains in retrieval effectiveness.

This work begins a new research direction for weighting question terms by incorporating dependency relation cues. In future work, we will further study the dependency relation based term weights by differentiating the importance of relation types and assigning relation-aware weights.

Acknowledgments

This work was supported by the Natural Science Foundation of China (Grant No. 61073129, 61073126), 863 State Key Project (Grant No. 2011AA01A207) and partially supported by NExT Search Centre, which is supported by the Singapore National Research Foundation & Interactive Digital Media R&D Program Office, MDA under research grant (WBS:R-252-300-001-490).

References

Cui, H., Sun, R., Li, K., Kan, M.-Y., and Chua, T.-S. (2005). Question answering passage retrieval using dependency relations. In *Proceedings 28th annual international ACM SIGIR conference*, pages 400–407.

de Marneffe, M.-C., MacCartney, B., and Manning, C. D. (2006). Generating typed dependency parses from phrase structure trees. In *LREC*.

Duan, H., Cao, Y., Lin, C.-Y., and Yu, Y. (2008). Searching questions by identifying question topic and question focus. In *ACL*, pages 156–164.

Gao, J., Nie, J.-Y., Wu, G., and Cao, G. (2004). Dependence language model for information retrieval. In *SIGIR*, pages 170–177.

Jeon, J., Croft, W. B., and Lee, J. H. (2005a). Finding semantically similar questions based on their answers. In *SIGIR*, pages 617–618.

Jeon, J., Croft, W. B., and Lee, J. H. (2005b). Finding similar questions in large question and answer archives. In *CIKM*, pages 84–90.

Jones, K. S., Walker, S., and Robertson, S. E. (2000). A probabilistic model of information retrieval: development and comparative experiments. In *Information Processing and Management*, pages 779–840.

Kwak, H., Lee, C., Park, H., and Moon, S. (2010). What is twitter, a social network or a news media? In *WWW'10: Proceedings of the 19th International World Wide Web Conference*.

Ming, Z., Chua, T.-S., and Cong, G. (2010). Exploring domain-specific term weight in archived question search. In *CIKM*, pages 1605–1608.

Park, J. H. and Croft, W. B. (2010). Query term ranking based on dependency parsing of verbose queries. In *SIGIR*, pages 829–830.

Park, J. H., Croft, W. B., and Smith, D. A. (2011). A quasi-synchronous dependence model for information retrieval. In *Proceedings of the 20th ACM international conference on Information and knowledge management*, CIKM '11, pages 17–26.

Ponte, J. M. and Croft, W. B. (1998). A language modeling approach to information retrieval. In *SIGIR '98: Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 275–281.

Robertson, S. E., Walker, S., Jones, S., Hancock-Beaulieu, M., and Gatford, M. (1994). Okapi at trec-3. In *TREC*.

Song, F. and Croft, W. B. (1999). A general language model for information retrieval. In *CIKM '99: Proceedings of the eighth international conference on Information and knowledge management*, pages 316–321.

Srikanth, M. and Srihari, R. K. (2002). Biterm language models for document retrieval. In *SIGIR*, pages 425–426.

Sun, R., Cui, H., Li, K., Kan, M.-Y., and Chua, T.-S. (2005). Dependency relation matching for answer selection. In *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '05, pages 651–652.

Sun, R., Ong, C.-H., and Chua, T.-S. (2006). Mining dependency relations for query expansion in passage retrieval. In *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '06, pages 382–389.

Teevan, J., Ramage, D., and Morris, M. R. (2011). # twittersearch : a comparison of microblog search and web search. In *WSDM*, pages 35–44.

Terra, E. and Clarke, C. L. A. (2003). Frequency estimates for statistical word similarity measures. In *NAACL '03: Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology*, pages 165–172.

Wang, K., Ming, Z., and Chua, T.-S. (2009). A syntactic tree matching approach to finding similar questions in community-based qa services. In *SIGIR*, pages 187–194.

Xue, X., Jeon, J., and Croft, W. B. (2008). Retrieval models for question and answer archives. In *SIGIR*, pages 475–482.

Long-tail Distributions and Unsupervised Learning of Morphology

Qiuye Zhao[1] *Mitch Marcus*[1]
(1) University Of Pennsylvania
qiuye@cis.upenn.edu, mitch@cis.upenn.edu

Abstract

In previous work on unsupervised learning of morphology, the long-tail pattern in the rank-frequency distribution of words, as well as of morphological units, is usually considered as following Zipf's law (power-law). We argue that these long-tail distributions can also be considered as lognormal. Since we know the conjugate prior distribution for a lognormal likelihood, we propose to generate morphology data from lognormal distributions. When the performance is evaluated by a token-based criterion, giving more weights to the results of frequent words, the proposed model preforms significantly better than other models in discussion. Moreover, we capture the statistical properties of morphological units with a Bayesian approach, other than a rule-based approach as studied in (Chan, 2008) and (Zhao and Marcus, 2011). Given the multiplicative property of lognormal distributions, we can directly capture the long-tail distribution of word frequency, without the need of an additional generative process as studied in (Goldwater et al., 2006).

Keywords: Morphological Learning, Zipf's law, Lognormal distribution, Long tail distribution, Gibbs Sampling, Bayesian approach.

Proceedings of COLING 2012: Technical Papers, pages 3121–3136,
COLING 2012, Mumbai, December 2012.

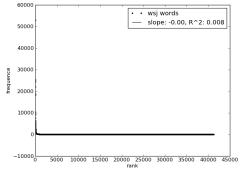
1 Introduction

Unsupervised learning of morphology is an active research area. In this work, we will focus on learning segmentations of words[1]. A segmentation of word w can be denoted as $w = t.f$, which means that concatenating stem t and suffix f gives word w.

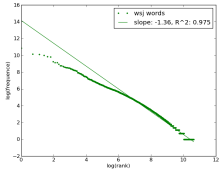
Assuming that stems and suffixes are independently distributed, a baseline generative morphology model is like this,

$$
\begin{aligned}
k^T, k^F &= \text{number of distinct stem types, number of distinct suffix types} \\
\alpha^T, \alpha^F &= \text{constant hyper-parameters} \\
\theta^T, \theta^F &\sim \text{Symmetric-Dirichlet}_{k^T}(\alpha^T), \text{Symmetric-Dirichlet}_{k^F}(\alpha^F) \\
t_{i=1...N}, f_{i=1...N} &\sim \text{Multinomial}(\theta^T), \text{Multinomial}(\theta^F) \\
w_{i=1...N} &\sim I(w = t.f)P(t|\theta^T)P(f|\theta^F)
\end{aligned}
$$

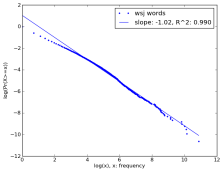
where N is the number of words and $I(w = t.f)$ is the indicator function taking on value 1 when concatenating stem t and suffix f gives word w and 0 otherwise.



(a) The rank-frequency plot of words.



(b) The log-log rank-frequency plot.



(c) The CCDF plot of words.

Figure 1: Observing long-tail distributions of word frequency in WSJ Penn Treebank.

In the above baseline model, there no special statistical property of word frequency is captured, nor of morphological units (e.g. stems or suffixes). As depicted in Figure 1-a, it is well-known that given a large corpus, a rank-frequency plot of words generally exhibits a long tail that is distinctively heavier than in normal (Gaussian) distributions. An alternative way to show this long-tail pattern is by plotting the corresponding Complementary Cumulative Distribution Function (CCDF) on logistic scales. As depicted in Figure 1-c, the CCDF plot behaves like a straight line and is smoother than the corresponding log-log rank-frequency plot in Figure 1-b, which also approaches a straight line for long-tail distributions. When words are segmented into stems and suffixes, distributions of these units can also be examined.

As we are going to show in Section 2.3, the rank-frequency distributions of morphological units also show up as straight lines in log-log rank-frequency plots and CCDF plots. In previous work on morphological learning, such as (Chan, 2008), (Zhao and Marcus, 2011) and (Goldwater et al., 2006, 2011), the straight lines on logistic scales are usually interpreted as following Zipf's law (Zipf, 1949). On the other hand, lognormal distributions with large variance also yield straight lines on both the log-log rank-frequency plot and the CCDF plot. In Section 2, we are going to discuss

[1] A segmentation model may not be well-defined for morphology, especially when morph-rich languages are considered; however, different forms of morphological analyses may be compared as simple segmentation analyses.

more about the long-tail distributions and argue that the signature straight lines may suggest either power-law or lognormal.

So as to take advantage of the special statistical property of morphological units, which is considered as following the Zipf's law in (Chan, 2008), Chan proposes a rule-based bootstrapping algorithm for morphology learning, which is revised in (Zhao and Marcus, 2011) for acquiring functional elements. Even though Zipf's law is discussed in both works, its specific definition does not really matter in the design of the algorithms. For the sake of comparison with other models, we implement a reduced version of the bootstrapping algorithm as described in (Zhao and Marcus, 2011), eliminating all ad-hoc linguistic assumptions encoded in the original algorithm. With this rule-based algorithm, the acquired segmentation model performs rather well when evaluated with the type-based criterion, but notably bad with the token-based evaluation, which gives more weights to the results of frequent words than the type-based evaluation. We will describe this algorithm in Section 3.1 with more details.

The rule-based bootstrapping algorithm utilizes type frequencies only, no matter what form of input is given. On the other hand, as shown in (Goldwater et al., 2006), when Dirichlet-multinomial model is assumed, the option of utilizing token frequency in generative model doesn't help, and in the contrast it hurts the inference of the generative model. Goldwater et al. (2006) argued that this morphology model doesn't capture the special statistical property of word frequency, therefore, an addition generative process is introduced to transform the word frequencies to exhibit the desired distribution. Since the long-tail pattern in word frequency is considered as following Zipf's law, a generalized Chinese restaurant process, Pitman-Yor process (Pitman and Yor, 1997), is exploited for producing power-law distributions. For the sake of comparison, we re-implement the morphology model in (Goldwater et al., 2006), and conduct more experiments with different configurations.

We propose to compute lognormal likelihood, instead of multinomial likelihood, for both stems and suffixes, in the generative morphology model. With this lognormal model, the option of utilizing token frequency, i.e. taking input of the unprocessed text data, dose help the inference of the generative model, especially when the token-based evaluation is preformed. When evaluated by the token-based criterion, which gives more weights to the results of frequent words, the proposed model performs significantly better than other models in discussion, no matter what form of input is fed to the multinomial model or the rule-based algorithm.

With the proposed model, the particular statistical properties of stems and suffixes are utilized in a Bayesian model instead of a rule-based model. Furthermore, as we will discuss in Section 2.3, given the multiplicative property of lognormal distributions, the word frequency distribution can also be predicted as lognormal. Therefore, we can directly capture the statistical property of word frequencies without the need of an additional generative process. Especially, the proposed generative model is more accurate with the token-based evaluation when utilizing token frequency, and more accurate with the type-based evaluation when utilizing type-frequency. This result pattern suggests that the proposed model is able to adapt to real data distribution by itself, therefore, we do not need to concern with justifying the appearance of type frequencies in morphology learning, as pursued in (Goldwater et al., 2006).

We are going to use Gibbs sampling, a standard Markov Chain Monte Carlo (MCMC) method for the inference of generative models. In each iteration, the morphological analysis of each word is sequentially re-sampled from its conditional posterior given morphological analyses of all other words. Since the sampling process is much more complex and time-consuming for the lognormal model than the multinomial model, we propose to constrain the learning of generative models with

the acquisition outputs of the rule-based model. Since the rule-based bootstrapping algorithm takes input of raw corpora only and so the generative models, the combination of these two processes results in a totally unsupervised learning process as well. Even though this method is motivated by the concern of training efficiency, the proposed use of acquisition outputs from a rule-based model also significantly improves the performance of generative models, consistently for both the lognormal and the multinomial model.

2 Long-tail distributions

2.1 The long-tail pattern

Given a large corpus, we can compute the word frequency of each word type by counting its occurrences in the corpus. When we plot word frequency against its rank, such as in Figure 1-a, there is a long tail of the curve composed of the large number of words that occur in low frequency. When plotted on a logistic scale, as in Figure 1-b, the rank-frequency plot behaves like a straight line. An equivalent form of the rank-frequency approach is to plot the corresponding Complementary Cumulative Distribution Function (CCDF). Instead of plotting a function of rank, we can also plot $P(F >= f)$ as a function of frequency f. As shown in Figure 1-c, the CCDF plot also behaves like a straight line on the logistic scale, which is smoother than the log-log rank-frequency plot.

We generally refer this kind of distributions as long-tail distributions, observing that a large portion of its population are composed of low-frequency events, which form a longer tail than normal (Gaussian) distributions. Long-tail patterns have been widely observed in various fields, but they may be studied as different distributions. For example, economists may be familiar with this pattern as Pareto distribution, which is also known as '80-20' rule. In Pareto's original study (Pareto, 1896), the long-tail pattern is shown on CCDF plots, so it took a while for people to understand that it is a power-law distribution and is synonymous with 'Zipf's law' (Newman, 2005). Zipf's law is proposed by linguist George Kingsley Zipf in his study of vocabulary distribution, and is widely used to interpret the straight lines on logistic scales in the study of language. In more recent works, the convention of treating long-tail patterns as power-law distributions has been challenged. For example, Downey (Downey, 2001) argues that many networks metrics, such as file sizes and transfer times, should be modeled as lognormal distributions. Lognormal distributions with large variance also yield straight lines on the log-log rank-frequency plot and the CCDF plot.

2.2 Generating Zipf's law and lognormal distributions

Based on the idea of preferential attachment, i.e. a 'richer-get-richer' process, if we generate new word occurrences more likely of popular word types than of rarely seen word types in previous process, then word frequency of the generated corpus may follow Zipf's law or be lognormal, depending on subtle differences in the generative processes.

More specifically, suppose that we are given i words for a start, $i \geq 1$. Let n_k^i denote the number of occurrences of all the words that occur exactly k times in the previous i words. Let $P(w_{i+1} = k)$ denote the probability that the $i + 1_{\text{th}}$ occurrence is a word that has already appeared k times in the previous i words. Consider the following process as described in (Simon, 1955),

$$P(w_{i+1} = k) = \alpha n_0 + F_i n_k^i,$$

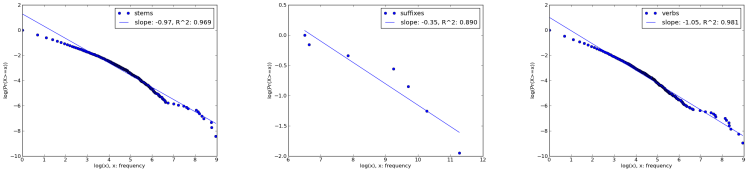
where n_0 and α are constants. If $F_i = \frac{(1-\alpha)}{i}$, then asymptotically, $P(w_i = k)$ will approach a power-law distribution. On the other hand, if the constant item is removed from the above process,

$$P(w_{i+1} = k) = F_i n_k^i,$$

and F_i are independent and identically distributed variables with finite mean and variance, then asymptotically, $P(w_i = k)$ will approach a lognormal distribution.

A even more naive generative model for Zipf's law is Miller's monkey (Miller, 1957), who can not only type with a keyboard, but also distinguish space bar from other keys. If Miller's monkey manages to hit the space bar with a constant probability and never hits the space bar twice subsequently, then the word frequency in the monkey's output follows a power law. One crucial assumption in Miller's demonstration is that all non-space letters are hit with equal probabilities. However, for the case that any two letters are hit with different probabilities, Perline (1996) argues that for all words of length up to a constant, their rank-frequency distribution converges to a lognormal distribution.

After reviewing a brief history of generative models for power-law and lognormal distributions, Mitzenmacher (2004) suggests that *"It might be reasonable to use which ever distribution makes it easier to obtain results."* [2]



(a) The CCDF plot of stems. (b) The CCDF plot of suffixes. (c) The CCDF plot of verbs.

Figure 2: Observing long-tail distributions for morphological units.

2.3 Lognormal distributions

One advantage of modeling long-tail patterns as lognormal is so that the multiplicative property holds, i.e. the product of independent lognormally-distributed random variables is itself lognormally distributed. For example, consider the baseline generative morphology model as described in the introduction, $P(w) = P(t)P(f)$. If both stems and suffixes are lognormally distributed, then word frequency is also lognormally distributed.

So as to examine the distributions of morphological units, with the help of the gold part-of-speech annotations, we segment all verbs into stems and suffixes in WSJ Penn Treebank (Marcus et al., 1993). As shown in Figure 2-a and b, the signature straight lines on CCDF plots suggest that both stems and suffixes of verbs can be modeled as lognormal. Then, given the multiplicative property of lognormal distributions, verbs should also be lognormally distributed, which is confirmed by its CCDF plot in Figure 2-c.

Another reason for one to consider modeling long-tail patterns as lognormal distributions is that we know the conjugate prior distribution for a lognormal likelihood, but not for power-law likelihood. Considering the generative model $P(w) = P(t)P(f)$, if we assume that t and f are lognormal, their mean μ and variance σ^2 can be drawn from Normal priors and Inverse-Gamma priors respectively.

[2] As also pointed out in (Mitzenmacher, 2004), if a power law distribution can have infinite mean and variance, then it is inaccurate to analyze it as lognormal. In present examples, we assume the exponent of power-law distributions is greater than 0, thus it is safe for us to experiment with either distribution.

3 A rule-based model

Given a set of stems $\mathbb{T}$ and a set of suffixes $\mathbb{F}$, we can divide a word w into stem t and suffix f, if $t \in \mathbb{T}$, $f \in \mathbb{F}$ and $w = t.f$. For example, if $\mathbb{T} = \{$*'laugh-'*, *'analyz-'*$\}$, and $\mathbb{F} = \{$*'-ed'*, *'-s'*$\}$, then *'analyzed'* can be segmented into *'analyz-'* and *'-ed'*, but *'red'* won't be segmented. So as to learn such a rule-based morphology model, we need to acquire a set of stems and a set of suffixes.

3.1 A bootstrapping algorithm for acquiring morph units

In this section, we are going to describe a bootstrapping algorithm, adapted from the algorithm for acquiring functional elements in (Zhao and Marcus, 2011). This line of algorithms is especially designed to account for the long-tail pattern observed for stems and suffixes, but not specific to either Zipf's law or lognormal distributions. As stated in (Chan, 2008), in which a bootstrapping algorithm is originally proposed for acquiring transformation-based morphological rules, *"what matters, though, is that the number of types per inflection decreases rapidly, ..., there are few highly frequent inflections, and many more infrequent ones."*

Following (Zhao and Marcus, 2011), our algorithm is built upon a distributional property of 'functional elements': they occur in diverse contexts. In the case of learning morphological segmentations, 'functional elements' are suffixes in context of stems. For example, an inflectional ending *'-ed'* can be concatenated to most verb stems to derive past tense forms, in contrast to which a non-sense suffix *'-roached'* can only be seen in few particular word types.

If all prefixes in all possible divisions of a word count as stems, then as computed from the WSJ corpus, the top three suffixes with the highest contextual diversity will be *'-s'*, *'-d'* and *'-e'*, two of which do not comply with common sense of morphological suffixes in English. In other words, for acquiring suffixes, we want to compute their contextual diversity according to properly justified stems only. The most simple way of justifying stems as proper contexts for suffixes is to check whether it serve as context of more than one type of suffixes. For example, stem *'laughin-'* should not be justified, because except for a particular suffix *'-g'*, it cannot be concatenated with other suffixes to form legal words.

Given a set of properly justified stems T, we measure the contextual diversity of a suffix f as

$$div(f, \mathbb{T}) = \sum_{t \in \mathbb{T}} \delta(t.f),$$

where $\delta(t.f)$ is set to 1 if $t.f$ forms any word, otherwise 0. For example, if we are given a set of properly justified stems, including *'laugh-'* but not *'b-'*, the diversity measurement of *'-ing'* will increase by one given the existence of word *'laughing'* but not word *'bing'*.

Algorithm 1 The bootstrapping algorithm for acquiring stems and suffixes

Require: A corpus $\mathbb{C}$ containing raw text only.
  Initialize set $\mathbb{F}_0$ to be empty and set $\mathbb{T}_0$ to contain all possible prefixes
  for $k = 1...K$ iterations **do**
    Let $\mathbb{F}_k$ contain the top k suffixes with the highest diversities measured by $div(f, \mathbb{T}_{k-1})$.
    Let $\mathbb{T}_k$ contain stems that form legal words with suffixes in $\mathbb{F}_k$
  end for
  return $\mathbb{F}_K$ and $\mathbb{T}_K$

We implement a reduced version of the bootstrapping algorithm in (Zhao and Marcus, 2011),

eliminating all ad-hoc linguistic assumptions encoded in the original algorithm. As depicted in Algorithm 1, the algorithm generates two sets of acquisition outputs during the bootstrapping process, both of which justify the proper set for measuring contextual diversity for each other. As the two sets alternately update during the bootstrapping process, the diversity measurement of either set is expected to be more and more accurate.

The only required input to this algorithm is a corpus $\mathbb{C}$ of raw text without any form of annotation. Set $\mathbb{F}_0$ is initialized to be empty and set $\mathbb{T}_0$ is initialized to contain all prefixes in all possible divisions of all words in corpus $\mathbb{C}$. At the k_{th} bootstrapping iteration, $k > 0$, we compute set $\mathbb{F}_k$ as the top k suffixes of the highest contextual diversity according to set $\mathbb{T}_{k-1}$. And set $\mathbb{T}_k$ contain stems that can form legal words with suffixes in $\mathbb{F}_k$. Since the diversity measurement of suffixes varies over iterations with respect to updated set of stems, a suffix that is selected to output at some iteration, is not guaranteed to be selected in the following iterations.

kth iter.	set $\mathbb{F}$	size of set $\mathbb{T}$
1th	*-s*	668
5th	*-ed, -ing, -s,* ***-e****, -es*	2274
10th	*-ed, -ing,* ***-e****, -s, -es, -er,* ***-ers****, -ion,* ***-ions****, -ly*	2776
20th	*..., -ion,* ***-ers****, -y,* ***-ions****, -al, -or,* ***-ors****,* ***-ings****, -able, -ive, -ly, -aly, -ies*	2993
Fix set $\mathbb{T}$ with all prefixes in all possible divistions of words.		
k=20	*-s,* ***-d****,* ***-e****, -ed,* ***-g****, -n,* ***-ng****, -y, -ing,* ***-t****,* ***-r****, -es, -er, -on,* ***-l****,* ***-rs****,* ***-a****, -ly, -ion,* ***-0***	7091

Table 1: The acquisition outputs by Algorithm 1 over WSJ Penn Treebank.

We run this bootstrapping algorithm for acquiring stems and suffixes from the WSJ corpus. For the sake of comparison, we also experiment without updating set $\mathbb{T}$ during the bootstrapping. With the unchanged set $\mathbb{T}$, which is initialized to contain all prefixes in all possible divisions of all words in corpus $\mathbb{C}$, the bootstrapping algorithm is degenerated to a simple counting function, which, for a given k, returns the top k frequent suffixes. So as to compare with other models more fairly, we didn't implement the mechanisms in the original algorithm for removing complex suffixes such as *-ers*, *-ings* and *-ors*, neither the trick for removing the most noisy suffix *-e*.

4 A generative model with multinomial likelihood

In this section, we are going to describe a generative morphology model that involves one more random variable than the baseline model we sketched in the introduction. In the baseline generative model, we assume without any condition that both stems and suffixes are independently and multinomially distributed. For the current model, stems and suffixes are independently and multinomially distributed in each inflectional class. A morphological analysis of word w can be denoted as (c, t, f), which means that $w = t.f$ and this analysis belongs to inflectional class c.

Assume a multinomial distribution over k^C inflectional classes, with parameters θ^C. To make predictions about new classes, we take symmetric Dirichlet priors α^Con parameters θ^C, which means that the way each inflectional class is used has little variation. When k^C is set as 1 in this model, it degenerates to the baseline generative model that assumes stems and suffixes are independently distributed. Again, let N be the number of words and $I(w = t.f)$ denote the indicator function taking on value 1 when concatenating stem t and suffix f gives word w and 0 otherwise.

Our generative morphology model is like this,

$$
\begin{aligned}
k^C &= \text{number of inflectional classes} \\
\alpha^C, \alpha^T, \alpha^F &= \text{constant hyper-parameters} \\
\theta^C &\sim \text{Symmetric-Dirichlet}(\alpha^C) \\
\theta^T_{i=1...k^C}, \theta^F_{i=1...k^C} &\sim \text{Symmetric-Dirichlet}(\alpha^T), \text{Symmetric-Dirichlet}(\alpha^F) \\
c_{i=1...N} &\sim \text{Multinomial}(\theta^C) \\
t_{i=1...N}, f_{i=1...N} &\sim \text{Multinomial}(\theta^T_{c_i}), \text{Multinomial}(\theta^F_{c_i}) \\
w_{i=1...N} &\sim I(w_i = t.f)P(c_i = c|\theta^C)P(t|c, \theta^T)P(f|c, \theta^F)
\end{aligned}
$$

This is the morphology model of choice in (Goldwater et al., 2011), following which we also use Gibbs sampler, a simple and widely-used Markov Chain Monte Carlo method, for inference. Assume the exchangeability of morphological analyses. The finite set of morphological analyses $\{a_1, ..., a_N\}$ is exchangeable, if for a permutation, π, of the integers from 1 to N,

$$P(a_1, ..., a_N) = P(a_{\pi(1)}, ..., a_{\pi(N)}).$$

At each iteration, from $\mathbf{a} = \{a_1, ..., a_N\}$, sample a'_1 given morphological analyses of all other words, i.e. $A_{-1} = \{a_2, ..., a_N\}$, then go to $\{a'_1, a'_2, ..., a_N\}$ and so on until $\{a'_1, a'_2, ..., a'_N\} = \mathbf{a}'$. It can be shown that this sampling process defines a Markov chain on $\mathbf{a}, \mathbf{a}', \mathbf{a}'', ...$ After a sufficient amount of time, the probability values are independent of the starting values and tend towards the stationary distribution $P(\mathbf{a})$. More about Gibbs sampling will be discussed in Section 5.2, where sampling processes for two generative models are compared.

Related work

The Dirichlet-multinomial model is not able to capture neither the particular statistical property of word frequencies, as studied in (Goldwater et al., 2006), nor the particular statistical property of morphological units, as studied in (Chan, 2008) and (Zhao and Marcus, 2011). As described in Section 3.1, a rule-based bootstrapping algorithm can be designed to take advantage of long-tail distributions of both stems and suffixes. However, as we will show with experimental results in Section 6, the rule-based model performs notably bad with the token-based evaluation, which gives more weights to the results of frequent words than the type-based evaluation.

In a two-stage learning framework proposed in (Goldwater et al., 2006), the morphology model as introduced above is used as a 'generator' for producing words. Word frequencies are then transformed to exhibit power-law by an additional generative process, called 'adaptor'. Especially, a two-parameter generalization of Chinese Restaurant Process (Pitman and Yor, 1997) is used as an adaptor. The Pitman-Yor process implements the principle of preferential attachment and guarantees the exchangability of its outputs. As we will also show in Section 6, the generator, i.e. the Dirichlet-multinomial model, learns reasonably well from the input of distinct word types, i.e. type-based input; however, the multinomial model itself cannot adapt to the input of unprocessed text data, i.e. the token-based input. Augmented with an adaptor, the generator may achieve its best performance with all forms of input; however, the introduction of such an adaptor does not improve the overall performance of the generator. In the following section, we will propose a generative model that learns well from both forms of input: type-based or token-based, without the need of an adaptor. The proposed model is able to utilize token frequency by itself, therefore, we do not need

to concern with justifying the appearance of type frequency in morphology learning, as pursued in (Goldwater et al., 2006).

5 A generative model with lognormal likelihood

We propose to compute lognormal likelihood, instead of multinomial likelihood, for both stems t and suffixes f. In this way, the particular statistical property of stems and suffixes is utilized in a Bayesian model other than a rule-based model. Furthermore, as we discussed in Section 2, given the multiplicative property of lognormal distributions, for each inflectional class, the generated word distribution is also lognormal. Therefore, with the proposed model, we can directly capture the statistical property of word frequencies without the need of an additional generative process.

5.1 The probability model

Again, assume that stems and suffixes are independent given inflectional classes. We still have a multinomial distribution over k^C classes, with parameters θ^C, and take symmetric Dirichlet priors α^C on θ^C. Now, in each inflectional class c, we assume a lognormal distribution of frequency for both stems and suffixes. It is equivalent to assume that the logarithms of stem/suffix frequency are normally distributed over the rank. For example, if the logarithms of stem frequency, $\mathfrak{L}(t)$, is normally distributed with mean μ^T and variance $(\sigma^T)^2$, then stem frequency t is lognormally distributed with mean e^{μ^T} and variance $(\sigma^T)^2$.

For a random variable X that is normally distributed, if both its mean μ and variance σ^2 are random, we will use the following distribution for priors, which can be shown to be conjugate to normal likelihood. Assume μ_0, γ_0, α, and β as the constant hyper-parameters, then we have

$$\begin{aligned}\sigma^2 &\sim \text{Inverse-Gamma}(\alpha, \beta)\\ \mu|\sigma^2 &\sim \text{Normal}(\mu_0, \gamma_0/\sigma^2)\\ x|\mu, \sigma^2 &\sim \text{Normal}(\mu, \sigma^2).\end{aligned}$$

In our case, we construct the probability distributions as follows:

$$\begin{aligned}\theta^C &\sim \text{Dirichlet}(\alpha^C)\\ c_{i=1\ldots N} &\sim \text{Multinomial}(\theta^C)\\ (\sigma^T_{i=1\ldots k^C})^2, (\sigma^F_{i=1\ldots k^C})^2 &\sim \text{Inverse-Gamma}(\alpha^T, \beta^T), \text{Inverse-Gamma}(\alpha^F, \beta^F)\\ \mu^T_{i=1\ldots k^C}, \mu^F_{i=1\ldots k^C} &\sim \text{Normal}(\mu^T_0, \gamma^T_0/(\sigma^T_i)^2), \text{Normal}(\mu^F_0, \gamma^F_0/(\sigma^F_i)^2)\\ t_{i=1\ldots N}, f_{i=1\ldots N} &\sim \text{Log-Normal}(e^{\mu^T_{c_i}}, (\sigma^T_{c_i})^2), \text{Log-Normal}(e^{\mu^F_{c_i}}, (\sigma^F_{c_i})^2)\\ w_{i=1\ldots N} &\sim I(w_i = t.f) P(t|c_i, \mu^T_{c_i}, (\sigma^T_{c_i})^2) P(f|c_i, \mu^F_{c_i}, (\sigma^F_{c_i})^2)\end{aligned}$$

where $\alpha^C, \mu^T_0, \gamma^T_0, \alpha^T, \beta^T, \mu^F_0, \gamma^F_0, \alpha^F$ and β^F are constant hyper-parameters .

5.2 Gibbs Sampling

As discussed in Sect 4, we use Gibbs sampler for the inference of generative models. At each iteration, we need to sample a morphological analysis for each word given the morphological analyses of all other words. For example, at initialization, each word receives a random analysis, then we proceed by sampling a morphological analysis a'_1 of the first word, w_1, given the random

Multinomial	$(c,t,f)\|w_i,\mathbf{A}_{-i} \sim I(w_i = t.f)\frac{\alpha^C+m_c}{k^C\alpha^C+N}\frac{\alpha^T+m_{c,t}}{k^T\alpha^T+m_c}\frac{\alpha^F+m_{c,f}}{k^F\alpha^F+m_c}$
Lognormal	$c\|\mathrm{A}_{-i} \sim \frac{\alpha^C+m_c}{k^C\alpha^C+m_c}$ $(\sigma_c^T)^2\|\mathrm{A}_{-i} \sim IG(\alpha^T+\frac{m_{c,t}}{2},\beta^T+\frac{\sum(\mathfrak{L}(t_i)-\overline{\mathfrak{L}(t)})^2}{2}+\frac{m_{c,t}\gamma_0^T(\overline{\mathfrak{L}(t)}-\mu_0^T)^2}{2(\gamma_0^T+m_{c,t})})$ $\mu_c^T\|(\sigma_c^T)^2,\mathrm{A}_{-i} \sim N(\frac{m_{c,t}\overline{\mathfrak{L}(t)}+\gamma_0^T\mu_0}{m_{c,t}+\gamma_0^T},\frac{\sigma_c^T}{m_{c,t}+\gamma_0^T})$ $(\sigma_c^F)^2\|\mathrm{A}_{-i} \sim IG(\alpha^F+\frac{m_{c,f}}{2},\beta^F+\frac{\sum(\mathfrak{L}(f_i)-\overline{\mathfrak{L}(f)})^2}{2}+\frac{m_{c,f}\gamma_0^F(\overline{\mathfrak{L}(f)}-\mu_0^F)^2}{2(\gamma_0^F+m_{c,f})})$ $\mu_c^F\|(\sigma_c^F)^2,\mathrm{A}_{-i} \sim N(\frac{m_{c,f}\overline{\mathfrak{L}(f)}+\gamma_0^F\mu_0}{m_{c,f}+\gamma_0^F},\frac{\sigma_c^F}{m_{c,f}+\gamma_0^F})$ $(t,f)\|w_i,\mathrm{A}_{-i} \sim I(w_i = t.f)\mathtt{lognormal}(e^{\mu_c^T},(\sigma_c^T)^2)\mathtt{lognormal}(e^{\mu_c^F},(\sigma_c^F)^2).$

Table 2: Sample a morphological analysis of the word w_i.

analyses of all other words, $A_{-1} = \{a_2, a_3, ..., a_N\}$. Then we sample a morphological analysis a_2' of word w_2, given $A_{-2} = \{a_1', a_3, ..., a_N\}$, and so on until it stabilizes.

For the multinomial model, we sample (c,t,f) together, with the following posterior conditional probability, $P((c,t,f)|w_i,A_{-i}) \propto I(w_i = t.f)P(c_i = c|\theta^C,A_{-i})P(t|c,\theta^T,A_{-i})P(f|c,\theta^F,A_{-i})$. Take the second term of the above equation as an example. As a result of our choice of conjugate priors, the posterior distribution, $P(t|c,\theta^T,A_{-i})$, is also multinomial but with a different parameter $\alpha' \sim \text{Dirichlet}(\alpha^F + m_{c,f})$, where $m_{c,f}$ is the number of analyses that contain both inflectional class c and suffix f. Therefore, the second term can be reduced to a form with θ^T integrated out, $P(t_i = t|\mathbf{A}_{-i},c) = \frac{\alpha^T+m_{c,t}}{\alpha_+^T+m_c}$. Similarly reductions can be done to other terms as well and putting together the results, we obtain the conditional probability for sampling (c,t,f) given the current word and morphological analyses of all other words, which is shown at the first row of Table 5.2.

For the lognormal model, we sample the inflection class c first, from the posterior distribution as discussed above. Then we sample (t,f) from its posterior distribution given the sampled c and morphological analyses of all other words. Again, given our choice of conjugate priors, the posterior distributions of stem t and suffix f are still lognormal with updated mean and variance. For the lognormal model, so as to update these parameters, we need to alternatively sample variance and mean from their own posterior distributions respectively. In practice, the updated mean and variance are sampled regarding the normal distribution of stem/suffix frequency's logarithms, $\mathfrak{L}(t/f)$. The specific sampling process is depicted in the second row of Table 5.2, for a comparison with the multinomial model[3]. We won't go into details of computing the updated parameters for sampling new mean and variance[4], but it is worth noticing that the data samples are logarithms of frequencies.

The sampling of lognormal distributions is obviously much more complex and time-consuming than multinomial distributions. We are motivated to constrain the learning of generative models

[3] Table 5.2 is constructed based on one of our reviewers' suggestion.
[4] A detailed discussion can be seen in (Jordan, 2010)

with the acquisition outputs from the rule-based model, which runs very fast by itself. As we will show by experimental results in Section 6, the constrained learning is not only much faster but also significantly improves the performance.

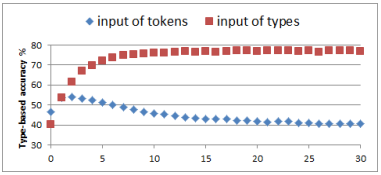
6 Experiments

In this section, we run experiments on unsupervised learning of morphology and compare the approaches we describe in Section 3, 4 and 5. Following (Goldwater et al., 2011), we will learn morphological segmentations for verbs in the WSJ corpus with the input of raw text only. Given the Penn Treebank guidelines, we consider words associated with tags of 'VB', 'VBP', 'VBZ', 'VBD', 'VBN' or 'VBG' as verbs. Using the gold part-of-speech annotations, we extracted 137,899 verbs from the whole WSJ corpus which belong to 7,728 distinct word types. With heuristics based on part-of-speech tags and spellings, we automatically segment each verb into a stem, which cannot be empty, and a suffix, which may be empty, and use these segmentations as gold standards for evaluation.

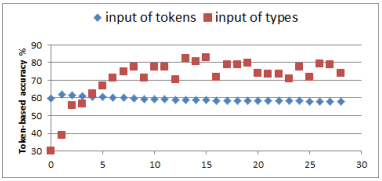
Given the gold analysis of each word, the accuracy of a morphology model can be evaluated in two ways. For a type-based evaluation, we compute the accuracy as the percentage of correctly analyzed word types out of all distinct word types that are ever seen in the corpus. For a token-based evaluation, we compute the accuracy as the percentage of correctly analyzed tokens out of all occurrences of words in the whole corpus. In most previous work on unsupervised learning of morphology, only the type-based evaluation is reported. However, we agree with Goldwater et al. (2006) that the token-based evaluation gives more weights to the results of frequent words, thus reflects better the performance of each approach as applied to real text data.

Different forms of input

In formal study of morphology, the acquisition input is usually taken as the list of distinct word types. For example, as shown in Section 3.1, the bootstrapping algorithm measures contextual diversity with type frequency only. On the other hand, natural text data typically use most types of words more than once. Furthermore, when a model is trained with the input of distinct types only, each word occurrence of the same type will always receive the same analysis by the model. However, if a model is trained with real text data, then with a generative model, a word may receive different analyses on different occurrences.



(a) Type-based evaluation.

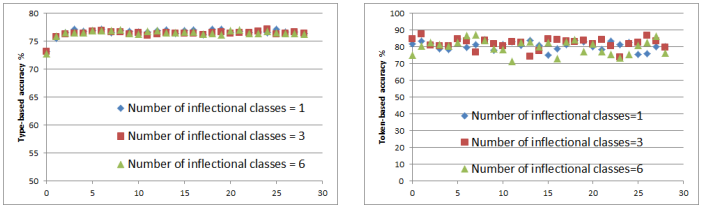


(b) Token-based evaluation.

Figure 3: Experiment with different forms of input.

First, we replicate the experiments in (Goldwater et al., 2006). The morphology model is multinomial as discussed in Section 4. As shown in Figure 3, the multinomial model learns well with the input of distinct word types, but poorly with the token-based input.

Inflectional classes



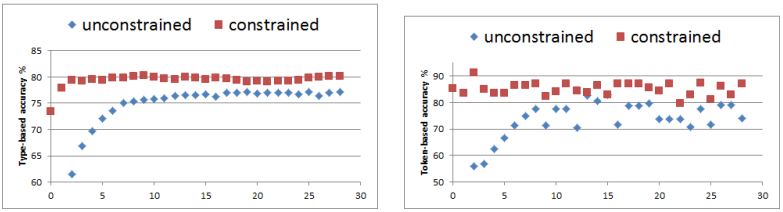
(a) Type-based evaluation. (b) Token-based evaluation.

Figure 4: Experiment with different numbers of inflectional classes.

In the above experiment, the number of inflectional classes is set as 6 following (Goldwater et al., 2006). However, this choice of the number of inflectional classes is rather arbitrary. Therefore, we experiment with different settings of this parameter. As shown in Figure 4, different choices of inflectional classes do not make significant differences in training results.

Constrained Learning

As we described in Section 5, the sampling of the lognormal model is much more complex and time-consuming than the sampling of the multinomial model. Motivated by the concern of training efficiency, we propose to constrain the learning of generative models with the acquisition outputs from the rule-based model that we described in Section 3.1. Since the rule-based model takes input of raw text only and so the generative models, the combination of these two processes results in a totally unsupervised learning process as well. More specifically, suppose that we have acquired a set of suffixes containing *-es* and *-s* only. With constrained learning, the only possible segmentations we need to consider for word *porches* are *porch+es*, *porche+s*, and *porches+""*, instead of all the 7 possible segmentations of this 7-character word. In practice, we constrain the learning of generative models with 20 suffixes acquired by the bootstrapping algorithm.



(a) Type-based evaluation. (b) Token-based evaluation.

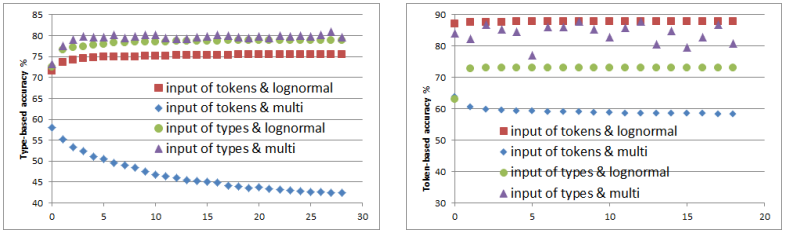
Figure 5: Experiment with constrained learning of multinomial models.

As shown in Figure 5, the constrained learning converges much faster than its unconstrained counterpart. Since the bootstrapping algorithm is very fast by itself, taking almost no time compared to the training of generative models, the total training time is saved a lot. Moreover, even though

this method is motivated by the concern of training efficiency, it also significantly improves the performance. As clearly shown in Figure 5, the constrained learning achieves notably higher performance by both the type-based and token-based evaluations.

The generative model with lognormal likelihood

As discussed in Section 5, by replacing the multinomial likelihood with lognormal likelihood, we can take advantage of the particular statistical property of morphological units with a Bayesian approach; moreover, we can capture the statistical property of word frequency without the need of an additional generating process. We experiment with the lognormal model over different forms of input, and evaluate it by different criteria. Based on the above experimental results, in this experiment, we set the number of inflectional class as 1, and apply the constrained learning with 20 suffixes acquired by the bootstrapping algorithm.



(a) Type-based evaluation.

(b) Token-based evaluation.

Figure 6: Experiment with generative models of multinomial or lognormal distributoins.

Overall, the proposed model performs significantly better than the multinomial model, especially when trained with unprocessed text data. As shown in Figure 6-a, when they are both given the input of distinct types, the results of the lognormal and multinomial models are not distinguishable by the type-based evaluation. However, in contrast to the multinomial model, the lognormal model is able to learn from unprocessed text data as well. As shown in Figure 6-b, by the token-based evaluation, the proposed lognormal model achieves its best performance with the input of text data, which is much more accurate than the best of the multinomial model. It is interesting to observe that the proposed lognormal model is more accurate with the token-based evaluation when trained with token-based learning, but more accurate with the type-based evaluation when trained with type-based learning. This result pattern suggests that the proposed model is able to adapt to real data distributions by itself, without the need of an additional generative process.

Compare all three models

We have shown the acquisition outputs of the bootstrapping algorithm in Table 1, upon which we can build a rule-based segmentation model. In contrast to the learning progress of generative models, which will converge to a relatively steady state, we stop the acquisition process after 20 bootstrapping iterations following previous experiments. Furthermore, so as to compare the generative models with the rule-based model, for each generative model, we compute the average accuracy of its last 5 training iterations.

	input form	type-based evaluation	token-based evaluatoin
bootstrapping	either	**83.59**%	64.04%
multinomial	type-based	79.98%	81.06%
lognormal	type-based	78.85%	73.10%
multinomial	token-based	42.36%	58.06%
lognormal	token-based	75.46%	**87.79**%

Table 3: Compare all three models with different forms of input.

As shown in Table 3, the rule-based model achieves a type-based accuracy as high as 83.59%, significantly higher than any other generative model. However, by the token-based evaluation, the rule-based model performs rather bad. The highest token-based accuracy, 87.79%, is achieved by the lognormal generative model. No matter what form of input is fed to the multinomial model, this level of token-based accuracy cannot be achieved.

7 Conclusion and future work

In previous work on unsupervised learning of morphology, the long-tail pattern observed for the rank-frequency distribution of words, as well as of morphological units, is usually considered as following Zipf's law (power-law). We argue that the signature straight lines on logistic scales may suggest either power-law or lognormal. We have also discussed that both based on the idea of preferential attachment, the generative processes for generating Zipf's law and lognormal distributions have only subtle differences. The advantage of considering the long-tail distributions of morphological units as lognormal is so that we can utilize the statistical property in a Bayesian model. Moreover, given the multiplicative property of lognormal distributions, we can directly capture the long-tail distribution of word frequency without the need of an adaptor.

The experimental results show that the proposed model performs significantly better than other models in discussion, especially when it is evaluated by a token-based criterion that respects more of the real distribution of text data. Moreover, the proposed model can not only learn from the list of distinct word types, which can be handled by other models as well, but also from the unprocessed text data, which cannot be handled by other models. Especially, the proposed generative model is more accurate with the token-based evaluation when trained by token-based learning, and more accurate with the type-based evaluation when trained by type-based learning. This result pattern suggests that the proposed model is able to adapt to real data distribution by itself.

In this work, our primary goal is to provide an alternative perspective on modeling the long-tail distributions for morphology learning, rather than to develop a state-of-the-art morphology learning system. We are aware of recent work on morphology learning that utilize more extra information and achieve good results on more data. Extra information that has been shown to be useful for morphology model includes syntactic context (Lee et al., 2011), document boundaries (Moon et al., 2009) and so on. The proposed model has a potential to be developed as a more complex learning system, thus, in future work, we plan to extend our model to integrate these extra information and compare with more benchmark systems.

References

Chan, E. (2008). *Structures and distributions in morphology learning*. PhD thesis, University of Pennsylvania.

Downey, A. B. (2001). The structural cause of file size distributions. In *Proceedings of the Ninth International Symposium in Modeling, Analysis and Simulation of Computer and Telecommunication Systems*, MASCOTS '01, Washington, DC, USA.

Goldwater, S., Griffiths, T., and Johnson, M. (2006). Interpolating between types and tokens by estimating power-law generators. In *NIPS*.

Goldwater, S., Griffiths, T. L., and Johnson, M. (2011). Producing power-law distributions and damping word frequencies with two-stage language models. *Journal of Machine Learning Research*, 12(Jul):2335–2382.

Jordan, M. I. (2010). The conjugate prior for the normal distribution. Lecture notes on Stat260: Bayesian Modeling and Inference.

Lee, Y. K., Haghighi, A., and Barzilay, R. (2011). Modeling syntactic context improves morphological segmentation. In *Proceedings of the Fifteenth Conference on Computational Natural Language Learning*, Portland, Oregon, USA. Association for Computational Linguistics.

Marcus, M., Santorini, B., and Marcinkiewicz, M. A. (1993). Building a large annotated corpus of english: The penn treebank. *Computational linguistics*, 19(2):313–330.

Miller, G. A. (1957). Some effects of intermittent silence. *American Journal of Psychology*, 70:311–314.

Mitzenmacher, M. (2004). A brief history of generative models for power law and lognormal distributions. *INTERNET MATHEMATICS*, 1:226–251.

Moon, T., Erk, K., and Baldridge, J. (2009). Unsupervised morphological segmentation and clustering with document boundaries. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 2 - Volume 2*, EMNLP '09, pages 668–677, Stroudsburg, PA, USA. Association for Computational Linguistics.

Newman, M. (2005). Power laws, pareto distributions and zipf's law. *Contemporary Physics*, 46(5):323–351.

Pareto, V. (1896). *Cours d' Economie Politique*. Droz, Geneva.

Perline, R. (1996). Zipf's law, the central limit theorem, and the random division of the unit interval. *Physical Review*, 54(1):220–223.

Pitman, J. and Yor, M. (1997). The two-parameter poisson-dirichlet distribution derived from a stable subordinator. *Annals of Probability*, 25(2):855–900.

Simon, H. A. (1955). On a class of skew distribution functions. *Biometrika*, 42(3/4):425–440.

Zhao, Q. and Marcus, M. (2011). Functional elements and pos categories. In *Proceedings of 5th International Joint Conference on Natural Language Processing*, pages 1198–1206, Chiang Mai, Thailand. Asian Federation of Natural Language Processing.

Zipf, G. K. (1949). *Human Behaviour and the Principle of Least Effort*. Addison-Wesley, Reading, MA.

User Behaviors Lend a Helping Hand: Learning Paraphrase Query Patterns from Search Log Sessions

Shiqi Zhao[1,2] *Haifeng Wang*[1] *Ting Liu*[2*]
(1) Baidu, Beijing, China
(2) School of Computer Science and Technology, Harbin Institute of Technology, Harbin, China
zhaoshiqi@baidu.com, wanghaifeng@baidu.com, tliu@ir.hit.edu.cn

ABSTRACT
Search log sessions contain a large number of paraphrases contributed by users during query rewriting. However, it is a big challenge to distinguish paraphrases from the simply related queries in the sessions. This paper addresses this problem by making innovative use of user behavior information embodied in query sessions. Specifically, we learn paraphrase patterns from the search log sessions with a classification framework, in which three types of user behavior features are exploited besides the conventional features. We evaluate the method using a query log of a commercial search engine. Experimental results demonstrate the effectiveness of our method, especially the significant contribution of the user behavior features. We extract over 250,000 pairs of paraphrase patterns from the used search log, with a precision over 76%.

TITLE AND ABSTRACT IN ANOTHER LANGUAGE (CHINESE)

基于用户行为特征从搜索会话中挖掘复述查询模板

搜索引擎用户在搜索时经常会对查询进行同义改写，以获得更好的搜索结果，因此搜索日志的查询会话中包含大量的复述资源。然而一个困难的问题是如何区分查询会话中哪些查询是复述关系，哪些则仅仅是语义相关而已。本文通过更好地利用查询会话中隐含的用户行为特征来解决这一问题。具体地，我们基于分类的方法从查询会话资源中挖掘复述查询模板。在分类模型中，我们不仅使用了传统的复述识别特征，还尝试了三种用户行为特征。我们使用一个商业搜索引擎的用户搜索日志来评估本文提出的方法。评估结果证明了本文方法的有效性，尤其是用户行为特征起到的明显作用。我们从所使用的用户日志中共抽取出了超过250,000对的复述模板，其准确率超过76%。

KEYWORDS: paraphrase, pattern, query, search log.

KEYWORDS IN CHINESE: 复述, 模板, 查询, 搜索日志.

* Corresponding author

Proceedings of COLING 2012: Technical Papers, pages 3137–3152,
COLING 2012, Mumbai, December 2012.

1 Introduction

Paraphrases have been shown to be useful in plenty of areas, such as machine translation (MT) (Callison-Burch et al., 2006; Madnani et al., 2007; Marton et al., 2009), question answering (QA) (Lin and Pantel, 2001; Ravichandran and Hovy, 2002; Duboue and Chu-Carroll, 2006; Riezler et al., 2007), and web search (Zukerman and Raskutti, 2002). In particular, the capability of paraphrasing is essential in web search, since in many cases the user queries need to be paraphrased so as to improve the quality of the search results.

This paper focuses on learning paraphrase query patterns from search log sessions, which could be useful in various applications, especially in query paraphrasing. Although search log sessions have been extensively exploited for mining related queries, this is the first work, as far as we know, to learn paraphrases from this data source. Mining paraphrase query patterns and related queries are both useful for search engines, but in different aspects. On the one hand, related queries can be used for query suggestion and recommendation, using which the users can extend their search interest and get some related information. However, the related queries are not suitable for direct query rewriting with the purpose of retrieving more and better results exactly reflecting the user's requirement, since the related queries often have different meanings from the original user query. On the other hand, the paraphrase query patterns are mined for query paraphrasing, which is to directly rewrite user queries during search. This is useful especially in the cases where the original user queries contain some uncommon words that need to be rewritten into more common expressions with the same meaning. The following examples show the difference between them:

Related queries:
q_1: 凯文 杜兰特 的 身高 是 多少 *(what is the height of Kevin Durant)*
q_2: 凯文 杜兰特 的 体重 是 多少 *(what is the weight of Kevin Durant)*
Paraphrase queries:
q_1: 凯文 杜兰特 的 身高 是 多少 *(what is the height of Kevin Durant)*
q_2: 凯文 杜兰特 的 真实 身高 *(true height of Kevin Durant)*

In a nutshell, our method involves two steps. In the first step, we induce candidate pattern pairs from query pairs co-occurring within the same query sessions, while in the second step, we recognize paraphrase patterns from the candidates using a classifier. We investigate comprehensive features in paraphrase recognition, including not only conventional features based on string similarities, but also novel features based on user behaviors. In detail, we design three types of user behavior features, namely, (1) the frequency based features for measuring the co-occurrence frequency between two patterns within sessions, (2) the lexical score features for estimating the lexical level paraphrasing likelihood, and (3) the distance based features for measuring the separation distance between queries in sessions.

We conduct experiments using a Chinese query log from Baidu[1], a commercial search engine. The results show that the classification based approach is effective in paraphrase recognition. Particularly, the user behavior features can significantly enhance the classification performance. More than 250,000 pairs of paraphrase patterns are learned from the used search log with a precision over 76%, which suggests that the search log sessions are rich in high-quality paraphrases. Furthermore, we find that our method is complementary to a previous method learning paraphrases from query-click pairs of query logs, which inspires us to integrate them in our future research.

[1] www.baidu.com

In what follows, we first review related studies in Section 2. We then introduce the pattern pair induction method in Section 3. The paraphrase pattern recognition method is proposed in Section 4, in which the user behavior features are described in detail. We present the experiment results in Section 5 and conclude the paper in Section 6.

2 Related Work

2.1 Paraphrase Learning

Plenty of methods have been proposed to extract paraphrases from various data sources. In (Barzilay and McKeown, 2001), the authors viewed multiple translation versions of the same literary works as monolingual parallel corpora and extracted paraphrases with a co-training algorithm. In (Barzilay and Lee, 2003; Dolan et al., 2004), researchers collected comparable news articles reporting on the same event, and further extracted parallel sentences for learning paraphrase phrases and patterns. There are also studies focusing on extracting paraphrases from large-scale monolingual corpora based on distributional hypothesis (Lin and Pantel, 2001; Bhagat and Ravichandran, 2008). The basic idea is that phrases or patterns appearing in similar contexts tend to have the same meaning.

Besides monolingual corpora, bilingual corpora have also been exploited for paraphrase extraction. Bannard and Callison-Burch (2005) first presented the method to learn paraphrase phrases from a bilingual phrase table. The key idea is that phrases aligned with the same foreign phrase could be paraphrases. Callison-Burch (2008) then improved the method by imposing syntax constraints to filter paraphrases with different syntactic structures. In addition, Zhao et al. (2008) extended this method to paraphrase pattern extraction.

To our knowledge, few studies have been conducted on learning paraphrases from query logs. Zhao et al. (2010)'s study might be the closest to our work. Their method is motivated by the assumption that user queries and the clicked titles are potential paraphrases. Accordingly, they train a classifier to recognize paraphrases from query-title pairs. They further extract query-query and title-title paraphrases from the query-title paraphrases based on the assumption that queries clicking on the same title and titles clicked on for the same query are also likely to be paraphrases. Additionally, they induce paraphrase patterns from the mined paraphrases. Our work differs from Zhao et al.'s mainly in that we learn paraphrase patterns from query sessions instead of query-click pairs. We compare these two methods in the experiments (Section 5.3).

2.2 Search Log Mining

Search engine query logs have been extensively exploited. Especially, there is a large body of research focusing on mining related queries from search logs and applying them in query rewriting and recommendation. Such research can be classified into three groups. The first group of methods utilizes user clicks when computing query similarity. The underlying assumption is that if users tend to click on similar documents for two queries, then the meanings of the queries should be similar (Wen et al., 2002; Baeza-Yates and Tiberi, 2007). In the second group of methods, researchers mine query rewriting terms directly from user clicked documents. Their basic idea is that terms from queries and user clicked documents are related (Cui et al., 2002; Riezler et al., 2008). The third group of methods learns related queries from query sessions. The assumption is that queries submitted by the same user within a short time might be related in meaning (Fonseca et al., 2005; Jones et al., 2006; Zhang and Nasraoui, 2006; Szpektor et al., 2011). Our work is close to the third group. However, what we learn are

paraphrase patterns rather than related queries or patterns.

3 Pattern Pair Induction

3.1 Concepts

Query. In this work, we collect user queries and other useful information from the used search logs and represent a query q as a triplet:

$$q = \langle qc, qt, cn \rangle,$$

where qc denotes the query content, qt is the time when qc is searched, and cn ($\geq$0) denotes the number of results clicked by the user. All queries in the search log are first preprocessed, including word segmentation, Part-of-Speech (POS) tagging, and Named Entity (NE) recognition.

Session. A query session (QS) is a sequence of queries submitted by the same user within a short time. We represent a query session with n queries as:

$$QS = q_1 ... q_i ... q_n,$$

where q_i is a query described above. Note that the order of queries in a session does matter, which records the sequence of user actions.

Pattern. A pattern is composed of two parts, i.e., pattern words and slots. Pattern slots can be instantiated with different words that meet certain constraints. We represent pattern slots as POS tags, which means that each slot can be instantiated by words with the specified POS. In our experiments, words with five kinds of POSes are allowed to form slots, including *noun* (n), *verb* (v), *adjective* (a), *numeral* (m), and *time* (t).

3.2 Induce Pattern Pairs

The reason why we learn paraphrase query patterns rather than directly extracting paraphrase queries is that paraphrase patterns usually achieve a higher coverage in applications than paraphrase instances. In addition, query patterns are much less sparse than queries. This work induces pattern pairs from query pairs co-occurring within the same query sessions. In detail, from a session $QS = q_1 ... q_i ... q_n$, we first extract all query pairs $\langle q_i, q_j \rangle$ ($1 \leq i < j \leq n$) in which q_i and q_j share at least one identical word. We then replace one or more shared identical word with their POS tags (slots), and thereby generate pattern pairs. It is obvious that we may induce more than one pattern pair from a query pair by selecting different slots. In addition, we assign a unique number to each pair of aligned slots in a pattern pair to distinguish slots with identical POS tags.

Figure 1 shows an example of pattern pair induced from a query pair. The slot $[n-1]$ denotes that the slot is the first slot in the pattern and the POS of fillers should be noun (n). In practice, we constrain that each pattern contains at least one content word besides slots, so as to filter meaningless patterns. In addition, since queries are mostly short, we constrain that each pattern contains at most two slots. We aggregate pattern pairs induced from all sessions in the search log and sum up the frequencies for each pair. Pattern pairs satisfying the following requirement are retained: (1) the frequency of the pattern pair exceeds a threshold T_1, (2) the number of unique fillers for each slot exceeds a threshold T_2.

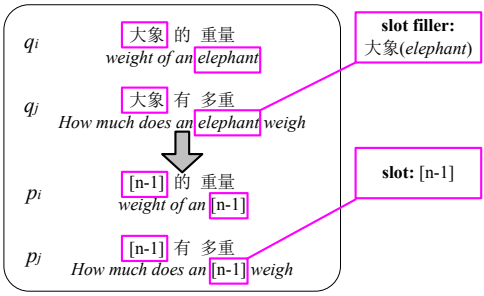


Figure 1: Example of pattern pair induction.

4 Paraphrase Pattern Recognition

Following the previous studies (Brockett and Dolan, 2005; Finch et al., 2005; Malakasiotis, 2009), we recast paraphrase pattern recognition as a classification problem. Each induced pattern pair is classified into one of the two classes, i.e., *paraphrase* and *non-paraphrase*. A Support Vector Machines (SVM) classifier is used in our experiments, since it has proven effective in this task (Brockett and Dolan, 2005; Finch et al., 2005). Our classification features can be divided into two groups: the baseline features examined in previous studies (Section 4.1) and user behavior based features proposed in this work (Section 4.2).

4.1 Baseline Features (F_{BL})

Conventional features for paraphrase recognition include three classes, i.e., lexical features, syntactic features, and semantic features. The lexical features measure the surface similarity between word sequences. Syntactic features compute the structural similarity between parse trees. Semantic features measure deep semantic relatedness based on some external knowledge base, such as WordNet in English. Our baseline features are mostly lexical features. In detail, given a pair of candidate patterns p_1 and p_2, our baseline features include: (1) length ratio feature, computed as the length of the shorter pattern divided by that of the longer one, (2) edit distance feature, computed as described in (Zhao et al., 2010), (3) cosine similarity feature, in which the words are weighted based on tf-idf, (4) word overlap rate feature, (5) character overlap rate feature[2], and (6) named entity feature, a boolean feature indicating whether p_1 and p_2 contain identical named entities.

We do not use syntactic features because most of user queries are not well-formed sentences but short n-grams, which cannot be parsed. We do not employ semantic features, either, since the underlying semantic knowledge bases are language-dependent.

4.2 User Behavior Features (F_{UB})

The most distinguishing characteristic of the query log, compared with other data sources, is that it contains rich information about users' searching and browsing behaviors, which could be

[2]Chinese words are composed of characters and words with the same meaning often contain similar characters.

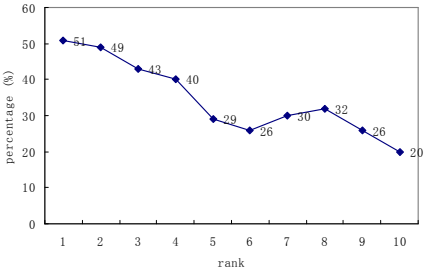


Figure 2: Percentage of paraphrases at each rank.

useful features for recognizing paraphrases. In this work, we design three types of user behavior features based on the observation and analysis of user behaviors from different aspects, which are detailed as follows.

Frequency based Features (F_{fr}). According to our observation, pattern pairs that frequently co-occur within sessions are more likely to be paraphrases. We substantiate our observation with an experiment, in which we randomly sampled 100 patterns with at least 10 candidate paraphrase patterns. We ranked all candidates according to their co-occurrence frequency with the target pattern and kept top-10. The 1000 pattern pairs are manually evaluated and the percentage of paraphrases at each rank is depicted in Figure 2. As can be seen, the percentage of paraphrases decreases as the rank gets lower. We therefore design the frequency based feature as:

$$F_{fr}(p_1, p_2) = \frac{freq(p_1, p_2)}{\sum_p freq(p_1, p) + C_1}, \tag{1}$$

where $freq(p_1, p_2)$ is the frequency of $\langle p_1, p_2 \rangle$ on the whole set of pattern pairs, C_1 is a constant parameter used to avoid overestimating the feature value when p_1 is too infrequent. We also compute the frequency based feature in the other direction, i.e., $F_{fr}(p_2, p_1)$ in the same way.

Lexical Score Features (F_{ls}). Inspired by lexical weight features used to measure phrase pair quality in machine translation (Koehn et al., 2003), we introduce lexical score features to measure the lexical level paraphrase likelihood of each pattern pair. We design a lexical scoring approach based on the observation that many words keep unchanged when users rewrite their queries within sessions. It is reasonable to assume that those unchanged words across queries should exclusively align with themselves, while the changed words may likely form paraphrase word pairs. Accordingly, given a pair of related queries $q_1 = w_{11}...w_{1m}$ and $q_2 = w_{21}...w_{2n}$ extracted from the same session, we compute two scores for any word pair $\langle w_{1i}, w_{2j} \rangle$ $(1 \le i \le m, 1 \le j \le n, w_{1i} \ne w_{2j})$ from them, namely, a *positive* score $a_+(w_{1i}, w_{2j})$ and a *negative* score $a\_(w_{1i}, w_{2j})$. Suppose W is the set of identical words shared by q_1 and q_2, and $l = |W|$, we compute $a_+(w_{1i}, w_{2j})$ and $a\_(w_{1i}, w_{2j})$ as:

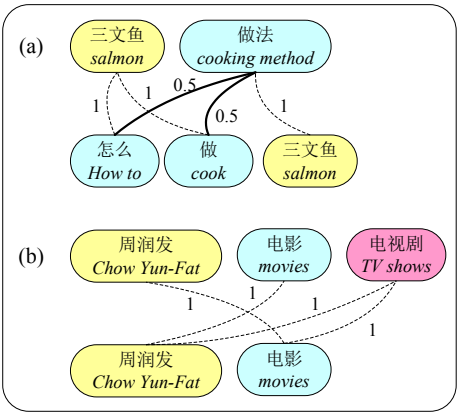


Figure 3: Examples of lexical scoring.

$$a_+(w_{1i}, w_{2j}) = \begin{cases} 0 & if\ w_{1i} \in W \vee w_{2j} \in W \\ \frac{1}{(m-l)*(n-l)} & otherwise \end{cases}$$

$$a_-(w_{1i}, w_{2j}) = \begin{cases} 1 & if\ w_{1i} \in W \vee w_{2j} \in W \\ 0 & otherwise \end{cases}$$

It can be interpreted that, if a word w in q_1 also appears in the other q_2, then w cannot *align* with other words in q_2 (i.e., such alignment gets a negative score). Otherwise, w will get an equal likelihood to *align* with each word in q_2 (i.e., gets a positive score). Examples in Figure 3 illustrate the lexical scoring process. A solid line denotes a positive alignment (a_+), whereas a dashed line denotes a negative alignment (a_-). The a_+ and a_- scores are also given in the figure. As can be seen, the lexical scoring approach assigns a_+ scores to the potential paraphrases (e.g., *cooking method* and *how to cook*) and a_- scores to the incorrectly aligned pairs (e.g., *movies* and *TV shows*).

We sum up a_+ and a_- scores for each word pair w_{1i} and w_{2j} over all the extracted query pairs and compute the lexical score $LS(w_{1i}, w_{2j})$ as follows:

$$LS(w_{1i}, w_{2j}) = \frac{\sum a_+(w_{1i}, w_{2j})}{\sum a_+(w_{1i}, w_{2j}) + \sum a_-(w_{1i}, w_{2j}) + C_2}, \quad (2)$$

where C_2 is a smoothing parameter. At run time, for a pattern pair $\langle p_1, p_2 \rangle$, we ignore slots, stop words, and the shared identical words from two patterns. Suppose the left words are

$p_1' = w_{11}...w_{1m}$ and $p_2' = w_{21}...w_{2n}$, we define the lexical score feature $F_{ls}(p_1, p_2)$ as follows:

$$F_{ls}(p_1, p_2) = \frac{1}{n}\sum_{j=1}^{n} \max_{1 \le i \le m} LS(w_{1i}, w_{2j}). \quad (3)$$

We compute feature $F_{ls}(p_2, p_1)$ in the same way.

Distance based Features (F_{dis}). It is obvious that we should not treat all query pairs from a session equally. Our observation of the user behaviors reveals that queries close to each other in a session are more likely to be paraphrases than those far apart. Therefore, given a pair of related queries $\langle q_i, q_j \rangle$ from a session QS, we measure the distance between two queries from three aspects:

- Query based distance dq. dq is defined as the number of queries between q_i and q_j in the session QS:
$$dq(q_i, q_j) = j - i - 1. \quad (4)$$
- Click based distance dc. The motivation here is that if a user clicks on a few retrieved results, it is likely that the user finds related information from the current results, and it is therefore less likely for her to further paraphrase the query. Given the number of clicks cn_k for query q_k, dc is defined as the sum of clicks between q_i and q_j:
$$dc(q_i, q_j) = \sum_{k=i}^{j-1} cn_k. \quad (5)$$
- Search time based distance dt. Given the timestamps qt_i and qt_j for queries q_i and q_j, dt is defined as the search time interval between q_i and q_j in the session QS:
$$dt(q_i, q_j) = qt_j - qt_i. \quad (6)$$

The distance of a pattern pair $\langle p_1, p_2 \rangle$ is defined as the average distance between query pairs from which $\langle p_1, p_2 \rangle$ is induced. Let $dx(p_1, p_2)$ be the distance between two patterns, we define the distance based feature as: $F_{dx}(p_1, p_2) = exp\{-dx(p_1, p_2)\}$, which guarantees that: (1) the smaller the distance, the larger the feature value, and (2) the feature values vary in the range (0,1].

5 Experiments

In our experiments, we used a query log from Baidu, a Chinese commercial search engine for extracting paraphrase query patterns. The queries were first segmented into sessions using an algorithm based on both time interval and content relatedness. A total of 87,744,130 sessions, containing 362,994,092 queries, were collected from the used query log after removing sessions with only one query. Query preprocessing, including word segmentation, POS tagging, and NE recognition[3], was performed using toolkits implemented based on the state-of-the-art models.

[3]Our NE classes include not only conventional classes like *person*, *location*, *organization*, *numeral*, and *time*, but also some classes frequently occur in user queries, including *movie*, *tv show*, *song*, *novel*, *brand*, *video game*, and *software*.

	P (%)	R (%)	F (%)
F_{BL}	73.09	57.88	64.45
$F_{BL}+F_{fr}$	73.84	63.01*	67.90*
$F_{BL}+F_{ls}$	75.19*	61.87*	67.82*
$F_{BL}+F_{dis}$	73.96	63.90*	68.48*
$F_{BL}+F_{fr}+F_{ls}$	75.85*	64.32*	69.54*
$F_{BL}+F_{fr}+F_{ls}+F_{dis}$	**76.39***	**67.36***	**71.54***

Table 1: P/R/F under different feature combinations. "*" indicates that the improvement is significant ($p < 0.05$) compared with the classifier using only baseline features.

We induced 868,243 pattern pairs from the sessions as described in Section 3. Note that, in practice, we eliminated pattern pairs in which one pattern subsumes the other, i.e., the case of expansion or reduction, as well as the pairs in which two patterns only have some trivial differences, such as inserting or deleting a stop word. The libsvm toolkit[4] was used as the classifier, with its default parameter settings. Some other parameters used in our method were set empirically: $T_1 = 5$, $T_2 = 3$, $C_1 = 20$, $C_2 = 10$.

5.1 Evaluation of the Classifier

We randomly sampled 5115 candidate pattern pairs to form the experimental data set. Two Chinese native speakers were asked to annotate the pattern pairs separately. A pattern pair should be annotated as *positive* (correct paraphrase patterns) or *negative* (otherwise). We follow the instance-based evaluation approach proposed by Szpektor et al. (2007). Particularly, we provide pattern slot fillers to the annotators along with the pattern pairs. A pattern pair is judged as paraphrase only when most of the instances generated by filling the slots with the provided fillers are paraphrases. We calculated the annotation agreement between two annotators. The result shows that the observed agreement is 0.96 and the Kappa value is 0.90. We believe that the high annotation agreement is due to the careful training of the annotators and the instance-based evaluation approach. A third annotator was asked to decide the final annotation for the disagreed pattern pairs.

To evaluate the classifier, we ran 5-fold cross validation with the human annotated data, in which we used 4/5 of the data for training and the rest 1/5 for testing in each run. The evaluation criteria are precision (P), recall (R), and f-measure (F) with regard to the positive class. The average P, R, and F of the classifier under different feature combinations over five runs are reported in Table 1.

The first line of table 1 shows the classification performance when we only use the baseline features. Lines 2-4 summarize the performance when we add each type of user behavior features separately. As can be seen, all the user behavior features can significantly improve the recall and f-measure. This is not surprising, since many paraphrase patterns are not similar enough at the surface level. The user behavior features supply additional evidences for measuring the semantic closeness between patterns, which help to recognize more paraphrase patterns with larger surface difference. Furthermore, lines 5-6 show that the classification performance keeps improving when the user behavior features are added one by one. We achieve the highest P, R, and F when all three types of user behavior features are used. This result indicates that the

[4] downloaded from: www.csie.ntu.edu.tw/∼cjlin/libsvm/.

	P (%)	R (%)	F (%)
all features	76.39	67.36	71.54
w/o F_{dq}	76.49	67.36	71.58
w/o F_{dc}	76.53	67.24	71.54
w/o F_{dt}	76.60	67.36	71.64
w/o F_{dq}+F_{dc}	76.74	67.42	71.71
w/o F_{dq}+F_{dt}	76.45	67.24	71.50
w/o F_{dc}+F_{dt}	75.42	64.14*	69.26*

Table 2: Analysis of the distance based features. "*" indicates that the degradation is significant ($p < 0.05$) compared with the classifier using all features.

user behavior features improve the performance from different aspects, it is thus necessary to combine them together.

We can find from line 3 of Table 1 that the lexical score features also significantly improve the precision of the classifier, which implies that this type of features is useful for filtering noise. We find after analyzing the data that, query pairs like "贾斯汀比伯 身高 (*Justin Bieber height*)" and "贾斯汀比伯 身高 体重 (*Justin Bieber height and weight*)" are quite common in users' search sessions, in which a query is expanded by adding a word closely related to the original query words. As a result, many closely related non-paraphrase word pairs, like 身高 (*height*) and 体重 (*weight*), get large $a\_$ scores and thereby penalized by the lexical score features. That's the main reason why the precision can be enhanced using this type of features.

People may wonder if our lexical score features can outperform the lexical weight features used in MT. For comparison, we implemented the latter on our data set. In detail, we conducted word alignment on the candidate pattern pairs and computed lexical weights in two directions as proposed in (Koehn et al., 2003). We then replaced our lexical score features with the lexical weights and evaluated the classifier via 5-fold cross validation. The average performance is: P=74.78%, R=67.30%, F=70.76%. As can be seen, the performance is lower than the current result (See the last line of Table 1), especially in precision. This result suggests that the lexical weighting approach in MT is unsuitable in paraphrase recognition. The main reason, we believe, is that it is unable to penalize the closely related non-paraphrases as our approach does.

In addition, since the distance based features are defined from three different aspects, we need to evaluate the individual contributions of these features. To this end, we omitted one or two distance based features from several runs and analyzed the influence. The results are given in Table 2. It is interesting to find from lines 2-4 of the table that, there is no degradation in performance when we omitted each distance based feature separately, which suggests that the features may not be independent of each other. Further results can be found from lines 5-7, where we omitted two features together each time. We can see that eliminating the click based and search time based distance features together (F_{dc}+F_{dt}) leads to an obvious degradation in recall and f-measure, while the query based distance feature (F_{dq}) seems helpless. It also implies that the F_{dc} and F_{dt} features follow the same trend. Actually, it is likely that the user's requirement has already been satisfied by the current search results if she spends a long time clicking on and browsing the results. It is therefore less possible for her to paraphrase the query any more. In other words, users' clicking and browsing behaviors are good indicators for recognizing paraphrases.

Phrase Substitution (42.80%)	
p_1: [n-1] 词汇 (*[n-1] vocabulary*)	p_2: [n-1] 词语 (*[n-1] words*)
p_1: [n-1] 为什么 坐牢 (*why was [n-1] imprisoned*)	p_2: [n-1] 为什么 入狱 (*why was [n-1] jailed*)
Information +/- (38.22%)	
p_1: 诺基亚手机 [n-1] (*Nokia mobile phone [n-1]*)	p_2: 诺基亚 的 [n-1] (*Nokia [n-1]*)
p_1: [n-1] [n-2] 考试成绩 (*[n-1] [n-2] exam results*)	p_2: [n-1] [n-2] 的 成绩 (*[n-1] [n-2] results*)
Spelling Correction (12.04%)	
p_1: [n-1] [v-2] 那里 (*where is [n-1] [v-2]*)	p_2: [n-1] [v-2] 哪里 (*where is [n-1] [v-2]*)
p_1: [n-1] tongyici (*[n-1] synonyms*)	p_2: [n-1] 同义词 (*[n-1] synonyms*)
Complex Paraphrase (6.94%)	
p_1: [n-1] 重要 吗 (*Is [n-1] important?*)	p_2: [n-1] 的 重要性 (*importance of [n-1]*)
p_1: 如何 治疗 [n-1] (*how to cure [n-1]*)	p_2: [n-1] 的 治疗方法 (*treatment of [n-1]*)

Table 3: Examples of the extracted paraphrase patterns under different types.

5.2 Evaluation of the Paraphrase Patterns

We used all the 5115 pairs of human annotated patterns to train a classifier, which was then applied to recognize paraphrase patterns from the candidate pattern pairs. A total of 252,963 pairs of paraphrase patterns were extracted in this way. Our statistics show that the average length of the patterns is 2.78 (words), which is mainly because the user queries are mostly short. One may argue that the short paraphrase patterns cannot cover long queries in applications such as query paraphrasing. We believe this problem can be alleviated by allowing partial match of patterns when applying them on long queries. Further statistics show that, over 77% of the paraphrase patterns contain only one slot, and over 90% slots are noun slots.

We randomly sampled 1000 pairs of paraphrase patterns for human assessment. The result shows that the precision is 76.4%. Typological analysis shows that the correct paraphrase patterns can be classified into four groups, including (1) phrase substitution, (2) adding or removing (+/-) information that does not change the meaning, (3) spelling correction, and (4) complex paraphrases, involving both word replacement and structural transformation. The distributions and examples are depicted in Table 3.

As can be seen, phrase substitution is the most represented class, with over 42% of all paraphrase patterns. This is consistent with the statistics reported in (Zhao et al., 2008). There are also quite a few paraphrase patterns in the class "information +/-", reflecting that many users tend to add or remove information to refine their queries while preserving the meaning. The third class, i.e., spelling correction was not conventionally regarded as a type of paraphrase. However, pattern pairs of this class actually convey the same requirement of users and are useful in applications such as query correction. Complex paraphrases are scarce compared with the other

three classes, suggesting that users do not often dramatically transform their queries if their requirement does not change. Note that, paraphrase patterns of the type *spelling correction* can only be applied in one direction, namely, to paraphrase the incorrect queries to the correct forms, while the other three types are not sensitive to the direction. Our method is not likely to learn paraphrase patterns with wrong direction, as people seldom paraphrase a correct query to an incorrect one in the search sessions.

It is interesting to find out to what extent the paraphrase patterns are dependent on the slot fillers. Our analysis shows that more than 76% of the paraphrase patterns are context-independent, which means that the pattern pairs convey the same meaning no matter what words instantiate the slots. For the other 24% of paraphrase patterns, the paraphrase relationship holds only under certain contexts. For example, the pattern pair "什么 [n-1] 好 (*what [n-1] is good*)" and "什么 [n-1] 好吃 (*what [n-1] is delicious*)" can be viewed as paraphrases only when the slot is filled with a food name. Judging in what context the paraphrase patterns can be applied is important in applications, which will be left in our future work.

5.3 Comparison Experiments

In this section, we compare our method with the method proposed in (Zhao et al., 2010), which is referred to as Zhao-10 hereafter. As mentioned above, Zhao-10 makes use of the click-through relationship between queries and clicked document titles from query logs. In particular, Zhao-10 learns paraphrase patterns in two steps. In the first one, it extracts candidate paraphrases from three sources, namely, pairs of queries and clicked titles, pairs of queries clicking on the same title, and pairs of titles clicked on for the same query. A classifier is employed to filter the candidates from each source. Then in the second step, it induces paraphrase patterns from each pair of paraphrases $\langle p_1, p_2 \rangle$ by abstracting one word shared by p_1 and p_2 as slot [X].

We ran Zhao-10 on our search log data and extracted 53,198 pairs of paraphrase patterns after removing the ones with only trivial differences between each other. We randomly selected 1000 pairs for manual annotation, and the result shows that the precision is 82.6%. Comparing this result with that reported in Section 5.2, we can find that the paraphrase patterns extracted with Zhao-10 are more precise than those extracted with the method proposed in this paper, but the quantity is smaller. We also analyze the types of the extracted correct paraphrase patterns as we do in section 5.2. The analysis result suggests that the paraphrase patterns extracted with Zhao-10 can also be classified into the four types as mentioned above, but the distribution is quite different: (1) phrase substitution: 82.13%, (2) information +/-: 9.79%, (3) spelling correction: 2.20%, (4) complex paraphrase: 5.88%.

Furthermore, we examined the intersection between paraphrase patterns extracted with our method and Zhao-10[5]. The result shows that the intersection is extremely small. Only 707 pairs of paraphrase patterns were extracted with both methods. This result implies that our method and Zhao-10 are quite complementary. It is promising to integrate these two methods for paraphrase extraction, whereby we can make full use of the search log information, including queries, clicks, and sessions.

[5]Since the pattern slots are not specified with POS tags in Zhao-10, we did not consider POS mismatch when counting the intersection of two sets of paraphrase patterns.

5.4 Analysis of Portability

People may wonder whether the proposed method can be extended to other languages or other applications. Here we analyze the portability of the method from two aspects:

Language portability. In our experiments, we used preprocessing tools to process Chinese query logs, which include word segmentation, POS tagging, and NE recognition (NER). Although these modules were implemented to process Chinese, the underlying algorithms and models can be language independent. However, please note that the models should be trained with annotated query data, since models trained with normal sentences or texts usually do not work well if they are directly applied on query corpora. Especially, in the NER module, we automatically mine new candidate NEs from the query logs everyday and update the NE dictionary, so as to handle the emerging NEs in user queries. To summarize, the preprocessing modules can be implemented based on language-independent models, but they should be specially trained and adapted for query data.

Application portability. The method is designed for mining paraphrase queries, which could then be used in query rewriting. However, the mined paraphrase patterns can also be used in other applications, especially the paraphrases with the type "phrase substitution", which we believe can be used in sentence rewriting and sentence similarity computation (i.e., matching paraphrases from two sentences when computing their similarity). We will examine the usefulness of the mined paraphrases in other applications in our future experiments.

6 Conclusions and Future Work

This paper proposes a classification-based method for learning paraphrase query patterns from search log sessions. The following conclusions can be drawn from the experiment results. Firstly, we for the first time demonstrate that search log session data is a rich source for extracting paraphrase patterns. Secondly, the classification-based method is effective in paraphrase pattern extraction. Especially, the proposed user behavior features evidently improve the classification performance. Thirdly, our method and the click-through based method (Zhao et al., 2010) are complementary. In our future work, we will improve our paraphrase extraction model by taking advantages of both the query session and click-through knowledge from search logs. In addition, we will try to automatically classify the extracted paraphrase patterns into different types (see Table 3), so as to suit different applications.

Acknowledgments

This work was supported by 863 State Key Project (Grant No. 2011AA01A207), National Natural Science Foundation of China (Grant No. 61073126, 61133012).

References

Baeza-Yates, R. and Tiberi, A. (2007). Extracting semantic relations from query logs. In *Proceedings of KDD*, pages 76–85.

Bannard, C. and Callison-Burch, C. (2005). Paraphrasing with bilingual parallel corpora. In *Proceedings of ACL*, pages 597–604.

Barzilay, R. and Lee, L. (2003). Learning to paraphrase: An unsupervised approach using multiple-sequence alignment. In *Proceedings of HLT-NAACL*, pages 16–23.

Barzilay, R. and McKeown, K. R. (2001). Extracting paraphrases from a parallel corpus. In *Proceedings of ACL/EACL*, pages 50–57.

Bhagat, R. and Ravichandran, D. (2008). Large scale acquisition of paraphrases for learning surface patterns. In *Proceedings of ACL-08: HLT*, pages 674–682.

Brockett, C. and Dolan, W. B. (2005). Support vector machines for paraphrase identification and corpus construction. In *Proceedings of IWP*, pages 1–8.

Callison-Burch, C. (2008). Syntactic constraints on paraphrases extracted from parallel corpora. In *Proceedings of EMNLP*, pages 196–205.

Callison-Burch, C., Koehn, P., and Osborne, M. (2006). Improved statistical machine translation using paraphrases. In *Proceedings of HLT-NAACL*, pages 17–24.

Cui, H., Wen, J.-R., Nie, J.-Y., and Ma, W.-Y. (2002). Probabilistic query expansion using query logs. In *Proceedings of WWW*, pages 325–332.

Dolan, B., Quirk, C., and Brockett, C. (2004). Unsupervised construction of large paraphrase corpora: Exploiting massively parallel news sources. In *Proceedings of COLING*, pages 350–356.

Duboue, P. A. and Chu-Carroll, J. (2006). Answering the question you wish they had asked: The impact of paraphrasing for question answering. In *Proceedings of HLT-NAACL*, pages 33–36.

Finch, A., Hwang, Y.-S., and Sumita, E. (2005). Using machine translation evaluation techniques to determine sentence-level semantic equivalence. In *Proceedings of IWP*, pages 17–24.

Fonseca, B. M., Golgher, P., Pôssas, B., Ribeiro-Neto, B., and Ziviani, N. (2005). Concept-based interactive query expansion. In *Proceedings of CIKM*, pages 696–703.

Jones, R., Rey, B., Madani, O., and Greiner, W. (2006). Generating query substitutions. In *Proceedings of WWW*, pages 387–396.

Koehn, P., Och, F. J., and Marcu, D. (2003). Statistical phrase-based translation. In *Proceedings of HLT-NAACL*, pages 127–133.

Lin, D.-K. and Pantel, P. (2001). Discovery of inference rules for question answering. *Natural Language Engineering*, 7(4):343–360.

Madnani, N., Ayan, N. F., Resnik, P., and Dorr, B. J. (2007). Using paraphrases for parameter tuning in statistical machine translation. In *Proceedings of the Second Workshop on Statistical Machine Translation*, pages 120–127.

Malakasiotis, P. (2009). Paraphrase recognition using machine learning to combine similarity measures. In *Proceedings of the ACL-IJCNLP 2009 Student Research Workshop*, pages 27–35.

Marton, Y., Callison-Burch, C., and Resnik, P. (2009). Improved statistical machine translation using monolingually-dervied paraphrases. In *Proceedings of EMNLP*, pages 381–390.

Ravichandran, D. and Hovy, E. (2002). Learning surface text patterns for a question answering system. In *Proceedings of ACL*, pages 41–47.

Riezler, S., Liu, Y., and Vasserman, A. (2008). Translating queries into snippets for improved query expansion. In *Proceedings of COLING*, pages 737–744.

Riezler, S., Vasserman, A., Tsochantaridis, I., Mittal, V., and Liu, Y. (2007). Statistical machine translation for query expansion in answer retrieval. In *Proceedings of ACL*, pages 464–471.

Szpektor, I., Gionis, A., and Maarek, Y. (2011). Improving recommendation for long-tail queries via templates. In *Proceedings of WWW*, pages 47–56.

Szpektor, I., Shnarch, E., and Dagan, I. (2007). Instance-based evaluation of entailment rule acquisition. In *Proceedings of ACL*, pages 456–463.

Wen, J.-R., Nie, J.-Y., and Zhang, H.-J. (2002). Query clustering using user logs. *ACM Trans. Information. Systems*, 20(1):59–81.

Zhang, Z. and Nasraoui, O. (2006). Mining search engine query logs for query recommendation. In *Proceedings of WWW*, pages 1039–1040.

Zhao, S., Wang, H., and Liu, T. (2010). Paraphrasing with search engine query logs. In *Proceedings of COLING*, pages 1317–1325.

Zhao, S., Wang, H., Liu, T., and Li, S. (2008). Pivot approach for extracting paraphrase patterns from bilingual corpora. In *Proceedings of ACL-08:HLT*, pages 780–788.

Zukerman, I. and Raskutti, B. (2002). Lexical query paraphrasing for document retrieval. In *Proceedings of COLING*, pages 1177–1183.

Exploiting Bilingual Translation for Question Retrieval in Community-Based Question Answering

Guangyou Zhou, Kang Liu and Jun Zhao
National Laboratory of Pattern Recognition
Institute of Automation, Chinese Academy of Sciences
95 Zhongguancun East Road, Beijing 100190, China
{gyzhou, kliu, jzhao}@nlpr.ia.ac.cn

ABSTRACT
Community-based question answering (CQA) has become an important issue due to the popularity of CQA archives on the web. This paper is concerned with the problem of question retrieval. Question retrieval in CQA archives aims to find historical questions that are semantically equivalent or relevant to the queried questions. However, question retrieval is challenging partly due to the word ambiguity and lexical gap between the queried questions and the historical questions in the archives. To deal with these problems, we propose the use of translated words to enrich the question representation, going beyond the words in the original language to represent a question. In this paper, each original language question (e.g., English) is automatically translated into an foreign language (e.g., Chinese) by machine translation services, and the resulting translated questions serves as a semantically enhanced representation for supplementing the original bag of words. Experiments conducted on real CQA data set demonstrate that our proposed approach significantly outperforms several baseline methods and achieves the state-of-the-art performance.

TITLE AND ABSTRACT IN ANOTHER LANGUAGE, L_2 (OPTIONAL, AND ON SAME PAGE)

利用双语翻译对社区问答进行问题检索

由于互联网上社区问答数据集的流行，使得社区问答的研究变得越来越流行。本文关注的是问题检索。 问题检索的目的是从历史问题数据集中查找与查询问题语义等价或相关的历史问题。然而，问题检索的挑战主要是词汇歧义和查询问题与历史问题 之间的词汇鸿沟。为了解决这些问题，我们提出利用翻译词来丰富问题的表示，而不单纯利用原始语言的词来表示问题。 在本文中，通过机器翻译，每个原始语言（例如：英语）的问题都被自动翻译成另一种外国语言（例如：汉语）， 经过翻译后的问题可以作为一种增强的语义表示来辅助原始的基于词袋的表示方法。在真实社区问答数据集上的实验表明，我们的方法可以极大提升基线系统的方法 并取得了最好的性能。

KEYWORDS: Community Question Answering, Question Retrieval, Bilingual Translation.

KEYWORDS IN L_2: 社区问答，问题检索，双语翻译

Proceedings of COLING 2012: Technical Papers, pages 3153–3170,
COLING 2012, Mumbai, December 2012.

1 引言

在过去的若干年中，大规模的问答数据集成了互联网上的重要信息资源。 这些资源包括传统的由专家或公司为他们的产品提供的常见问题解答集以及新出现的基于社区的在线服务，例如Yahoo! Answers和Live QnA，在这些在线社区上，人们可以回答他人提出的问题。这种在线社区称为基于社区的问答服务。 在这些社区中，任何人都可以提问和回答关于任何主题的问题，寻找信息的人与那些知道答案的人就联系起来了。 由于社区问答上的答案通常以显式的形式由人们提供，它们对回答真实问题起到了很好的作用 (Wang et al., 2009)。

为了更好地利用大规模的问答对，具备帮助用户检索先前答案的功能非常必要 (Duan et al., 2008)。因此， 检索与查询问题语义等价或相关的问题是一件非常有意义的任务。然而，问题检索的挑战主要是词汇歧义和查询问题与历史问题 之间的词汇鸿沟。词汇歧义通常会引发问题检索模型检索出许多与用户查询意图不匹配的历史问题。这也是由问题和用户的高度多样化造成的。例如，依据不同的用户，词"interest"既可以指"curiosity"也可以指"a charge for borrowing money"。另外一个挑战是查询问题与历史问题的词汇鸿沟。查询问题中的词不同于历史问题中的词但是它们之间是相关的词。 词汇鸿沟问题对社区问答的问题检索而言更加严重，主要是问答对通常很短，查找相同的内容表达往往使用不同的词(Xue et al., 2008)。

为了解决词汇鸿沟问题，大多数学者将问题检索任务看作是一个统计机器翻译的问题，并利用IBM模型1(Brown et al., 1993)来学习词与词之间的翻译概率(Berger et al., 2000; Jeon et al., 2005; Xue et al., 2008; Lee et al., 2008; Bernhard and Gurevych, 2009)。 实验结果一致表明基于词的翻译模型取得了比传统检索方法更好的性能。 最近，Riezler et al. (2007)和Zhou et al. (2011)提出了基于统计短语翻译的问题和答案检索方法。基于短语的翻译模型可以刻画上下文信息，在翻译的过程中对整个短语建模， 从而在某种程度上降低了词汇歧义的问题。然而，目前公开发表的工作都是基于单语的方法，仅仅利用了原始语言的信息， 而没有利用来自其它语言潜在的丰富的语义信息。通过其它语言，可以利用各种方法增加原始问题的语义信息，从而提高仅仅利用原始语言方法的性能。

通过利用外国语言，我们提出利用翻译表示通过外国语言词汇来替换原始语言中的词，其中外国语言是指不同于原始语言的。 利用双语信息进行问题检索的基本思想如下：（1）从一种语言翻译成另一种语言的过程中可以利用上下文信息，如表1所示，英文单词"interest"和 "bank"在不同的上下文中有多种意思，在利用**Google Translate** (Google-Trans)翻译的过程中正确的意思可以得到纠正。因此，问题中词的歧义在翻译的过程中可以根据上下文信息得到解决。（2）多个语言相关的词在某种语言中可以被翻成另外一种语言的唯一表示。如表1所示，英文单词例如"company"和"firm"可以被翻译成中文单词"公司 (gōngsī)"，"rheum"和"catarrh"可以被翻译成中文单词"感冒(gǎnmào)"。

在本文中，通过机器翻译，每个原始语言（例如：英语）的问题都被自动翻译成另一种外国语言（例如：汉语）， 经过翻译后的问题可以作为一种增强的语义表示来辅助原始的基于词袋的表示方法。具体来说， 原始语言与外国语言的词汇之间通过翻译联系起来，对解决上述两个问题的解决起到重要的作用。 首先，每个原始语言句子中的词可以被翻译成另一种语言中的多个词，因此在给定原始语言中词的上下文的情况下， 词汇歧义在翻译的过程中可以得到解决。同时，语义相关的多个词可以被翻译成另一外国语言中的一个词。 因此，原始语言中的词汇鸿沟在某种程度上可以通过另一种外国语言中的翻译词来解决。

我们利用来自Yahoo! Answers的大规模数据集做实验。采用两种商业翻译服务（例如，Google Translate和Yahoo Babel Fish和一种基于词典的基线翻译将大规模的英文问题翻译成中文问题。 实验表明，我们的方法可以极大提升基线系统的方法并取得了最好的

	英语	汉语
词汇歧义	How do I get a loan from a **bank**?	我(wǒ) 如何(rúhé) 从(cóng) **银行(yí nháng)** 贷款(dàikuǎn) ？
	How to reach the **bank** of the river?	如何(rúhé) 前往(qiánwǎng) **河岸(héàn)** ？
词汇鸿沟	company firm	公司(gōngsī) 公司(gōngsī)
	rheum catarrh	感冒(gǎnmào) 感冒(gǎnmào)

Table 1: 谷歌翻译（Google translate）：一些例子。

性能。

论文的组织结构如下。第三部分介绍了我们方法的框架。第四部分详细介绍了我们的方法。第五部分给出了实验结果。 在第六部分，我们总结了全文并对未来工作做了展望。

2 Introduction

Over the past few years, large-scale question and answer archives have become an important information resource on the Web. These include the traditional FAQ archives constructed by the experts or companies for their products and the emerging community-based online services, such as Yahoo! Answers[1] and Live QnA[2], where people answer questions posed by other people. This is referred as the community-based question answering services. In these communities, anyone can ask and answer questions on any topic, and people seeking information are connected to those who know the answers. As answers are usually explicitly provided by human, they can be helpful in answering real world questions (Wang et al., 2009).

To make use of the large-scale archives of question-answer pairs, it is critical to have functionality of helping users to retrieve previous answers (Duan et al., 2008). Therefore, it is a meaningful task to retrieve the semantically equivalent or relevant questions to the queried questions. However, question retrieval is challenging partly due to the **word ambiguity** and **lexical gap** between the queried questions and the historical questions in the archives. **Word ambiguity** often causes a question retrieval model to retrieve many historical questions that do not match the user's intent. This problem is also amplified by the high diversity of questions and users. For example, depending on different users, the word "interest" may refer to "curiosity", or "a charge for borrowing money". Another challenge is **lexical gap** between the queried questions and the historical questions. The queried questions may contain words that are different from, but related to, the words in the relevant historical questions. The lexical gap is substantially bigger for question retrieval in CQA largely due to the fact that the question-answer pairs are usually short and there is little chance of finding the same content expressed using different wording (Xue et al., 2008).

To solve the lexical gap problem, most researchers regarded the question retrieval task as a statistical machine translation problem by using IBM model 1 (Brown et al., 1993) to learn the word-to-word translation probabilities (Berger et al., 2000; Jeon et al., 2005; Xue et al., 2008; Lee et al., 2008; Bernhard and Gurevych, 2009). Experiments consistently reported that the word-based translation models could yield better performance than the traditional methods. Recently, Riezler et al. (2007) and Zhou et al. (2011) proposed a phrase-based translation model for question and answer retrieval. The phrase-based translation model can capture some contextual information in modeling the translation of phrases as a whole, thus the word ambiguity problem is somewhat alleviated. However, most existing studies in the literature are basically *monolingual approaches* which are restricted to the use of original language of questions, without taking advantage of potentially rich semantic information drawn from other languages. Through other languages, various ways of adding semantic information to a question could be available, thereby leading to potentially more improvements than using the original language only.

Taking a step toward using other languages, we propose the use of *translated representation* by alternatively presenting the original questions with the words of a foreign language, one that is different from the original language of questions. The idea of exploiting bilingual information for question retrieval is based on the following observations: (1) Contextual information is exploited during the translation from one language to another. As shown in Table 2, English words "interest" and "bank" that have multiple meanings under different contexts are correctly

[1] http://answers.yahoo.com
[2] http://qna.live.com

	English	Chinese
Word ambiguity	How do I get a loan from a **bank**?	我(wǒ) 如何(rúhé) 从(cóng) **银行(yínháng)** 贷款(dàikuǎn) ?
	How to reach the **bank** of the river?	如何(rúhé) 前往(qiánwǎng) **河岸(héàn)** ?
Lexical gap	company firm	公司(gōngsī) 公司(gōngsī)
	rheum catarrh	感冒(gǎnmào) 感冒(gǎnmào)

Table 2: Google translate: some illustrative examples.

addressed by **Google Translate**[3] (GoogleTrans). Thus, word ambiguity based on contextual information is naturally involved when questions are translated. (2) Multiple words that are semantically similar in one language may be translated into unique words or a few words in a foreign language. For example in Table 2, English words such as "company" and "firm" are translated into "公司 (gōngsī)", "rheum" and "catarrh" are translated into "感冒(gǎnmào)" in Chinese.

In this paper, each original question is automatically translated into a foreign language by machine translation services, and the resulting translated questions serve as a semantically enhanced representation for supplementing the original bag of words. Specially, the vocabularies of the original and foreign languages are connected via translation, which could bring about important benefits in dealing with the two addressed problems. First, an original language word can be translated into multiple candidate words in a foreign language. Therefore, the word ambiguity problem can be resolved during the translation in a given context of an original language word. Conversely, various different original language words that refer to similar meanings are translated into a single word or a few words in a foreign language. Thus, the lexical gap problem in the original language is to some extent ameliorated by using the translated words in a foreign language.

We conduct experiments on a large-scale data set from Yahoo! Answers. Two commercial machine translation services (e.g., Google Translate and Yahoo Babel Fish[4]) and a baseline dictionary-based system are used for translating English questions into Chinese questions. Experimental results show that our proposed method significantly outperforms several baseline methods and achieves state-of-the-art performance.

The remainder of this paper is organized as follows. Section 3 introduces the framework of the proposed method. Section 4 describes our proposed method in detail. Section 5 presents the experimental results. In Section 6, we conclude with ideas for future research.

3 Framework of the Proposed Approach

The framework of the proposed approach for question retrieval is summarized in Figure 1. Each historical question in original language (e.g., English) is translated into the corresponding foreign language (e.g., Chinese) via machine translation services. Note that in the framework, different machine translation services can be used to obtain different translation. When a queried question is given, the queried question is translated using the same machine translator. Next, question retrieval on both representations (e.g., English representation and Chinese

[3]http://translate.google.com/translate_t
[4]http://babelfish.yahoo.com/translte_txt

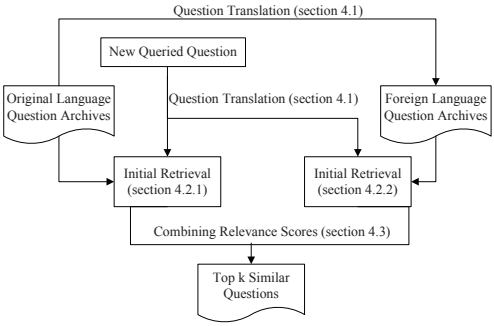


Figure 1: Framework of our approach by using question translated representation.

representation) is performed, and the two resulting relevance scores are combined to produce a final ranked list of semantically similar questions.

4 Our Approach

4.1 Question Translation

Translating historical questions in original language (e.g., English) into the corresponding foreign language is the first step of the proposed approach. Manual translation is time-consuming and labor-intensive, and it is not feasible to manually translate a large amount of questions in original language in real applications. Fortunately, machine translation techniques have been well developed in the NLP filed, though the translation quality is far from satisfactory. A few commercial machine translation services can be publicly accessed. In this paper, the following two commercial machine translation services and one baseline system are used to translate English questions into Chinese questions.

Google Translate (GoogleTrans): Google Translate is one of the state-of-the-art commercial machine translation systems used today. Google Translate employs statistical machine learning methods to build a translation model based on large-scale bilingual parallel corpus. Contextual information is utilized during the translation from one language text to the aligned text in anther language.

Yahoo Babel Fish (YahooTrans): Different from Google Translate, Yahoo Babel Fish uses SYSTRAN's rule-based translation engine. SYSTRAN is one of the earliest developers of machine translation software. SYSTRAN employs complex sets of specific rules defined by linguists to analyze and then transfer the grammatical structure of the source language into the target language. During the translation, word ambiguity can be resolved based on the contextual information.

Baseline Translate (DicTran): We simply develop a translation method based only on one-to-one word translation using an English to Chinese lexicon in StarDict[5].

[5]StarDict is an open source dictionary software, available at http://stardict.sourceforge.net/.

4.2 Bilingual Retrieval Method

4.2.1 Retrieval model

Language models have been performed quite well empirically in many information retrieval tasks (Zhai and Lafferty, 2001), and also have performed very well in question retrieval (Jeon et al., 2005; Cao et al., 2009, 2010). In this paper, we use the language modeling approach for question retrieval. In the language modeling approach to question retrieval, language models are constructed for each queried question $\mathbf{q}$ and each historical question $\mathbf{d}$ in CQA archives C. The historical questions in C are ranked by the distance to a given queried question $\mathbf{d}$ according to the language models. The most commonly used language model in question retrieval is the unigram model, in which words are assumed to be independent of each other.

One of the commonly used measures of the similarity between query model and historical question model is negative Kullback-Leibler (KL) divergence (Zhai and Lafferty, 2001). With unigram model, the negative KL-divergence between model $\theta_{\mathbf{q}}$ of query $\mathbf{q}$ and model $\theta_{\mathbf{d}}$ of historical question $\mathbf{d}$ is computed as follows:

$$\begin{aligned} Score(\mathbf{q},\mathbf{d}) &= -\sum_{w\in V} p(w|\theta_{\mathbf{q}}) \log \frac{p(w|\theta_{\mathbf{q}})}{p(w|\theta_{\mathbf{d}})} \\ &= \sum_{w\in V} p(w|\theta_{\mathbf{q}}) \log p(w|\theta_{\mathbf{d}}) - \sum_{w\in V} p(w|\theta_{\mathbf{q}}) \log p(w|\theta_{\mathbf{q}}) \\ &= \sum_{w\in V} p(w|\theta_{\mathbf{q}}) \log p(w|\theta_{\mathbf{d}}) + E(\theta_{\mathbf{q}}) \end{aligned} \tag{1}$$

where $p(w|\theta_{\mathbf{q}})$ and $p(w|\theta_{\mathbf{d}})$ are the generative probabilities of a word w from the models $\theta_{\mathbf{q}}$ and $\theta_{\mathbf{d}}$, V is the vocabulary of C, and $E(\theta_{\mathbf{q}})$ is the entropy of $\mathbf{q}$.

Let $tf(w,\mathbf{q})$ and $tf(w,\mathbf{d})$ as the frequencies of w in $\mathbf{q}$ and $\mathbf{d}$, respectively. Generally, $p(w|\theta_{\mathbf{q}})$ is calculated with maximum likelihood estimation (MLE):

$$p(w|\theta_{\mathbf{q}}) = \frac{tf(w,\mathbf{q})}{\sum_{w'\in\mathbf{q}} tf(w',\mathbf{q})} \tag{2}$$

To calculate $p(w|\theta_{\mathbf{d}})$, several smoothing methods have been proposed to overcome the data sparseness problem of a language model constructed from one historical question (Zhai and Lafferty, 2001). Therefore, $p(w|\theta_{\mathbf{d}})$ with the Dirichlet prior smoothing can be calculated as follows:

$$p(w|\theta_{\mathbf{d}}) = \frac{tf(w,\mathbf{d}) + \lambda p(w|\theta_C)}{\sum_{w'\in V} tf(w',\mathbf{d}) + \lambda} \tag{3}$$

where λ is the prior parameter in the Dirichlet prior smoothing method, and $p(w|\theta_{\mathbf{C}})$ is the probability of w in C, which is often computed with MLE:

$$p(w|\theta_C) = \frac{\sum_{\mathbf{d}\in C} tf(w,\mathbf{d})}{\sum_{\mathbf{d}\in C}\sum_{w'\in V} tf(w',\mathbf{d})} \tag{4}$$

4.2.2 Retrieval model for translated representation

We now extend the retrieval model described in subsection 4.2.1 in order to support translated representation. Let $\pi(\mathbf{d})$ be the translated representation result by using the machine translation service π (e.g., Google Translate) for a given historical question $\mathbf{d}$, and $\pi(\mathbf{q})$ be the translated representation result by using the machine translation service π for a queried question $\mathbf{q}$. Therefore, the query language model $p(w|\theta_{\pi(\mathbf{q})})$ based on the translated representation can be calculated as follows:

$$p(w|\theta_{\pi(\mathbf{q})}) = \frac{tf(w, \pi(\mathbf{q}))}{\sum_{w' \in \pi(\mathbf{q})} tf(w', \pi(\mathbf{q}))} \tag{5}$$

Similarly, by replacing $tf(w, \mathbf{d})$ in equation (3) with $tf(w, \pi(\mathbf{d}))$, we obtain the following smoothed model $p(w|\theta_{\pi(\mathbf{d})})$:

$$p(w|\theta_{\pi(\mathbf{d})}) = \frac{tf(w, \pi(\mathbf{d})) + \lambda p(w|\theta_{\pi(C)})}{\sum_{w' \in V_F} tf(w', \pi(\mathbf{d})) + \lambda} \tag{6}$$

where V_F is vocabulary of the translated foreign language, and $p(w|\theta_{\pi(\mathbf{C})})$ is defined by

$$p(w|\theta_{\pi(C)}) = \frac{\sum_{\mathbf{d} \in C} tf(w, \pi(\mathbf{d}))}{\sum_{\mathbf{d} \in C} \sum_{w' \in V_F} tf(w', \pi(\mathbf{d}))} \tag{7}$$

Finally, we calculate the relevance score of the historical question $\mathbf{d}$ with respect to the queried question $\mathbf{q}$ using $Score(\pi(\mathbf{q}), \pi(\mathbf{d}))$ based on their translated representation.

4.3 Combining Relevance Score for Bilingual Representation

After obtaining the two relevance scores from the original and translated representation perspective, we can rank the final similar historical questions based on the linear combination and refined ranking approach, respectively.

4.3.1 Linear combination

To produce a single ranked list from the two relevance scores using equation (1) on the original and translated representation, we use the following linear combination:

$$Score_{E+F}(\mathbf{q}, \mathbf{d}) = \alpha Score(\mathbf{q}, \mathbf{d}) + (1 - \alpha) Score(\pi(\mathbf{q}), \pi(\mathbf{d})) \tag{8}$$

In equation (8), the importance of relevance scores on the original and translated representation is adjusted through α. When $\alpha = 1$, the final retrieval model is based on the original representation. When $\alpha = 0$, the final retrieval model is based on the translated representation.

4.3.2 Refined ranking approach

Based on the original and translated representation, we can obtain two kinds of ranked lists $\vec{R}_E(\mathbf{q})$ and $\vec{R}_F(\pi(\mathbf{q}))$, which reflect the similarity between a queried question and a historical

questions from two different perspectives. If the retrieval model based on the original representation cannot capture the similarity due to the word ambiguity and lexical gap, then the retrieval model based on the translated representation should be good for dealing with the word ambiguity and lexical gap problems. Therefore, we consider a refined ranking approach to boost the question retrieval performance.

In order to measure the similarity between the two ranked results, we utilize a measurement, similar to Jaccard coefficient, which is defined as the size of the intersection divided by the size of the union of these two top k ranked results,

$$J = \frac{|\vec{R}_E(\mathbf{q}) \bigcap \vec{R}_F(\pi(\mathbf{q}))|}{|\vec{R}_E(\mathbf{q}) \bigcup \vec{R}_F(\pi(\mathbf{q}))|} \tag{9}$$

This measurement implies the following meaning: a large value is reached if the retrieval model based on the translated representation could retrieve many common relevant historical questions within the top-k results. Based on this scheme, we adopt a measurement for an adaptive ranking refinement. Let $R_E(\mathbf{q}, \mathbf{d})$ be the rank of historical question $\mathbf{d}$ for a given queried question $\mathbf{q}$, and let $R_F(\pi(\mathbf{q}), \pi(\mathbf{d}))$ be the rank of translated representation $\pi(\mathbf{d})$ for a given translated queried question $\pi(\mathbf{q})$. Therefore, we define a refined score $Score(\mathbf{q}, \mathbf{d})$ based on the following function:

$$Score(\mathbf{q}, \mathbf{d}) = \frac{1}{R_E(\mathbf{q}, \mathbf{d})} + \varphi(\mathbf{d}) \cdot J \cdot \frac{1}{\hat{R}_F(\pi(\mathbf{q}), \pi(\mathbf{d}))} \tag{10}$$

where $\varphi(\mathbf{d}) = 1$ if $\mathbf{d} \in \vec{R}_E(\mathbf{q})$ and $\pi(\mathbf{d}) \in \vec{R}_F(\pi(\mathbf{q}))$, otherwise $\varphi(\mathbf{d}) = 0$. By applying the refined ranking strategy, we obtain the refined ranking model shown in Algorithm 1.

Algorithm 1 Refined Model for Question Retrieval

Input: Given a queried question $\mathbf{q}$;

Step1: Retrieve the top-k most relevant historical questions based on the original representation using equations (1) to (4), and then obtain the ranked results $\vec{R}_E(\mathbf{q})$;

Step2: Retrieve the top-k most relevant historical questions based on the translated representation using equations (5) to (7), and then obtain the ranked results $\vec{R}_F(\pi(\mathbf{q}))$;

Step3: Refine with equation (10) and get the final ranked results.

Output: Return the ranked historical questions $\{\mathbf{d}_1, \mathbf{d}_2, \cdots, \mathbf{d}_k\}$.

4.4 Category-Sensitive Language Model for Bilingual Representation

In CQA, when a user asks a question, the user typically needs to choose a category for the question from a predefined hierarchy of categories. Hence, each question in CQA archive has a category label and questions in CQA services are organized into hierarchies of categories (Cao et al., 2009, 2010). Based on these observations, it is naturally to employ the category information for bilingual representation. Let $c(\mathbf{d})$ be the leaf category of historical question $\mathbf{d}$, then *category-term frequency* of $\mathbf{d}$ for word w is defined as follows:

$$tf(w, \mathbf{d} \cup c(\mathbf{d})) = tf(w, \mathbf{d}) + \mu \cdot tf(w, c(\mathbf{d})) \tag{11}$$

where μ is the weight of category frequency, and $tf(w, c(\mathbf{d}))$ is frequency of word w in $c(\mathbf{d})$. Finally, model $\theta_{\mathbf{d}}$ defined in equation (3) is written as:

$$\begin{aligned} p(w|\theta_{\mathbf{d},c(\mathbf{d})}) &= \frac{tf(w, \mathbf{d} \cup c(\mathbf{d})) + \lambda p(w|\theta_C)}{\sum_{w' \in V} tf(w', \mathbf{d} \cup c(\mathbf{d})) + \lambda} \\ &= \frac{tf(w, \mathbf{d}) + \mu \cdot tf(w, c(\mathbf{d})) + \lambda p(w|\theta_C)}{\sum_{w' \in V} tf(w', \mathbf{d}) + \sum_{w' \in V} tf(w', c(\mathbf{d})) + \lambda} \end{aligned} \tag{12}$$

Similarly, we could define the translated representation for model $p(w|\theta_{\pi(\mathbf{d})})$ as follows:

$$p(w|\theta_{\pi(\mathbf{d}),\pi(c(\mathbf{d}))}) = \frac{tf(w, \pi(\mathbf{d})) + \mu \cdot tf(w, \pi(c(\mathbf{d}))) + \lambda p(w|\theta_{\pi(C)})}{\sum_{w' \in V} tf(w', \pi(\mathbf{d})) + \sum_{w' \in V} tf(w', \pi(c(\mathbf{d}))) + \lambda} \tag{13}$$

Given the bilingual representation, we again combine the two category-sensitive relevance scores with the above linear combination and refined ranking approach, respectively.

5 Experiments

5.1 Data Set and Evaluation Metrics

We collect the data set from Yahoo! Answers and use the *getByCategory* function provided in Yahoo! Answers API[6] to obtain CQA threads from the Yahoo! site. More specifically, we utilize the *resolved* questions and the resulting question repository that we use for question retrieval contains 2,288,607 questions. Each resolved question consists of four parts: "question title", "question description", "question answers" and "question category". For question retrieval, we only use the "question title" part. It is assumed that the titles of the questions already provide enough semantic information for understanding the users' information needs (Duan et al., 2008). There are 26 categories at the first level and 1,262 categories at the leaf level. Each question belongs to a unique leaf category. Table 3 shows the distribution across first-level categories of the questions in the archives.

Category	#Size	Category	# Size
Arts & Humanities	86,744	Home & Garden	35,029
Business & Finance	105,453	Beauty & Style	37,350
Cars & Transportation	145,515	Pet	54,158
Education & Reference	80,782	Travel	305,283
Entertainment & Music	152,769	Health	132,716
Family & Relationships	34,743	Sports	214,317
Politics & Government	59,787	Social Science	46,415
Pregnancy & Parenting	43,103	Ding out	46,933
Science & Mathematics	89,856	Food & Drink	45,055
Computers & Internet	90,546	News & Events	20,300
Games & Recreation	53,458	Environment	21,276
Consumer Electronics	90,553	Local Businesses	51,551
Society & Culture	94,470	Yahoo! Products	150,445

Table 3: Number of questions in each first-level category

We use the same test set in previous work (Cao et al., 2009, 2010). This set contains 252 queried questions and can be freely downloaded for research communities.[7] For each method, the top 20 retrieval results are kept. Given a returned result for each queried question, an

[6] http://developer.yahoo.com/answers

[7] The data set is available at http://homepages.inf.ed.ac.uk/gcong/qa/

annotator is asked to label it with "relevant" or "irrelevant". If a returned result is considered semantically equivalent to the queried question, the annotator will label it as "relevant"; otherwise, the annotator will label it as "irrelevant". Two annotators are involved in the annotation process. If a conflict happens, a third person will make judgement for the final result. In the process of manually judging questions, the annotators are presented only the questions.

Evaluation Metrics: We evaluate the performance of question retrieval using the following metrics: **Mean Average Precision** (MAP) and **Precision@N** (P@N). MAP rewards methods that return relevant questions early and also rewards correct ranking of the results. P@N reports the fraction of the top-N questions retrieved that are relevant. We perform a significant test, i.e., a t-test with a default significant level of 0.05.

Parameter Selection: We tune the parameters on a small development set of 50 questions. This development set is also extracted from Yahoo! Answers, and it is not included in the test set. For the smoothing parameter λ, we set $\lambda = 2000$ empirically in the language modeling approach for both English representation and Chinese translated representation. For parameter μ used in equation (11), we set $\mu = 0.8$ empirically. For parameter α, we do an experiment on the development set to determine the optimal values among 0.1, 0.2, $\cdots$, 0.9 in terms of MAP. As a result, we set $\alpha = 0.6$ in the experiments empirically as this setting yields the best performance. For parameter k described in Algorithm 1, we try several different values on the development set. Finally, we set $k = 30$ empirically as this setting gives the better performance.

5.2 Question Retrieval Results using Language Model

Table 4 shows a comparison of the results obtained using monolingual and bilingual representation using language model (LM) defined in subsection 4.2.1 and subsection 4.2.2 for question retrieval. In Table 4, **E** denotes the baseline LM using English representation (queried questions and historical questions). **C** denotes the run of LM using Chinese representation via English-Chinese translation (queried questions and historical questions). **E + C** denotes the run of LM with the combination of English and Chinese representation, where **Linear E + C** denotes the linear combination using equation 8, and **Refined E + C** denotes the refined ranking approach using equations 9 and 10. There are some clear trends in the results of Table 4:

Translation tools	#	Methods	MAP	P@10
-	1	E	0.385	0.242
GoogleTrans	2	C	0.350	0.234
	3	Linear E + C	0.468†	0.269†
	4	Refined E + C	**0.483**†	**0.275**†
YahooTrans	5	C	0.327	0.214
	6	Linear E + C	0.441†	0.258†
	7	Refined E + C	0.465†	0.267†
DicTran	8	C	0.246	0.178
	9	Linear E + C	0.398	0.246
	10	Refined E + C	0.414†	0.249

Table 4: Comparison of bilingual and monolingual representation using language model (LM) for question retrieval. The mark † indicates statistical significance over E.

(1) Using the bilingual translated representation, question retrieval performance can be significantly improved (row 1 vs. row 3 and row 4; row 1 vs. row 6 and row 7; row 1 vs. row 9 and

row 10). The reason is that various different words in English that refer to similar meanings can be translated into only a few words or a single word in Chinese. Thus, the lexical gap problem in English is to some extent ameliorated by using translated words in Chinese.

(2) We can see that question retrieval performance relies positively on the translated bilingual representation, and **GoogleTrans** performs the best while **DicTran** performs the worst (row 3 vs. row 9; row 4 vs. row 10), which is consistent with the fact **GoogleTrans** is deemed the best of the three machine translation systems, while **DicTran** is the weakest one. Moreover, **DicTran** performs translation without taking into account the surrounding words as contextual information, while **GoogleTrans** and YahooTrans are context-dependent and thus produce different translated Chinese words depending on the context of an English word. Therefore, the word ambiguity problem can be resolved during the English-Chinese translation in a given context of an English word.

(3) Comparing the two combination strategies **Linear** and **Refined**, it is seen that the refined ranking strategy (Refined E + C) gives the better results than linear combination regarding the different translation tools (row 3 vs. row 4; row 6 vs. row 7; row 9 vs. row 10).

5.3 Question Retrieval Results using Category-Sensitive Language Model

Table 5 shows the comparison results of monolingual and bilingual representation using category-sensitive language model (CSLM) for question retrieval. In Table 5, **E** denotes the baseline CSLM using the English representation only, and **E + C** denotes the run of CSLM based on the English and Chinese representation.

Translation tools	#	Methods	MAP	P@10
-	1	E	0.441	0.258
GoogleTrans	2	C	0.396	0.247
	3	Linear E + C	0.493†	0.282†
	4	Refined E + C	**0.525**†	**0.290**†
YahooTrans	5	C	0.358	0.237
	6	Linear E + C	0.476†	0.272†
	7	Refined E + C	0.492†	0.281†
DicTran	8	C	0.283	0.191
	9	Linear E + C	0.455	0.263
	10	Refined E + C	0.470†	0.270†

Table 5: Comparison of bilingual and monolingual representation using category-sensitive language model (CSLM) for question retrieval. The mark † indicates statistical significance over E.

Category-sensitive language model (CSLM) without considering the translated representation (e.g., row 1 in Table 5) is highly effective for question retrieval, achieving about 5.6% MAP increase over the baseline LM (e.g., row 1 in Table 4), with statistical significance. Similar findings have also been found by Cao et al. (2009) and Cao et al. (2010). Additionally, using the bilingual translated representation (E + C) achieves further improvements over CSLM (e.g., row 1 vs. row 3 and row 4). Specially, our refined ranking approach (Refined E + C) using GoogleTrans achieves about 8.4% further increase of MAP over the baseline CSLM (E) for question retrieval, finally leading to a noticeable increase of 14% MAP over the baseline LM.

5.4 Comparison with Different Methods

The motivation of this paper is to solve the lexical gap and word ambiguity problems for question retrieval. Jeon et al. (2005) proposed a word-based translation model for automatically fixing the lexical gap problem. Experimental results demonstrated that the word-based translation model significantly outperformed the traditional methods (i.e., VSM, BM25, LM). Xue et al. (2008) proposed a word-based translation language model for question retrieval. The results indicated that word-based translation language model further improved the retrieval results and obtained the state-of-the-art performance. Zhou et al. (2011) proposed a monolingual phrase-based translation model for question retrieval. This method can capture some contextual information in modeling the translation of phrases as a whole. To implement the word-based translation models, we use the GIZA++ alignment toolkit[8] trained on one million question-answer pairs from another data set[9] to learn the word-to-word translation probabilities. For phrase-based translation model described in (Zhou et al., 2011), we employ Moses toolkit[10] to extract the phrase translation and set the maximum length of phrases to 5. Recently, Singh (2012) extended the word-based translation model and explored strategies to learn the translation probabilities between words and the concepts using the CQA archives and a popular entity catalog. However, these existing studies in the literature are basically monolingual translation, which are restricted to the use of the original language of the CQA archives, without taking advantage of potentially rich semantic information drawn from other languages. In this paper, we propose the use of translated words to enrich question representation, going beyond the words in original language to represent the questions.

#	Methods	MAP	P@10
1	Jeon et al. (2005)	0.405	0.247
2	Xue et al. (2008)	0.436	0.261
3	Zhou et al. (2011)	0.452	0.268
4	Singh (2012)	0.450	0.267
5	**Refined E + C (LM, GoogleTrans)**	**0.483**†	**0.275**†

Table 6: Comparison with different methods for question retrieval without considering the category information. The mark † indicates statistical significance over previous work.

The comparisons with different methods for question retrieval are shown in Table 6. The results in Table 6 show that we propose the use of translated words to enrich question representation is much better than traditional monolingual approaches (row 1, row 2, row 3 and row 4 vs. row 5). Significant tests using t-test show the difference between our proposed approach and traditional monolingual approaches for cases marked in the table are statistically significant.

To further analyze why traditional monolingual approaches fail to give the satisfactory results for solving the word ambiguity and lexical gap problems, we identify two key challenges in adapting traditional monolingual translation approaches for question retrieval(Jeon et al., 2005; Xue et al., 2008; Zhou et al., 2011). First, unlike bilingual text, question-answer pairs are not semantically equivalent, leading to a wider range of possible phrases for a given phrase.

[8] http://www-i6.informatik.rwth-aachen.de/Colleagues/och/software/GIZA++.html

[9] The Yahoo! Webscope dataset Yahoo answers comprehensive questions and answers version 1.0, available at http://reseach.yahoo.com/Academic_Relations.

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

Furthermore, both sides of question-answer parallel text are written in the same language (e.g., English). Thus, the most strongly associated word or phrase pairs found by the off-the-shelf word alignment and phrase extraction tools are identical pairs. Second, in question-answer pairs, there are far more unaligned words than in bilingual pairs. Also, there are more large phrase pairs that cannot be easily decomposed. These difficult cases confuse the IBM word alignment models. To the best of our knowledge, it is the first work to give a thorough analysis the key challenges in adapting traditional monolingual translation approaches for question retrieval.

Besides, we are aware of only two published studies (Cao et al., 2009) and (Cao et al., 2010) on utilizing category information for question retrieval. Now we compare our proposed category-sensitive language model (CSLM) for bilingual representation with these two studies. Cao et al. (2009) employed classifiers to compute the probability of a queried question belonging to different categories, and then incorporated the classified categories into language model for question retrieval. Cao et al. (2010) introduced the different combinations to compute the global relevance and local relevance, the combination VSM + TRLM showed the superior performance than others. In this paper, we compare the proposed method with the combination VSM + TRLM. To implement these two methods, we employ the same parameter settings with Cao et al. (2009) and Cao et al. (2010). Table 7 shows the comparison. From this table, we can see that our proposed category-sensitive language model (CSLM) for bilingual representation can significantly improve the performance. The results also validate the effectiveness of the proposed method.

#	Methods	MAP	P@10
1	Cao et al. (2009)	0.408	0.247
2	Cao et al. (2010)	0.456	0.269
3	**Refined E + C (CSLM, GoogleTrans)**	**0.525**†	**0.290**†

Table 7: Comparison with previous work for question retrieval by considering the category information. The mark † indicates statistical significance over previous work.

5.5 Parameter Sensitivity of Combination

To combine the relevance scores for question retrieval, we propose to use the linear combination and the refined ranking approach to rank the final similar questions. In linear combination, we use parameter α to control the relative importance of original question representation and translated representation. In refined ranking approach, we retrieve the top-k similar questions from two perspectives for each queried question. To investigate the effect of these two parameters, we design the following experiments.

To examine the effect of α, we choose the best translation service GoogleTrans and evaluate α with different values among 0, 0.1, $\cdots$, 0.9 in terms of MAP on a small development set of 50 questions. This development set is also extracted from Yahoo! Answers data, and it is not included in the test set. The experimental results for different α are illustrated in Figure 2. Monolingual baselines E and C are used for reference. Figure 2(Left) shows that, for MAP, **E + C** performs better than baselines E and C when $\alpha \in (0.2, 0.9)$. Therefore, a relative broad set of good parameter value is observed. When $\alpha = 0.6$, **E + C** gives the best performance.

To investigate the effect of parameter k, we also choose the best translation service GoogleTrans with several different values from 10 to 50 in terms of MAP on this development set.

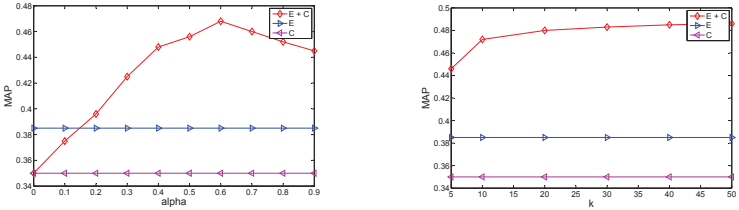


Figure 2: Left: The effect of parameter α for the linear combination using MAP metric; Right: The effect of parameter k for the refined ranking using MAP metric.

The experimental results for different k are illustrated in Figure 2(Right). We can see the performance becomes better for greater k used in the refined ranking approach. We believe the reason is that more historical questions may contain more similar questions. However, a larger k may result in longer processing time. Therefore, a good tradeoff is to set $k = 30$.

6 Related Work

6.1 Question Retrieval in CQA

The research of question retrieval has been further extended to the CQA data. The major challenge for question retrieval in CQA is the word ambiguity and lexical gap problems. Jeon et al. (2005) proposed a word-based translation model for automatically fixing the lexical gap problem. Xue et al. (2008) proposed a word-based translation language model for question retrieval. The results indicated that word-based translation language model further improved the retrieval results and obtained the state-of-the-art performance. Subsequent work on word-based translation models focused on providing suitable parallel data to learn the translation probabilities. Lee et al. (2008) tried to further improve the translation probabilities based on question-answer pairs by selecting the most important terms to build compact translation models. Bernhard and Gurevych (2009) proposed to use as a parallel training data set the definitions and glosses provided for the same term by different lexical semantic resources. Cao et al. (2010) explored the category information into the word-based translation model for question retrieval.

Recently, Riezler et al. (2007) and Zhou et al. (2011) proposed a phrase-based translation model for question and answer retrieval. The phrase-based translation model can capture some contextual information in modeling the translation of phrases as a whole, thus the word ambiguity and lexical gap problems are somewhat alleviated. Singh (2012) addressed the lexical gap issues by extending the lexical word-based translation model to incorporate semantic information (entities).

However, most existing works in the literature are basically monolingual approaches which are restricted to the use of the original language of the CQA archives, without taking advantage of potentially rich semantic information drawn from other languages. In this paper, we intend to address two fundamental issues in question retrieval: word ambiguity and lexical gap. To solve these problems, we enrich the question representation via bilingual translation. Compared to the traditional monolingual approaches, our proposed bilingual translation is much more effective due to the recent advance in statistical machine translation. To the best of our knowledge, it is the first work to improve question retrieval in CQA via bilingual translation.

6.2 WSD and Query Expansion for Monolingual Information Retrieval

Besides in CQA, word ambiguity and lexical gap have been investigated in information retrieval (IR). Zhong and Ng (2012) proposed a novel approach to incorporate word senses into the language modeling approach to IR. Experimental results showed that word sense disambiguation (WSD) can significantly improve a state-of-the-art IR system. Query expansion has been one of the most effective approaches to resolve the lexical gap problem, which enrich the original query by adding some additional words (Lv and Zhai, 2010; Xu et al., 2009). Recently, Trieschnigg et al. (2010) enriched the original word-based representation with a concept-based representation, thereby proposing the translation of the original word language to a concept language. However, their translation models are based solely on the use of translation at the lexical level (e.g., word-to-concept), and thus their method is very different from our context-dependent style of translation. Na and Ng (2011) also applied automatic translation for monolingual retrieval. However, they used the expected frequency of a word computed from all possible translated representations, while we use the state-of-the-art commercial machine translation service (e.g., Google Tranlate), which is much simpler than their translation strategies.

6.3 Machine Translation for Cross-Lingual Information Retrieval

Cross-lingual retrieval information retrieval (CLIR) addresses the problem of retrieving documents written in a language different from the query language. The common approach in CLIR is to perform query translation or document translation using a machine translation system(Chen and Gey, 2004; Kraaij et al., 2003). However, the major difference is that our goal is to improve monolingual question retrieval and not CLIR. Moreover, these studies performed translation without taking into account the context information of an original word(Chen and Gey, 2004; Kraaij et al., 2003). On the contrary, our approach is context-dependent and thus produces different translated words depending on the context of a word in original language.

Conclusion and Future Work

In this paper, we intend to address two fundamental issues in question retrieval: word ambiguity and lexical gap. To solve these problems, we propose the use of bilingual question representation, encouraged by the fact that a translated word in a foreign language can be used to enrich the original question representation. We employ the statistical machine translation services to automatically translate all questions, producing bilingual representations. Then, the relevance score between a queried question and a historical question is computed by combining two evidences derived from the bilingual perspectives. Experimental results conducted on large-scale CQA data set from Yahoo! Answers show that by using English-Chinese translation, our approach achieves improvements over monolingual approaches, and the improvements are in many cases statistically significant.

There are some ways in which this research could be continued. First, we would like to extend the current experiments by considering other languages (e.g., English-French, Chinese-English, etc.). We want to see how strongly the linguistic diversity between original and foreign languages affects question retrieval performance. Second, we will try to investigate the use of the proposed approach for other kinds of data set, such as categorized questions from forum sites and FAQ sites.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (No. 61070106, No. 61272332 and No. 61202329), the National High Technology Development 863 Program of China (No. 2012AA011102), the National Basic Research Program of China (No. 2012CB316300), Tsinghua National Laboratory for Information Science and Technology (TNList). We thank the anonymous reviewers for their insightful comments. We also thank Dr. Gao Cong for providing the data set.

References

Berger, A., Caruana, R., Cohn, D., Freitag, D., and Mittal, V. (2000). Bridging the lexical chasm: statistical approach to answer-finding. In *ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2000)*, pages 192–199.

Bernhard, D. and Gurevych, I. (2009). Combining lexical semantic resources with question & answer archives for translation-based answer finding. In *Annual Meeting of the Association for Computational Linguistics (ACL 2009)*, pages 728–736.

Brown, P., Pietra, V., Pietra, S., and Mercer, R. (1993). The mathematics of statistical machine translation: parameter estimation. *Computational Linguistics*, 19(2):263–311.

Cao, X., Cong, G., Cui, B., and Jensen, C. (2010). A generalized framework of exploring category information for question retrieval in community question answer archives. In *International Conference on World Wide Web (WWW 2010)*, pages 201–210.

Cao, X., Cong, G., Cui, B., Jensen, C., and Zhang, C. (2009). The use of categorization information in language models for question retrieval. In *ACM Conference on Information and Managemenet (CIKM 2009)*, pages 265–274.

Chen, A. and Gey, F. (2004). Multilingual information retrieval using machine translation, relevance feedback and decompounding. *Information Retrieval*, 7(1).

Duan, H., Cao, Y., Lin, C., and Yu, Y. (2008). Searching questions by identifying questions topics and question focus. In *Annual Meeting of the Association for Computational Linguistics (ACL 2008)*, pages 156–164.

Jeon, J., Croft, W. B., and Lee, J. H. (2005). Finding similar questions in large question and answer archives. In *ACM Conference on Information and Managemenet (CIKM 2005)*, pages 84–90.

Kraaij, W., Nie, J., and Simard, M. (2003). Embedding web-based statistical translation models in cross-language information retrieval. *Computational Linguistics*, 29.

Lee, J. T., Kim, S. B., Song, Y. I., and Rim, H. C. (2008). Bridging lexical gaps between queries and questions on large online q&a collections with compact translation models. In *Empirical Methods on Natural Language Processing (EMNLP 2008)*, pages 410–418.

Lv, Y. and Zhai, C. (2010). Positional relevance model for pseudo-relevance feedback. In *ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2010)*.

Na, S. and Ng, H. (2011). Enriching document representation via translation for improved monolingual information retrieval. In *ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2011)*.

Riezler, S., Vasserman, A., Tsochantaridis, I., Mittal, V., and Liu, Y. (2007). Statistical machine translation for query expansion in answer retrieval. In *Annual Meeting of the Association for Computational Linguistics (ACL 2007)*, pages 464–471.

Singh, A. (2012). Entity based q&a retrieval. In *Empirical Methods on Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL 2012)*, pages 1266–1277.

Trieschnigg, D., Hiemstra, D., Jong, F., and Kraajj, W. (2010). A cross-lingual framework for monolingual biomeidcal information retrieval. In *ACM Conference on Information and Managemenet (CIKM 2010)*.

Wang, K., Ming, Z., and Chua, T.-S. (2009). A syntactic tree matching approach to finding similar questions in community-based qa services. In *ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2009)*, pages 187–194.

Xu, Y., Jones, G., and Wang, B. (2009). Query dependent pseudo-relevance feedback based on wikipedia. In *ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2009)*.

Xue, X., Jeon, J., and Croft, W. B. (2008). Retrieval models for question and answer archives. In *ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2008)*, pages 475–482.

Zhai, C. and Lafferty, J. (2001). A study of smooth methods for language models applied to ad hoc information retrieval. In *ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2001)*, pages 334–342.

Zhong, Z. and Ng, H. (2012). Word sense disambiguation improves information retrieval. In *Annual Meeting of the Association for Computational Linguistics (ACL 2012)*, pages 273–282.

Zhou, G., Cai, L., Zhao, J., and Liu, K. (2011). Phrase-based translation model for question retrieval in community question answer archives. In *Annual Meeting of the Association for Computational Linguistics (ACL 2011)*, pages 653–662.

Exploiting Lexical Dependencies from Large-Scale Data for Better Shift-Reduce Constituency Parsing

Muhua ZHU Jingbo ZHU Huizhen WANG
Natural Language Processing Laboratory
Northeastern University, China
zhumuhua@gmail.com, {zhujingbo,wanghuizhen}@mail.neu.edu.cn

Abstract

This paper proposes a method to improve shift-reduce constituency parsing by using lexical dependencies. The lexical dependency information is obtained from a large amount of auto-parsed data that is generated by a baseline shift-reduce parser on unlabeled data. We then incorporate a set of novel features defined on this information into the shift-reduce parsing model. The features can help to disambiguate action conflicts during decoding. Experimental results show that the new features achieve absolute improvements over a strong baseline by 0.9% and 1.1% on English and Chinese respectively. Moreover, the improved parser outperforms all previously reported shift-reduce constituency parsers.

Title and Abstract in Chinese

利用大规模数据词汇依存关系改进移进－归约成分句法分析

本文提出了一种利用词汇依存关系改进移进－归约成分句法分析的方法。首先，我们利用基准系统在大规模无标注数据上进行自动句法分析并从分析结果中抽取词汇依存关系。其后，我们在词汇依存信息的基础上定义了一组新特征并将这些特征整合到移进－归约句法分析模型 中。新特征用于帮助消除移进－归约过程中的动作歧义。实验结果表明，新特征在英文和中文数据上分别取得了0.9% 和1.1%的性能改进。最终得到的句法分析器的性能优于相关研究工作中所报告的移进－归约句法分析器的性能。

Keywords: Shift-reduce Constituency Parsing, Lexical Dependencies, Large-scale Data.

Keywords in Chinese: 移进－归约成分句法分析，词汇依存，大规模数据.

Proceedings of COLING 2012: Technical Papers, pages 3171–3186,
COLING 2012, Mumbai, December 2012.

1 Introduction

Due to the simplicity and running efficiency, shift-reduce parsing has been studied extensively for a variety of grammars, ranging from constituency parsing (Sagae and Lavie, 2005, 2006; Zhang and Clark, 2009) through dependency parsing (Nivre, 2004; Yamada and Matsumoto, 2003; Zhang and Clark, 2008) to CCG parsing (Zhang and Clark, 2011). In dependency and CCG parsing, shift-reduce parsing is among the best-performing algorithms (Huang and Sagae, 2010; Zhang and Clark, 2011). However, compared to commonly-used statistical parsers available on the web such as Charniak-Johnson (Charniak and Johnson, 2005) and Petrov-Klein (Petrov and Klein, 2007), shift-reduce constituency parsers still have room left for further improvements on parsing accuracy.

There exist at least two major directions to advance shift-reduce constituency parsing. One direction is to design better training and decoding algorithms. For example, in the respect of decoding, Sagae and Lavie (2006) proposed a best-first search strategy to expand the search space. In the respect of training, Zhang and Clark (2009) replaced local classifiers with a global learning algorithm. The other direction is to enrich feature representations for better shift-reduce constituency parsing, which will be the focus of this paper. In this direction, previous work has extensively studied a variety of features, all in the framework of supervised learning (Sagae and Lavie, 2005, 2006; Wang et al., 2006; Zhang and Clark, 2009).

How to further enrich feature representations for better shift-reduce constituency parsing becomes a very challenging problem. In this paper, we solve this issue by using the information of lexical head-modifier[1] relations (a.k.a. lexical dependencies) (Collins, 1996). Previous work on other constituency parsers have shown the effectiveness of lexical dependency information on disambiguating syntactic structures (Collins, 1996, 1997; Eisner and Satta, 1999). But in shift-reduce constituency parsing, such information is not fully used. For instance, Zhang and Clark (2009) completely neglected lexical dependency information. Sagae and Lavie (2005) and Wang et al. (2006) only incorporated as features the most recently recognized (left and right) modifiers of some designated words. Unlike previous work on shift-reduce constituency parsing, this paper aims to incorporate features that encode the information of whether words in an input sentence tend to have head-modifier relations. In addition, although it is feasible to get lexical dependency information from human-labeled treebank data by using head-finding rules (Collins, 1999), we find that lexical dependencies obtained from this source suffer from data sparseness (Section 5.1). We propose to solve this problem by utilizing additional large-scale unlabeled data.

The basic idea of our approach is to provide shift-reduce parsers with lexical dependency information that is obtained from large-scale auto-parsed data. To this end, we first parse unlabeled data with a baseline parser and afterwards extract bigram and trigram lexical dependencies from automatically parsed trees. Based on the extracted lexical dependencies, we finally design a set of features to enhance the baseline parser. The experiments in Section 5 show that new features can improve a strong baseline parser by 0.9% and 1.1% on English and Chinese data sets respectively. Moreover, our parser outperforms previously reported shift-reduce constituency parsers while maintaining efficiency.

Specifically, we make the following contributions in this paper:

- We propose a set of novel features for better shift-reduce constituency parsing that is based on lexical dependencies obtained from large-scale auto-parsed data;

[1] By 'head-modifier' we mean the linguistic notion that a word (*modifier*) modifies another word (*head*).

- We empirically compare two different sources for obtaining lexical dependencies: human-labeled treebank data and large-scale auto-parsed data respectively, and show the superiority of using auto-parsed data (Section 5.1);
- We empirically analyze major sources of shift-reduce parsing errors (Section 3.1) and verify the effectiveness of new features in resolving shift-reduce action conflicts (Section 5.6.2);

2 Baseline Parser

We use the beam-search shift-reduce parser (Zhang and Clark, 2009) as the baseline system in this paper. [2] In what follows, we describe the parser in brief.

2.1 The Shift-Reduce Parsing Process

The shift-reduce process in the baseline parser assumes binary-branching trees, so binarization and debinarization are required for transforming training data and parsing output, respectively (Zhang and Clark, 2009). Given an input sentence (words and POS tags), any possible parse tree yielding the sentence corresponds *exactly* to one sequence of states. Formally, each state in the sequence is denoted by a tuple $\langle S, Q \rangle$, where S is a stack containing partial parses and Q is a queue of word-POS pairs that remain unprocessed. In particular, the initial state is $\langle \phi, w_1 \dots w_n \rangle$ where S is empty and Q contains the entire input sentence. The final state is $\langle S, \phi \rangle$ where S contains a single parse tree with a pre-designated root label and Q is empty. Thus, the shift-reduce parsing process is a transition process from the initial state to the final state by performing a sequence of the following actions.

1. shift, which moves a pair of word and POS tag from the head of the queue to the stack. Here the queue is required to be non-empty.
2. reduce-unary-X, which extends the top item on the stack by applying a unary rule and then replaces the top item with the newly generated constituent. Here X represents a treebank phrase label, such as *NP*, which is to be used as the root label of the new constituent.
3. reduce-binary-{L/R}-X, which moves top two items out of the stack and pushes a new item onto the stack. The new item has X as its root label and consists of two children with the first popped item becoming the right child and the second popped item becoming the left child. The switch L/R indicates whether the left (L) or the right (R) child becomes the head child.
4. terminate, which pops the root node off the stack and ends parsing. This action is applicable only when the stack contains a single parse and the queue is empty.

2.2 Beam Search Extension

The shift-reduce parsing process described above can be extended with beam search, as presented in Algorithm 1. The algorithm starts by initializing a beam of size K with the initial state. In each iteration after the initialization, states are popped in turn out of the beam. For each popped state, all applicable actions are then evaluated with respect to the state. Scored action-state pairs are sorted in a temporary priority queue. When the beam gets empty, top K highest-scored action-state pairs are fetched from the priority queue and next states corresponding to the action-state pairs are inserted back into the beam. If the highest-scored state in the beam is a final state, it will be returned as the parsing result; else the iteration continues. The algorithm has time complexity of $O(nK)$, where n is the sentence length and K is the beam size.

[2] www.cl.cam.ac.uk/~yz360/zpar.html

Algorithm 1 Beam-search shift-reduce parsing

Input: a POS-tagged word sequence $w_1 \dots w_n$
beam size K and *action set*

```
1: B ← {⟨ϕ, w_1 ... w_n⟩}        // initialize beam
2: loop
3:     priority queue P = []
4:     while B not empty do
5:         state ← pop(B)
6:         for all act ∈ action set do
7:             score ← evaluate act for state
8:             P·insert (⟨score, act, state⟩)
9:     for i = 0 to K do
10:        ⟨score, act, state⟩ ← Pop-Top (P)
11:        next ← apply act to state
12:        insert next to B
13:    best ← highest-scored state in B
14:    if best is complete then
15:        return best
```

2.3 Model and Learning Algorithm

To score an action A with respect to a state $Y = \langle S, Q \rangle$, we use a linear model as defined by

$$Score(\langle A, Y \rangle) = \vec{w} \cdot \Phi(\langle A, Y \rangle) = \sum_i \lambda_i f_i(\langle A, Y \rangle)$$

where $f_i(\langle A, Y \rangle)$ are features extracted jointly from the action A and state Y. To learn parameters λ_i, we use the generalized perceptron algorithm proposed in Collins (2002).

Generalized perceptron is an online learning algorithm that learns one instance at a time. The basic procedure is to use the beam-search parsing algorithm (Algorithm 1) to parse the yield of a gold parse tree. Whenever the gold partial parse is pruned from the beam, parameters will be updated immediately and the learner moves to the next training instance. Such a strategy is known as "early-update" (Collins and Roark, 2004). Finally, model parameters are set to be an average of the weight vectors obtained during the online learning.

2.4 Baseline Features

Features used in the baseline parser are similar as those used in Zhang and Clark (2009). For convenience of reference, we repeat the features in Table 1, where the symbol S_i represents the i_{th} item from the top of the stack S and the symbol Q_i denotes the i_{th} item from the front end of the queue Q. The symbol w represents the lexical head for an item; c represents the label for an item; and t denotes POS of a lexical head. Note that Zhang and Clark (2009) also used bracket-related and separator features for Chinese parsing, which have been removed in the latest release of their parser. So in this article we choose to ignore such language specific features.

3 Our Approach

3.1 Motivation

We first empirically analyze major sources of shift-reduce parsing errors with the parsing results of the baseline parser on the English development set. The baseline parser is trained on human-labeled training data. Regarding the parsing results, we are especially concerned with *first mistakes* that the baseline parser makes because future mistakes are often caused by previous ones. There are

Description	Templates
Unigrams	$S_0tc, S_0wc, S_1tc, S_1wc, S_2tc, S_2wc, S_3tc, S_3wc,$ $Q_0wt, Q_1wt, Q_2wt, Q_3wt,$ $S_0lwc, S_0rwc, S_0uwc, S_1lwc, S_1rwc, S_1uwc$
Bigrams	$S_0wS_1w, S_0wS_1c, S_0cS_1w, S_0cS_1c,$ $S_0wQ_0w, S_0wQ_0t, S_0cQ_0w, S_0cQ_0t,$ $Q_0wQ_1w, Q_0wQ_1t, Q_0tQ_1w, Q_0tQ_1t,$ $S_1wQ_0w, S_1wQ_0t, S_1cQ_0w, S_1cQ_0t$
Trigrams	$S_0cS_1cS_2c, S_0wS_1cS_2c, S_0cS_1wQ_0t, S_0cS_1cS_2w,$ $S_0cS_1cQ_0t, S_0wS_1cQ_0t, S_0cS_1wQ_0t, S_0cS_1cQ_0w$

Table 1: A summary of baseline feature templates, where S_i represents the i_{th} item in stack S and Q_i denotes the i_{th} item in the queue Q from the front end.

ID	Mistake Type	Ratio (Count)
1	shift vs. red-binary	47.8% (451)
2	shift vs. red-unary	18.1% (171)
3	red-binary vs. red-unary	5.4% (92)
4	red-binary-L/R-{X vs. X*}	16.5% (156)
5	red-unary-{X_1 vs. X_2}	5.7% (54)

Table 2: Types and ratios of first mistakes made by the baseline parser on the English development set with auto-assigned POS.

944 first mistakes in total in the parsing results. Table 2 shows the types and ratios of the top 5 most frequent first mistakes. Cases 1-2 consist of conflicts between shift and reduce actions. Case 3 is comprised of conflicts between reduce-binary and reduce-unary actions. Mistakes in cases 4-5 are caused by wrong choices of labels where the symbols X, X_1, and X_2 refer to treebank phrase labels and the symbol X* denotes a temporary label which is introduced when a constituent with label *X* is binarized. From the table we notice that action conflicts between shift and reduce-binary are the largest source of parsing errors, which cover nearly half of first mistakes.

Intuitively, lexical dependency information is beneficial to resolving shift and reduce-binary conflicts. In the following, we use a real example to make clear the intuition. Figure 1 illustrates the shift-reduce parsing process of the baseline parser on a sentence with auto-assigned POS tags. The baseline parser proceeds correctly until it reaches the state in Figure 1-(a). At that point, there is a conflict between reduce-binary (Figure 1-(b)) and shift (Figure 1-(c)) actions. We find that the baseline parser wrongly chooses the reduce-binary action because the word *bore* is (incorrectly) tagged as a verb.[3] The baseline parser tends to group the words preceding a verb as a constituent. However, if the parser is informed that the words *a* and *bore* have a lexical dependency relationship, the parser may correct its choice and switch to the shift action. In addition, we find that the human-labeled data used to train the baseline parser does not contain lexical dependencies between *a* and *bore*. We can see that extracting lexical dependencies solely from human-labeled training data has a data sparseness problem. This motivates us to utilize unlabeled data as an additional source for lexical dependency extraction.

[3]All the occurrences of *bore* in the training data of the POS tagger have the POS tag *VBD*.

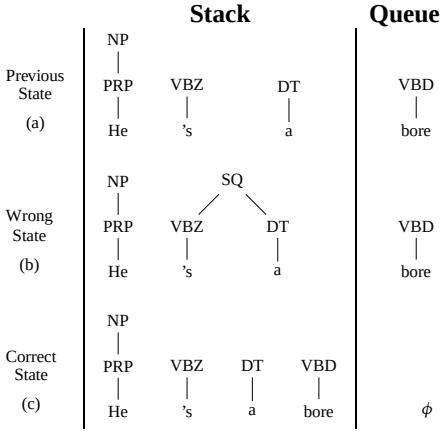


Figure 1: An example of shift-reduce conflicts, illustrating how lexical dependency information helps to disambiguate the conflicts.

3.2 Data Preprocessing

Before conducting lexical dependency extraction, we use the baseline parser to generate constituency parse trees from unlabeled data. Since shift-reduce parsers require POS tags as input, automatical POS tagging should be performed on unlabeled data before performing syntactic parsing. For unspaced languages such as Chinese, automatical word segmentation is also needed. To simplify the extraction process, we convert automatically parsed constituency trees into dependency trees with Penn2Malt (or other conversion tools).[4]

3.3 Extraction of Lexical Dependencies

After the tree conversion, the following lexical dependencies are read off from dependency trees. Here we restrict the dependencies to those between two words (bigram lexical dependencies) and those between three words (trigram lexical dependencies).

Bigram Lexical Dependencies

If two words are connected by an arc in a dependency tree, we claim these two words maintain a bigram lexical dependency. For pairs of words that have dependencies, we make a record of the words as well as their head-modifier relations. Formally, bigram lexical dependencies are denoted as $\langle w_1, w_2, L/R \rangle$ where L/R indicates the direction of the dependency arc that connects w_1 and w_2. Moreover, lexical dependencies are word-order sensitive, that is, $\langle w_1, w_2, L \rangle$ is regarded to be different from $\langle w_2, w_1, R \rangle$.

Trigram Lexical Dependencies

Trigram lexical dependencies encode a head-modifier relationship among three words. As with bigram lexical dependencies, trigram lexical dependencies are also word-order sensitive. In this

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

Bigram Dependency Features			
$f_L(s_1w, s_0w)$	$f_L(s_1w, s_0w) \circ s_1t \circ s_0t$	$f_R(s_1w, s_0w)$	$f_R(s_1w, s_0w) \circ s_1t \circ s_0t$
$f_L(s_1w, q_0w)$	$f_L(s_1w, q_0w) \circ s_1t \circ q_0t$	$f_R(s_1w, q_0w)$	$f_R(s_1w, q_0w) \circ s_1t \circ q_0t$
$f_L(s_0w, q_0w)$	$f_L(s_0w, q_0w) \circ s_0t \circ q_0t$	$f_R(s_0w, q_0w)$	$f_R(s_0w, q_0w) \circ s_0t \circ q_0t$
Trigram Dependency Features			
$f_L(s_1w, s_1rdw, s_0w)$	$f_L(s_1w, s_1rdw, s_0w) \circ s_1t \circ s_0t$	$f_R(s_1w, s_0ldw, s_0w)$	$f_R(s_1w, s_0ldw, s_0w) \circ s_1t \circ s_0t$
$f_L(s_0w, s_0rdw, q_0w)$	$f_L(s_0w, s_0rdw, q_0w) \circ s_0t \circ q_0t$	$f_R(s_0w, NONE, q_0w)$	$f_R(s_0w, NONE, q_0w) \circ s_0t \circ q_0t$

Table 3: New features designed on the basis of lexical dependencies. Here the symbol w represents a word and the symbol t represents a POS tag.

paper, we only consider the type of trigram lexical dependencies that have the first or the last word be the head and that require the other two words to be siblings among all the modifiers of the head. Such lexical dependencies can be represented formally as $\langle w_1, w_2, w_3, L/R \rangle$. Here the switch L/R indicates the head among the three words. Specifically, the symbol L specifies w_1 to be the head and the symbol R designates w_3 to be the head. In addition, we also consider the special case that w_2 is *NONE*, which indicates that w_1 (w_3) is the rightmost (leftmost) modifier of w_3 (w_1).

3.4 Proposed Features

After extracting all lexical dependencies, we group bigram and trigram lexical dependencies *separately* into three categories according to their frequencies. Specifically, if a dependency relation is among top-10% most frequent records, then it receives the group tag *High Frequency (HF)*; else if it is in top-20%, then we use the tag *Middle Frequency (MF)*; else we use the tag *Low Frequency (LF)*. Although such a grouping strategy is heuristic in some sense, it has been proven effective in Chen et al. (2009). After the grouping, we finally get two lists, containing bigram and trigram lexical dependencies respectively.

Based on the bigram and trigram lexical dependency lists, we propose a set of dependency features which is described in detail in the following. Here s_i denotes the i_{th} item from the top of the stack S, and q_i the i_{th} item from the front end of the queue Q. In addition, s_iw (s_it) refers to the head word (POS) of s_i and q_iw (q_it) refers to the word (POS) of q_i.

3.4.1 Bigram Dependency Features

Bigram dependency features have a generic form of $f_{L/R}(w_1, w_2)$ which returns a group tag (HF, MF, or LF) if the lexical dependency $\langle w_1, w_2, L/R \rangle$ is found in the bigram lexical dependency list; else it returns *NULL*. The above feature template is instantiated into three pairs of features: $\{f_L(s_1w, s_0w), f_R(s_1w, s_0w)\}$, $\{f_L(s_0w, q_0w), f_R(s_0w, q_0w)\}$, and $\{f_L(s_1w, q_0w), f_R(s_1w, q_0w)\}$.

We also combine the above features with POS tags of w_1 and w_2. Thus we have three more pairs of features in the generic form of $f_{L/R}(w_1, w_2) \circ t(w_1) \circ t(w_2)$, where $t(w_i)$ represents the POS tag of the word w_i. All the bigram dependency features are listed in Table 3.

3.4.2 Trigram Dependency Features

Trigram dependency features have the generic form of $f_{L/R}(w_1, w_2, w_3)$. In this paper, this feature template is instantiated into two pairs of features. The feature function $f_L(s_1w, s_1rdw, s_0w)$ returns a group tag if $\langle s_1w, s_1rdw, s_0w, L \rangle$ is found in the trigram lexical dependency list, where s_1rdw denotes the rightmost modifier of s_1w that has been recognized so far during the shift-reduce parsing process. Note that s_1rdw might be *NONE* if no right modifiers have been recognized for s_1w. The

other trigram dependency features, $f_R(s_1w, s_0ldw, s_0w)$, $f_L(s_0w, s_0rdw, q_0w)$, and $f_R(s_0w$, *NONE*, $q_0w)$ can be explained in a similar way. As with bigram dependency features, POS tags are combined with above features to obtain richer feature representations. Trigram dependency features used in the paper are summarized in Table 3.

3.5 Parsing with Proposed Features

To use the proposed dependency features, we only need to update the scoring function defined in Section 2.3. The new scoring function is shown in the following.

$$Score'(\langle A, Y\rangle) = \sum_i \lambda_i f_i(\langle A, Y\rangle) + \sum_j \lambda_j^d f_j^d$$

where $f_i(\langle A, Y\rangle)$ are the baseline features listed in Table 1 and f_j^d refer to the dependency features defined above.

4 Experimental Setup

4.1 Data Preparation

For English experiments, our labeled data came from the Wall Street Journal (WSJ) corpus of the Penn Treebank (Marcus et al., 1993). We used the standard divisions: sections 2-21 were used as training data, section 24 was used for system development, and section 23 was held out for performance evaluation. In terms of English unlabeled data, we used the TIPSTER corpus (LDC93T3A) which contains news articles from various sources, though in this paper we only used Wall Street Journal articles. In addition, we did not remove the sentences of the WSJ portion of the Penn Treebank from the TIPSTER corpus.

For Chinese experiments, we used Chinese Treebank (CTB) version 5.1 (Xue et al., 2005) as labeled data. Specifically, articles 001-270 and 440-1151 were used as training data, articles 271-300 were held out for performance evaluation, and articles 301-325 were used as development data. In the respect of Chinese unlabeled data, we utilized the corpus of Chinese Gigaword (LDC2003T09) after some basic cleanups.

We conducted necessary preprocessing on English and Chinese unlabeled data before they were fed to the baseline parser for automatic parsing. Specifically, we applied OpenNLP for English sentence boundary detection and tokenization.[5] For English POS tagging, SVMTool was used which achieves a per-token accuracy of 97.1% on section 23 of the WSJ corpus.[6] For the Chinese unlabeled data, we conducted sentence boundary detection simply according to sentence ending punctuations, including question marks, full stop marks, and exclamation marks. Raw sentences were automatically segmented with a CRF-based word segmenter which achieves a segmentation accuracy of 97.2% on the testing data of CTB 5.1. For automatic Chinese POS tagging, we utilized the Stanford POS tagger. [7] We trained the tagger on the CTB 5.1 training data and achieved a tagging accuracy of 95.4% on the CTB 5.1 testing data.

Table 4 contains detailed data statistics of all the above corpora.

[5] `http://incubator.apache.org/opennlp/`
[6] `http://www.lsi.upc.edu/~nlp/SVMTool/`
[7] `http://nlp.stanford.edu/software/tagger.shtml`

Language	Statistics	Train	Dev	Test	Unlabeled
English	# sentences	39.8k	1.7k	2.4k	3,139.1k
	# words	950.0k	40.1k	56.7k	76,041.4k*
	# ave. length	28.9	25.1	25.1	25.22*
Chinese	# sentences	18.1k	350	348	11,810.7k
	# words	493.8k	8.0k	6.8k	269,057.2k*
	# ave. length	27.3	19.5	23.0	22.8*

Table 4: Data statistics including the number of words and sentences, together with average sentence length. * The numbers are approximate due to the use of automatic preprocessing techniques.

	English			Chinese		
Data Source	**LR**	**LP**	**F1**	**LR**	**LP**	**F1**
Baseline	88.2	88.2	88.2	83.7	84.4	84.0
Human-Labeled	88.5	88.7	88.6	83.8	84.4	84.1
Auto-Parsed	89.1	89.4	89.3	85.2	85.1	85.1
Combined	89.3	89.5	89.4	85.3	85.2	85.2

Table 5: Comparative results on English and Chinese developments sets with lexical dependencies extracted from diverse sources.

4.2 Performance Scoring

For performance evaluation in all the following experiments, we used *EVALB* to provide bracket scoring as well as complete match scoring.[8] For significance tests, we adopted the comparator developed by Daniel Bikel to compute *p-value*.[9]

4.3 Running Parameters

We set the beam size to 16 in both training and decoding which maintains a good trade-off between parsing efficiency and accuracy (Zhang and Clark, 2009). With respect to the iteration number of perceptron learning, we tuned the parameter on the English and Chinese development sets and finally set the value to 21 for both English and Chinese experiments.

5 Experimental Results

5.1 Comparison of Different Sources

Table 5 shows the comparative results on English and Chinese development sets with lexical dependencies obtained from different sources. We experimented with four different settings: no lexical dependency information was used (Baseline), lexical dependencies were solely from human-labeled training data (Human-Labeled), lexical dependencies were solely from auto-parsed data (Auto-Parsed), and lexical dependencies were from the combination of human-labeled and auto-parsed data (Combined). For the latter three settings, all the dependency features listed in Table 3 were incorporated. In the data combination, we simply gave our human-labeled training data a relative weight of one.

From the results we can see that, although lexical dependency information from human-labeled training data can improve the performance on both English and Chinese, the improvement on Chi-

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

[9] http://www.cis.upenn.edu/~dbikel/download/compare.pl

Sentences with #Words ≤ 40				
Features	**LR**	**LP**	**F1**	**EX**
Baseline	90.1	89.8	90.0	41.3
+Bigram Features	90.6	90.5	90.6	42.1
+Trigram Features	90.9	90.9	90.9	42.9
Sentences with Unlimited Words				
Features	**LR**	**LP**	**F1**	**EX**
Baseline	89.6	89.4	89.5	39.0
+Bigram Features	90.1	90.1	90.1	39.7
+Trigram Features	90.4	90.5	90.4	40.7

Table 6: Main results on section 23 of the WSJ corpus, using automatically assigned POS tags and lexical dependencies extracted from auto-parsed data. Two types of dependency features are added **incrementally**.

nese is marginal (+0.1% in F1-score). One reason is that lexical dependencies from human-labeled training data have a data sparseness problem. By contrast, the use of large-scale auto-parsed data brings on much bigger improvements (+1.1% in F1-score on both English and Chinese). In addition, we find that data combination has trivial effect on the performance. One possible reason is that lexical dependencies form human-labeled training data are overwhelmed by those from auto-parsed data. We will leave further discussions on data combination to our future work and focus on the setting of obtaining lexical dependencies from auto-parsed data.

5.2 Main Results on English Data

Table 6 shows the main results on the English test set, where the two types of dependency features were added incrementally to the baseline parser. As the results show, both bigram and trigram dependency features have positive effect on the parsing accuracy. Specifically, on the whole test set the overall improvement over the baseline parser is +0.9% in F1-score, where bigram dependency features contribute an absolute 0.6% improvement and trigram dependency features further improve the performance by 0.3% over the results of using bigram dependency features. Significance tests show that the overall improvement on the whole test set is statistically significant on the level of $p < 10^{-4}$.

5.3 Comparative Results on English

Table 7 shows the comparison of our parser with a large body of representative related work. For fair comparison, here we disregarded parsers that are based on combination methods such as Petrov (2010) and Zhang et al. (2009). Following the taxonomy adopted in Huang et al. (2010), we grouped the related work into single parsers (SINGLE), discriminative reranking approaches (RE), and self-training (SELF). Note that our parser belongs to the category of self-training. From the results we can see that our parser outperforms all the single parsers listed in the table except Carreras et al. (2008). However, compared with Carreras et al. (2008), our parser has much smaller time-complexity: $O(nK)$ vs. $O(n^3G)$, where K is the beam size used in our parser and G is a grammar constant in Carreras et al. (2008). Compared with reranking and self-training parsers, our parser has relatively low parsing accuracy. But the reranking technique is actually complementary with our approach, so we might enhance our parser with this technique in the future.

Type	Parser		LR	LP	F1
SINGLE	Ratnaparkhi (1997)		86.3	87.5	86.9
	Collins (1999)		88.1	88.3	88.2
	Charniak (2000)		89.5	89.9	89.5
	Sagae and Lavie (2005)*		86.1	86.0	86.0
	Sagae and Lavie (2006)*		87.8	88.1	87.9
	Petrov and Klein (2007)		90.1	90.2	90.1
	Carreras et al. (2008)		90.7	91.4	91.1
RE	Charniak and Johnson (2005)		91.2	91.8	91.0
	Huang (2008)		92.2	91.2	91.7
SELF	Huang and Harper (2009)		91.1	91.6	91.3
	McClosky et al. (2006)		92.1	92.5	92.3
	Huang et al. (2010)†		91.4	91.8	91.7
	This Paper	Baseline	89.6	89.4	89.5
		Auto-Parsed	90.4	90.5	90.4

Table 7: Comparison with related work on section 23 of the WSJ corpus with automatically assigned POS tags. * The parsers based on shift-reduce parsing. † The results of self-training with a single latent annotation grammar.

Sentences with #Words≤ 40

Features	LR	LP	F1	EX
Baseline	82.9	83.6	83.2	32.8
+Bi-lexical Features	84.3	85.1	84.7	32.8
+Tri-lexical Features	84.7	85.9	85.3	34.1
Sentences with Unlimited Words				
Features	**LR**	**LP**	**F1**	**EX**
Baseline	79.5	80.7	80.1	28.2
+Bi-lexical Features	80.3	81.6	80.9	28.2
+Tri-lexical Features	80.6	81.9	81.2	29.3

Table 8: Main results on the CTB 5.1 test set, using automatically assigned POS tags and lexical dependencies extracted from auto-parsed data. Two types of dependency features are added **incrementally**.

5.4 Main Results on Chinese Data

In parallel to the results on the English test set, Table 8 shows the main results on the Chinese test set, using auto-assigned POS tags and lexical dependencies extracted from auto-parsed data. From the results on the whole test set we can see that dependency features contribute a bigger absolute improvement on Chinese than that on English ($+1.1\%$ vs. $+0.9\%$). One possible reason is that the size of Chinese unlabeled data used in the paper is much bigger. Significance tests show that the overall improvement induced by bigram and trigram dependency features on the whole test is statistically significant on the level of $p < 10^{-3}$. These results indicate that the new features are very effective.

5.5 Comparative Results on Chinese

Comparing Chinese constituency parsers is difficult in the sense that previously reported results were achieved frequently on different versions of CTB and/or with different data split standards. Zhang and Clark (2009) presented a detailed comparison between the baseline parser of this paper and a large body of related work on CTB 2.0. Here we only compared our parser with the parsers available on the web for Chinese parsing, as shown in Table 9. From the results we can see

Type	Parser		LP	LR	F1
SINGLE	Charniak (2000)*		79.6	82.1	80.8
	Bikel (2004)†		79.3	82.0	80.6
	Petrov and Klein (2007)		81.9	84.8	83.3
RE	Charniak and Johnson (2005)*		80.8	83.8	82.3
	This Paper	Baseline	79.5	80.7	80.1
		Auto-Parsed	80.6	81.9	81.2

Table 9: Comparison with related work on the test set of CTB 5.1 with automatically assigned POS tags. * Huang (2009) adapted the parsers to Chinese parsing on CTB 5.1. † We run the parser on CTB 5.1 to get the results.

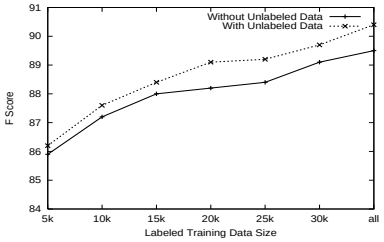
that, on Chinese parsing our parser outperforms Bikel (2004) and Charniak (2000) by 0.6% and 0.4%, respectively. However, our parser lags behind Petrov and Klein (2007) and the reranking parser (Charniak and Johnson, 2005).

5.6 Additional Analysis

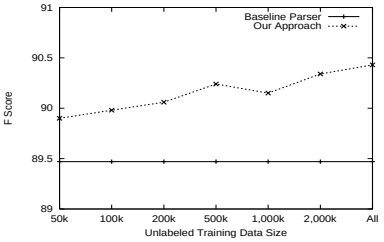
We performed several types of analysis, focusing on English, to investigate the effect of different sizes of human-labeled training data and auto-parsed data, as well as how lexical dependency information changes the distribution of first mistakes made by the baseline parser.

5.6.1 Effect of Different Sizes of Labeled and Unlabeled Data

We studied the effect of varying sizes of human-labeled training data by a randomly sampling from sections 2-21. Meanwhile, we always used the whole set of auto-parsed data. The results are depicted in Figure 2-(a). As the results show, auto-parsed data improves parsing accuracy even when the human-labeled training data is small in size. In addition, by using our approach, a fraction of sections 2-21 plus the whole set of auto-parsed data is sufficient to achieve the F1-scores obtained by the parser trained solely on the whole sections 2-21.



(a) varying labeled training data



(b) varying unlabeled data

Figure 2: Results with varying sizes of human-labeled training data and unlabeled data.

We also examined the effect of varying sizes of unlabeled data. In this experiment, we used sections 2-21 as human-labeled training data and changed the size of unlabeled data through random sampling. The results are depicted in Figure 2-(b). From the results we can see that improvements

achieved by using unlabeled data are enlarged with the increment of the size of unlabeled data until the performance finally levels off.

5.6.2 Reduction on First Mistakes

The baseline parser made 1,522 first mistakes on the English test set. We analyzed how our approach changed the first mistakes. We grouped the changes into four cases. *No-Change* (1,090) refers to the case that the baseline and our parser make the same first mistakes at the same positions. In the case of *Correct* (249), first mistakes made by the baseline parser are corrected by our parser. *Wrong* (121) means that our parser makes first mistakes earlier than the baseline parser. Finally, *Others* (47) refers to the case that our parser and the baseline parser make first mistakes of different types at the same positions. In addition, we are especially interested in how our approach reduces first mistakes of the type *shift vs. reduce-binary*. So we compared the numbers of first mistakes of this type in the *Correct* and *Wrong* cases, which are 168 and 78 respectively.

6 Related Work

Shift-reduce parsing has been widely studied for constituency parsing. Sagae and Lavie (2005) proposed a classifier-based shift-reduce parsing algorithm which was extended with a best-first search strategy in Sagae and Lavie (2006). Wang et al. (2006) adapted the parser in Sagae and Lavie (2005) to Chinese parsing and compared some representative classifiers. Zhang and Clark (2009) proposed a global learning algorithm to replace local classifiers. Shift-reduce parsing has also widely applied to parsing with other grammars (Nivre, 2004; Zhang and Clark, 2008; Huang and Sagae, 2010; Zhang and Clark, 2011). In this paper we focus on improving Zhang and Clark (2009) with a set of novel features defined on lexical dependencies obtained from auto-parsed data. To the best of our knowledge, semi-supervised approaches to shift-reduce constituency parsing have not been widely studied before.

The approach used in this paper belongs to the category of semi-supervised learning. In the respect of semi-supervised learning for constituency parsing, self-training has been extensively studied (McClosky et al., 2006; Huang and Harper, 2009; Huang et al., 2010). The difference is that we use partial information derived from auto-parsed data instead of entire automatically parsed trees. Chen et al. (2009) and Noord (2007) exploited lexical dependencies from unlabeled data for dependency and HPSG parsing, respectively. In this paper we for the first time use lexical dependency information for advancing the state-of-the-art shift-reduce constituency parsers. It is noteworthy that, although Chen et al. (2009) and our work use the same strategy to extract lexical dependencies, features defined on lexical dependencies are distinct. Specifically, Chen et al. (2009) proposed features for a graph-based dependency parser while this paper focuses on a transition-based constituency parser. More recently, Bansal and Klein (2011) proposed features for both dependency and constituency parsing based on Web counts from the Google n-grams corpus. By contrast, lexical dependency information used in this paper is derived from auto-parsed data. Moreover, Web counts from the Google n-grams corpus represent surface evidence of lexical affinities while lexical dependencies are able to encode information on deep syntactic structures.

7 Conclusion

We have presented a method to utilize lexical dependency information to improve shift-reduce constituency parsing. A set of new features was proposed based on the lexical dependency information and integrated into the shift-reduce parser. Our method well addressed the action conflict problem. We evaluated the proposed method on English and Chinese data. The results show that our

new parsers provide comparable accuracies with state-of-the-part parsers while maintaining the advantage in parsing speed.

Acknowledgments

We would like to thank Wenliang Chen for his help in extracting lexical dependencies and discussions on designing features. This work was supported in part by the National Science Foundation of China (61073140, 61272376, 61100089, 61003159), Specialized Research Fund for the Doctoral Program of Higher Education (20100042110031) and the Fundamental Research Funds for the Central Universities (N110404012).

References

Bansal, M. and Klein, D. (2011). Web-scale features for full-scale parsing. In *the 49th Annual Meeting of the Association for Computational Linguistics (ACL 2011)*, pages 693–702.

Bikel, D. M. (2004). On the parameter space of generative lexicalized statistical parsing models. In *Ph.D. thesis, University of Pennsylvania*.

Carreras, X., Collins, M., and Koo, T. (2008). Tag, dynamic programming and the perceptron for efficient, feature-rich parsing. In *Conference on Computational Natural Language Learning (CoNLL 2008)*, pages 9–16.

Charniak, E. (2000). A maximum-entropy-inspired parser. In *Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL 2000)*, pages 132–139.

Charniak, E. and Johnson, M. (2005). Coarse-to-fine n-best parsing and maxent discriminative reranking. In *the 43rd Annual Meeting of the Association for Computational Linguistics (ACL 2005)*, pages 173–180.

Chen, W., Kazama, J., Uchimoto, K., and Torisawa, K. (2009). Improving dependency parsing with subtrees from auto-parsed data. In *the 2009 Conference on Empirical Methods in Natural Language Processing (EMNLP 2009)*, pages 570–579.

Collins, M. (1996). A new statistical parser based on bigram lexical dependencies. In *the 34th Annual Meeting of the Association for Computational Linguistics (ACL 1996)*.

Collins, M. (1997). Three generative, lexicalised models for statistical parsing. In *the 35th Annual Meeting of the Association for Computational Linguistics (ACL 1997)*.

Collins, M. (1999). Head-driven statistical models for natural language parsing. In *Ph.D. thesis, University of Pennsylvania*.

Collins, M. (2002). Discriminative training methods for hidden markov models: theory and experiemnts with perceptron algorithm. In *the 2002 Conference on Empirical Methods in Natural Language Processing (EMNLP 2002)*, pages 1–8.

Collins, M. and Roark, B. (2004). Incremental parsing with the perceptron algorithm. In *the 32rd Annual Meeting of the Association for Computational Linguistics (ACL 2004)*.

Eisner, J. and Satta, G. (1999). Efficient parsing for bilexical context-free grammars and head automaton grammars. In *the 37th Annual Meeting of the Association for Computational Linguistics (ACL 1999)*.

Huang, L. and Sagae, K. (2010). Dynamic programming for linear-time incremental parsing. In *the 48th Annual Meeting of the Association for Computational Linguistics (ACL 2010)*, pages 1077–1086.

Huang, L.-Y. (2009). Improve Chinese parsing with Max-Ent reranking parser. In *Master Project Report, Brown University*.

Huang, Z. and Harper, M. (2009). Self-training PCFG grammars with latent annotations across languages. In *the 2009 Conference on Empirical Methods in Natural Language Processing (EMNLP 2009)*, pages 832–841.

Huang, Z., Harper, M., and Petrov, S. (2010). Self-training with products of latent variable grammars. In *the 2010 Conference on Empirical Methods in Natural Language Processing (EMNLP 2010)*, pages 12–22.

Marcus, M. P., Santorini, B., and Marcinkiewiz, M. A. (1993). Building a large annotated corpus of English. In *Computational Linguistics, 19(2)*, pages 313–330.

McClosky, D., Charniak, E., and Johnson, M. (2006). Reranking and self-training for parser adaptation. In *the 44th Annual Meeting of the Association for Computational Linguistics and 21st International Conference on Computational Linguistics (ACL-COLING 2006)*, pages 337–344.

Nivre, J. (2004). Incrementality in deterministic dependency parsing. In *Incremental Parsing: Bringing Engineering and Cognition Together: Workshop at ACL 2004*.

Noord, G. (2007). Using self-trained bilexical preferences to improve disambiguation accuracy. In *the 10th International Conference on Parsing Technologies (IWPT 2007)*.

Petrov, S. (2010). Products of random latent variable grammars. In *Human Language Technologies: The 11th Annual Conference of the North American Chapter of the Association for Computational Linguistics (HLT-NAACL 2010)*, pages 19–27.

Petrov, S. and Klein, D. (2007). Improved inference for unlexicalized parsing. In *Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL 2007)*, pages 404–411.

Ratnaparkhi, A. (1997). A linear observed time statistical parser based on maximum entropy models. In *the 1997 Conference on Empirical Methods in Natural Language Processing (EMNLP 1997)*.

Sagae, K. and Lavie, A. (2005). A classifier-based parser with linear run-time complexity. In *the 9th International Conference on Parsing Technologies (IWPT 2005)*, pages 125–132.

Sagae, K. and Lavie, A. (2006). A best-first probabilistic shift-reduce parser. In *the 44th Annual Meeting of the Association for Computational Linguistics and 21st International Conference on Computational Linguistics (ACL-COLING 2006)*, pages 691–698.

Wang, M., Sagae, K., and Mitamura, T. (2006). A fast, accurate deterministic parser for Chinese. In *the 44th Annual Meeting of the Association for Computational Linguistics and 21st INternational Conference on Computational Linguistics (ACL-COLING 2006)*, pages 25–32.

Xue, N., Xia, F., dong Chiou, F., and Palmer, M. (2005). The Penn Chinese treebank: phrase structure annotation of a large corpus. In *Natural Language Engineering, 11(2)*, pages 207–238.

Yamada, H. and Matsumoto, Y. (2003). Statistical dependency analysis with support vector machines. In *the 8th International Conference on Parsing Technologies (IWPT 2003)*, pages 195–206.

Zhang, H., Zhang, M., Tan, C. L., and Li, H. (2009). K-best combination of syntactic parsers. In *the 2009 Conference on Empirical Methods in Natural Language Processing (EMNLP 2009)*, pages 1552–1560.

Zhang, Y. and Clark, S. (2008). A tale of two parsers: investigating and combining graph-based and transition-based dependency parsing. In *the 2008 Conference on Empirical Methods in Natural Language Processing (EMNLP 2008)*, pages 562–571.

Zhang, Y. and Clark, S. (2009). Transition-based parsing of the Chinese treebank using a global discriminative model. In *the 11th International Conference on Parsing Technologies (IWPT 2009)*, pages 162–171.

Zhang, Y. and Clark, S. (2011). Shift-reduce CCG parsing. In *the 49th Annual Meeting of the Association for Computational Linguistics (ACL 2011)*, pages 683–692.